Distributed Video Coding with Geometric Transforms

Pedro André Faria Monteiro

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Examination Committee

Chairperson: Prof. Fernando Duarte Nunes
Supervisor: Prof. Fernando Manuel Bernardo Pereira
Co-Supervisor: Prof. João Miguel Duarte Ascenso

Members of the Committee: Prof. Nuno Miguel Morais Rodrigues

March 2013
Acknowledgments

First, I would like to thank Professor Fernando Pereira, my supervisor, for the constant guidance, patience and enthusiasm. It was a unique privilege working under his supervision. The outstanding working methodology and organization, the profound devotion and dedication, the welcoming environment and the exceptional advices and suggestions significantly influenced the outcome of this Thesis.

I would also like to express my gratitude to Professor João Ascenso, my co-supervisor, for the constant availability he showed to address my question and provide meaningful and important advices, even when it interfered with his schedule. His technical and conceptual knowledge really helped me and were essential to the work developed in this Thesis.

Due to Professor Fernando Pereira and Professor João Ascenso unconditional help and support, this Thesis was, by far, the most remarkable academic experience I have ever had.

I am also thankful to Professor Catarina Brites and PhD student Diego Felix, for always being available to help. Big thanks to the multimedia signal processing research group and to Instituto de Telecomunicações for providing me with excellent working conditions and a welcoming and friendly environment.

I would also like to thank my family for giving me all the possible conditions to get me to this stage, for always trusting my decisions and for the unconditional support.

I would also like to express my deepest gratitude to my beloved girlfriend Madalena Leitão, for the immense support and motivation during my entire academic journey. Her constant patience and emotional support were really important to conclude this Thesis. Thanks for always being there for me.

Finally, I would like to thank my friends that help me get this far in my academic life.

Thank you all.
Abstract

The Distributed Video Coding (DVC) paradigm is based on two well-known information theory results: the Slepian-Wolf and the Wyner-Ziv theorems. In a DVC codec, the correlation between video signals is exploited at the decoder, providing a flexible distribution of the computational complexity between the encoder and the decoder and an error robustness to the channel errors, since the rate control is performed at the decoder. To exploit the correlation between frames in a DVC codec, a translational motion model is typically used. However, this model is not accurate enough for complex motion such as rotations and zooms. In this Thesis, there are proposed two geometric transforms based motion models to generate the side information in a distributed video codec. The side information is an estimation of the original frame to code created at the decoder.

After reviewing some relevant geometric transform based video coding solutions and side information creation techniques available in the literature, two SI creation solutions were designed, implemented and assessed: the Unidirectional Warping Side information (UWSI) and the Bidirectional Warping Side information (BWSI). While the UWSI solution performs a motion estimation with geometric transforms, the BWSI uses an initial bidirectional search with geometric transforms and, later, the same motion estimation with geometric transforms. Also, the BWSI has the possibility of using a unidirectional motion compensation mode to improve the side information quality when occlusions occur.

Experimental results show PSNR gains of up to 1 dB in side information quality and 0.66 dB in Rate-Distortion performance for some video sequences.

Keywords: Distributed Video Coding; Side Information; Geometric Transform.
Resumo

A Codificação Distribuída de Vídeo (CDV) é um paradigma baseado em dois resultados da Teoria da Informação: os teoremas de Slepian-Wolf e Wyner-Ziv. Num codec CDV, a correlação entre sinais de vídeo é explorada no descodificador, o que oferece uma flexibilidade em distribuir a complexidade computacional entre o codificador e descodificador e uma resiliência aos erros do canal. Para explorar a correlação entre tramas num codec CDV, um modelo de movimento translacional é tipicamente usado. No entanto, este modelo não é suficientemente exacto para descrever movimentos mais complexos, como rotações e zooms. Esta Tese propõe dois modelos de movimento baseados em transformadas geométricas para gerarem side information num codec CDV. A side information é uma estimação da trama original criada no descodificador.

Depois de rever algumas soluções baseadas em transformadas geométricas e algumas técnicas de criação de side information disponíveis na literatura, dois módulos de criação de side information são propostos: Unidirectional Warping Side information (UWSI) e Bidirectional Warping Side information (BWSI). Enquanto que a solução UWSI usa uma estimação de movimento baseada em transformadas geométricas, a solução BWSI usa uma estimação inicial bidireccional baseada em transformadas geométricas e, depois, a mesma estimação de movimento bidireccional com transformadas geométricas. A solução BWSI contém ainda a possibilidade de usar um modo de compensação de movimento unidireccional para melhorar a qualidade da side information quando ocorrem oclusões.

Os resultados experimentais demonstraram ganhos PSNR até 1 dB em termos de qualidade da side information e 0.66 dB em desempenho débito-distorção para algumas sequências de vídeo.

Palavras-Chave: Codificação Distribuída de Vídeo; Side Information; Transformadas Geométricas.
# Table of Contents

Chapter 1  Introduction ........................................................................................................... 1  
1.1 Context and Motivation .................................................................................................. 1  
1.2 Objectives .................................................................................................................... 3  
1.3 Thesis Structure .......................................................................................................... 3  

Chapter 2  Geometric Transforms and Video Coding: Reviewing the Relevant Background ...... 5  
2.1 Reviewing Geometric Transforms based Predictive Video Coding ............................... 5  
   2.1.1 Basics on Geometric Transforms ........................................................................... 5  
      2.1.1.1 Affine Transform ............................................................................................. 7  
      2.1.1.2 Perspective Transform .................................................................................. 8  
      2.1.1.3 Bilinear Transform ...................................................................................... 9  
      2.1.1.4 Geometric Transforms Comparison ............................................................. 10  
   2.1.2 Predictive Video Coding: Basics and State-of-the-Art ........................................... 10  
      2.1.2.1 Basics ........................................................................................................... 10  
      2.1.2.2 State-of-the-Art ........................................................................................... 13  
   2.1.3 Most Relevant Predictive Video Coding with Geometric Transforms Solutions ...... 15  
      2.1.3.1 Picture-Level Parametric Motion Representation for Efficient Motion Compensation .................................................................................................................. 15  
      2.1.3.2 A Block-Adaptive Skip Mode for Inter Prediction Based on Parametric Motion Models .................................................................................................................. 20  

2.2 Reviewing Distributed Video Coding ............................................................................ 25  
   2.2.1 Basic Definitions and Theorems .......................................................................... 25  
   2.2.2 DISCOVER WZ Codec: a Concise Description ...................................................... 26  
   2.2.3 Most Relevant Side Information Creation Methods .............................................. 29  
      2.2.3.1 Advanced side information creation techniques and framework for Wyner-Ziv video coding .................................................................................................................. 30  
      2.2.3.2 Mesh-Based Motion-Compensated Interpolation for Side Information Extraction in Distributed Video Coding .................................................................................................. 35
2.2.3.3 High Order Motion Interpolation for Side Information Improvement in DVC

Chapter 3  Unidirectional Warping Side Information Creation: Architecture and Tools

3.1 UWSI Creation Framework
3.2 Geometric Transforms
3.3 UWSI Creation Tools
  3.3.1 Backward Motion Estimation
  3.3.2 Bidirectional Translational Motion Estimation
  3.3.3 Reference Frames Up-sampling
  3.3.4 Bidirectional Geometric Motion Estimation
  3.3.5 Block Size Adaptation
  3.3.6 Motion Model Decision
  3.3.7 Motion Compensation
3.4 Summary

Chapter 4  Bidirectional Warping Side Information Creation: Architecture and Tools

4.1 BWSI Creation Framework
4.2 BWSI Creation Tools
  4.2.1 Backward Geometric ME
  4.2.2 GT Vectors Fusion
  4.2.3 Motion Compensation
4.3 Summary

Chapter 5  Warping Side Information Creation Methods: Performance Evaluation

5.1 Test Conditions
  5.1.1 Video Sequences
  5.1.2 Coding Conditions
  5.1.3 Coding Benchmarks
  5.1.4 Performance Evaluation Metrics
5.2 Side information Performance Evaluation
5.3 RD Performance Evaluation
5.4 Conclusions

Chapter 6  Final Remarks
6.1 Summary ................................................................................................................................. 83
6.2 Achievements .......................................................................................................................... 84
6.3 Future work ............................................................................................................................. 84
Index of Figures

Figure 1.1 – Example of the numerous devices and applications capable of processing video content. .......................................................................................................................... 1

Figure 1.2 – (a) Down-link versus (b) up-link application scenarios. ................................................................. 2

Figure 2.1 – Some geometric transforms examples. ................................................................................................ 6

Figure 2.2 – Homogeneous vector \([u, v, h]\) and its projection on the plane \(h = 1\). .......................... 7

Figure 2.3 – Perspective transform of points \(A, B, C\) and \(D\) onto \(A', B', C'\) and \(D'\) [9]. ..................... 8

Figure 2.4 – 8 x 8 DCT basis functions used in image and video coding standards [10]. ................. 12

Figure 2.5 – Basic predictive video coding architecture [6]. ................................................................. 12

Figure 2.6 – Typical architecture of a H.264/AVC predictive video codec [1]. ..................................... 14

Figure 2.7 – Basic architecture of the proposed picture-level parametric motion representation algorithm ................................................................................................................................ 15

Figure 2.8 – Example of six correspondence points to be grouped later [11]. ........................................... 16

Figure 2.9 – An example of two warped reference pictures (\(W1\) and \(W2\)), where \(W1\) is based on the background motion and \(W2\) is based on the foreground motion (cards) [11]. ........................................ 16

Figure 2.10 – Basic architecture of the proposed parametric motion estimation [12]. ....................... 20

Figure 2.11 – Parametric SKIP incorporated in a common hybrid video encoding [12]. ...................... 22

Figure 2.12 – Coding structure of the test conditions used: (a) low delay and (b) random access (DO regards the display order and CO coding order) [12]. .................................................. 23

Figure 2.13 – Example of the PSKIP mode usage: (a) without PSKIP mode; (b) with PSKIP mode available (orange = SKIP, green = INTER, blue = INTRA and magenta = PSKIP) [12]. ................. 24

Figure 2.14 – Slepian-Wolf coding architecture for two correlated signals, \(X\) and \(Y\), i.d.d. [17]...... 25

Figure 2.15 – Wyner-Ziv coding architecture for two correlated signals, \(X\) and \(Y\), i.d.d. [17] ...... 26

Figure 2.16 – DISCOVER Wyner-Ziv video codec architecture [5]. ......................................................... 26

Figure 2.17 – RD performance for the adopted test sequences [5]. .......................................................... 29

Figure 2.18 – Encoding complexity measured in terms of encoding time for the Coastguard video sequence [5]. .................................................................................................................... 29

Figure 2.19 – Side information creation framework [18]. .......................................................................... 30

Figure 2.20 – Adaptive Search Range calculation [18].................................................................................. 31

Figure 2.21 – Bidirectional motion estimation with linear trajectory (one dimension) [18] ................. 32

Figure 2.22 – RD results regarding the tested sequences [18]. .............................................................. 34

Figure 2.23 – Basic architecture of the proposed hybrid solution. .......................................................... 35
Figure 2.24 – Mesh deformation: (a) starting mesh for the past reference frame and (b) deformed mesh for the future reference frame [20].

Figure 2.25 – Frame interpolated with: (a) block based approach, (b) mesh-based approach and (c) proposed hybrid solution, (d) signals the use of the block-based approach (light pixels) versus the mesh-based approach (dark pixels) for the proposed hybrid SI creation solution [20].

Figure 2.26 – RD performance for the Foreman sequence [20].

Figure 2.27 – Basic architecture of the proposed high order interpolation method.

Figure 2.28 – Bidirectional motion estimation as performed in the DISCOVER WZ codec [23].

Figure 2.29 – Proposed high order interpolation method for motion estimation [23].

Figure 3.1 – Architecture of the proposed UWSI creation solution.

Figure 3.2 – Quadrilateral to square mapping [6].

Figure 3.3 – Half pixel and quarter pixel positions definition. Capital letters represent the integer pixels and lower case letters represent the half and quarter pixel positions [1].

Figure 3.4 - (a) Full pixel and (b) Half and quarter pixel samples illustration.

Figure 3.5 - Architecture of the proposed Bidirectional Geometric ME.

Figure 3.6 - (a) Initial GT vectors positions and (b) Possible best GT vector positions after performing the search for the first vertex (considering only the backward direction).

Figure 3.7 – Linear trajectory for the GT vector of a given SI block vertex.

Figure 3.8 – (a) Input translational motion vectors and (b) translational GT vectors.

Figure 3.9 – Blocks warping scheme.

Figure 3.10 – Bilinear pixel interpolation for a regular square grid [6].

Figure 3.11 – Example of the block size adaptation for one direction: (a) GT vectors positions for the larger block and (b) obtained GT vectors positions for the four smaller blocks.

Figure 4.1 - Architecture of the proposed BWSI creation solution.

Figure 4.2 – Backward geometric motion estimation model.

Figure 4.3 – Graphical representation of distance di.

Figure 4.4 – Graphical representation of the two transforms $T_{si,b}$ and $T_{si,f}$ obtained for an SI block.

Figure 4.5 – An example of a typical occlusion [25].

Figure 4.6 – Typical scenario for (a) backward unidirectional and (b) forward unidirectional compensation modes.

Figure 4.7 – Backward unidirectional mode compensation example.

Figure 5.1 – Video sequences selected: (a) first frame of Container; (b)frame 63 of Hall Monitor; (c) first frame of Foreman; (d) first frame of Mobile and Calendar; (e) frame 50 of Coastguard; (f) frame 99 of Table Tennis; (g) frame 29 of Bus; (h) first frame of Stefan; (i) first frame of Soccer.

Figure 5.2 - $4 \times 4$ DCT quantization matrices corresponding to the eight RD points [16].

Figure 5.3 – Temporal evolution of the SI PSNR for the Container sequence.
Figure 5.4 - Temporal evolution of the SI PSNR for the Mobile and Calendar sequence. ............ 76
Figure 5.5 - Temporal evolution of the SI PSNR for the Bus sequence. .................................. 76
Figure 5.6 – Evolution of the camera motion in the Mobile and Calendar sequence. ............... 77
Figure 5.7 – DVC-UWSI and DVC-BWSI codecs RD performance for the Container sequence... 78
Figure 5.8 – DVC-UWSI and DVC-BWSI codecs RD performance for the Hall Monitor sequence.
........................................................................................................................................ 78
Figure 5.9 – DVC-UWSI and DVC-BWSI codecs RD performance for the Foreman sequence.... 78
Figure 5.10 – DVC-UWSI and DVC-BWSI codecs RD performance for the Mobile and Calendar
sequence. ................................................................................................................................. 79
Figure 5.11 – DVC-UWSI and DVC-BWSI codecs RD performance for the Coastguard sequence.
........................................................................................................................................ 79
Figure 5.12 – DVC-UWSI and DVC-BWSI codecs RD performance for the Table Tennis sequence.
........................................................................................................................................ 79
Figure 5.13 – DVC-UWSI and DVC-BWSI codecs RD performance for the Bus sequence. ....... 80
Figure 5.14 – DVC-UWSI and DVC-BWSI codecs RD performance for the Stefan sequence..... 80
Figure 5.15 – DVC-UWSI and DVC-BWSI codecs RD performance for the Soccer sequence.... 80
Figure 6.1 – (a) Continuous method and (b) discontinuous method [6]. ................................. 85
Index of Tables

Table 2.1 – Spatial resolution of the additional test sequences [11]. ................................................. 18
Table 2.2 - BD rate metric for low delay and random access conditions [11]................................. 19
Table 2.3 - Sequences and test results in terms of BD-rate and BD-PSNR [12]. ......................... 23
Table 2.4 – Test conditions [18]. ........................................................................................................ 33
Table 2.5 – PSNR SI quality for each module of the SI creation framework (RD point Qd) [18]. ... 33
Table 2.6 – ΔPSNR [dB] for full, half and quarter pixel precisions regarding DISCOVER WZ codec [23]. ......................................................................................................................... 40
Table 2.7 – Optimal value for λ for different GOP sizes [23]. ............................................................. 41
Table 2.8 - ΔPSNR [dB] for different GOP sizes regarding DISCOVER [23]............................ 41
Table 2.9 – RD Bjontegaard metrics comparing the proposed and the DISCOVER SI creation processes [23]. .................................................................................................................. 42
Table 5.1 – Test video sequences characteristics. .............................................................................. 69
Table 5.2 - Key frames quantization parameters for each RD point, Qi............................................ 70
Table 5.3 – SI PSNR quality for each video sequence tested. ............................................................. 74
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASR</td>
<td>Adaptive Search Range</td>
</tr>
<tr>
<td>AVC</td>
<td>Advanced Video Coding</td>
</tr>
<tr>
<td>BD</td>
<td>Bjøntegaard Distortion</td>
</tr>
<tr>
<td>BMA</td>
<td>Block Matching Algorithm</td>
</tr>
<tr>
<td>BWSI</td>
<td>Bidirectional Warping Side Information</td>
</tr>
<tr>
<td>CABAC</td>
<td>Context-Adaptive Binary Arithmetic Coding</td>
</tr>
<tr>
<td>CAVLC</td>
<td>Context-Adaptive Variable-Length Coding</td>
</tr>
<tr>
<td>CIF</td>
<td>Common Intermediate Format</td>
</tr>
<tr>
<td>CNM</td>
<td>Correlation Noise Model</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>DCT</td>
<td>Discrete Cosine Transform</td>
</tr>
<tr>
<td>DISCOVER</td>
<td>DIStributed COding for Video sERvices</td>
</tr>
<tr>
<td>DPCM</td>
<td>Differential Pulse-code Modulation</td>
</tr>
<tr>
<td>DSC</td>
<td>Distributed Source Coding</td>
</tr>
<tr>
<td>DVC</td>
<td>Distributed Video Coding</td>
</tr>
<tr>
<td>FMO</td>
<td>Flexible Macroblock Order</td>
</tr>
<tr>
<td>GOP</td>
<td>Group of Pictures</td>
</tr>
<tr>
<td>GT</td>
<td>Geometric Transforms</td>
</tr>
<tr>
<td>HEVC</td>
<td>High Efficiency Video Coding</td>
</tr>
<tr>
<td>HVS</td>
<td>Human Visual System</td>
</tr>
<tr>
<td>IDCT</td>
<td>Inverse Discrete Cosine Transform</td>
</tr>
<tr>
<td>IST</td>
<td>Instituto Superior Técnico</td>
</tr>
<tr>
<td>JCT-VC</td>
<td>Joint Collaborative Team – Video Coding</td>
</tr>
<tr>
<td>KLT</td>
<td>Kanade-Lucas-Tomasi</td>
</tr>
<tr>
<td>LDPC</td>
<td>Low-Density Parity-Check</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>LDPCA</td>
<td>Low-Density Parity-Check Accumulate</td>
</tr>
<tr>
<td>ITU-T</td>
<td>International Telecommunication Union – Telecommunication standardization sector</td>
</tr>
<tr>
<td>MAD</td>
<td>Mean Absolute Difference</td>
</tr>
<tr>
<td>MB</td>
<td>MacroBlock</td>
</tr>
<tr>
<td>MC</td>
<td>Motion Compensation</td>
</tr>
<tr>
<td>MCFI</td>
<td>Motion Compensated Frame Interpolation</td>
</tr>
<tr>
<td>ME</td>
<td>Motion Estimation</td>
</tr>
<tr>
<td>MPEG</td>
<td>Moving Picture Experts Group</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Square Error</td>
</tr>
<tr>
<td>OSA</td>
<td>Orthogonal Search Algorithm</td>
</tr>
<tr>
<td>PCM</td>
<td>Pulse-Code Modulation</td>
</tr>
<tr>
<td>PSKIP</td>
<td>Parametric SKIP</td>
</tr>
<tr>
<td>PSNR</td>
<td>Peak Signal-to-Noise Ratio</td>
</tr>
<tr>
<td>QCIF</td>
<td>Quarter Common Intermediate Format</td>
</tr>
<tr>
<td>QP</td>
<td>Quantization Parameter</td>
</tr>
<tr>
<td>RD</td>
<td>Rate-Distortion</td>
</tr>
<tr>
<td>SAD</td>
<td>Sum of Absolute Differences</td>
</tr>
<tr>
<td>SE</td>
<td>Side information Estimator</td>
</tr>
<tr>
<td>SMF</td>
<td>Spatial Motion Filtering</td>
</tr>
<tr>
<td>SSD</td>
<td>Sum of Squared Differences</td>
</tr>
<tr>
<td>SW</td>
<td>Slepian-Wolf</td>
</tr>
<tr>
<td>SI</td>
<td>Side Information</td>
</tr>
<tr>
<td>TMuC</td>
<td>Test Model under Consideration</td>
</tr>
<tr>
<td>TSS</td>
<td>Three Step Search</td>
</tr>
<tr>
<td>UWSI</td>
<td>Unidirectional Warping Side Information</td>
</tr>
<tr>
<td>WZ</td>
<td>Wyner-Ziv</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The main target of this chapter is to provide context to this Thesis, explain the motivation and problems behind this research work and, finally, the objectives to be addressed. Additionally, the organization of the Thesis is also reported.

1.1 Context and Motivation

Nowadays, image, video and audio coding technologies are widely used everywhere and everyday by a significant amount of the world population (see Figure 1.1). Operations like uploading and downloading video streams, exchanging emails and making voice calls over the Internet are more and more common. Sometimes without noticing, people all over the world exchange experiences and information based on telecommunications services; all together, this leads to a huge volume of data being transmitted and stored, especially when video data is involved.

![Figure 1.1 – Example of the numerous devices and applications capable of processing video content.](image)

The key objective of digital audiovisual coding techniques is to compress the original audiovisual information into the minimum number of bits for a target decoded signal quality, eventually also fulfilling other relevant requirements such as random access and scalability. Nowadays, the video coding paradigm adopted by the largely deployed MPEG and ITU-T standards relies mostly on the following tools: motion compensated temporal prediction between video frames, to exploit the temporal correlation; a discrete cosine transform (DCT) coding, to exploit the spatial redundancy;
quantization of the DCT coefficients, to exploit the perceptual redundancy; and entropy coding, to exploit the statistical redundancy of the created symbols.

The state-of-the-art on predictive video coding is represented by the H.264/AVC (Advanced Video Coding) video coding standard [1], which is largely adopted by most video enabled services and devices. Comparing to the previous coding standards, the H.264/AVC standard provides a compression efficiency gain up to 50%, this means the same quality for half the bitrate. To achieve this compression efficiency, many complex techniques must be employed, notably at the encoder where the most important decisions are taken, which means that the encoder complexity is rather high compared to be decoder. In fact, the H.264/AVC encoder complexity can be 10-100 times larger than the decoder complexity [2]; this type of complexity budget suits well a down-link broadcast model, where few encoders provide coded content to many simpler and cheaper decoders.

However, some emerging video applications are not well characterized by the down-link model but rather follow an up-link model, which means that some simple devices deliver information to a central, eventually rather complex, receiver (see Figure 1.2).

![Figure 1.2](image)

Figure 1.2 – (a) Down-link versus (b) up-link application scenarios.

These novel requirements and needs led to the emergence of a new video coding paradigm based on the Slepian-Wolf and the Wyner-Ziv (WZ) information theory theorems [3] [4], well known as Distributed Video Coding (DVC). Among other goals, DVC targets to use light encoding systems while achieving the same compression efficiency as the best predictive video coding schemes available. The DVC paradigm also provides improved error resilience, codec independent scalability and a good exploitation of the multi-view correlation without the encoders communicating among them [5]. These characteristics address important needs from emerging applications such as wireless low-powered video surveillance, visual sensor networks and mobile video communications.

Considering the DVC paradigm, it is crucial that the correlation between video signals can be properly exploited at the decoder side, in order to achieve satisfying Rate-Distortion (RD) performance results. Therefore, the module responsible for this task, typically the side information (SI) creation module, is rather important, and its performance strongly affects the overall performance of the DVC codec. Typically, to create side information a block level translational motion model is used to estimate the SI frame. However, this motion representation is not powerful and accurate enough to efficiently estimate complex motions, like zooms, rotations and shears. So, in order to overcome the limitations of the translational motion model, the use of a geometric motion model to estimate the side
information frame is explored on this Thesis. The proposed side information creation method with geometric transforms (GT) can improve the side information quality when complex motion occurs, especially for the cases where the translational motion model cannot perform well.

1.2 Objectives

Since the techniques to generate the side information at the decoder significantly influence the RD performance on the Wyner-Ziv (WZ) video coding solutions, the main objectives of this Thesis are to design, implement and evaluate DVC solutions capable of better estimate the side information at the decoder side using geometric transforms. With this purpose in mind, the following tasks are defined:

- **Detailed review of geometric transforms** – Initially, the theoretical foundations and some geometric transform based video coding solutions available in the literature are reviewed in order to understand which related state-of-the-art solutions are available.
- **Detailed review of side information creation techniques** – Then, state-of-the-art side information creation techniques available in the literature are reviewed in detail, since this regards the main topic of the Thesis.
- **Side information creation design and implementation** – After, side information creation solutions using geometric transforms are designed and implemented. During this process, specific and local assessment evaluations are performed to help solving possible issues and achieve better results in terms of side information quality.
- **DVC codec performance evaluation** – Finally, the proposed side information creation solutions are integrated in a DVC codec and the RD performance are evaluated to conclude the benefits such approach could bring in the distributed video coding scenario.

In summary, the proposed side information creation solutions are expected to improve side information quality and, after integrated in a DVC codec, should provide a better RD performance.

1.3 Thesis Structure

This Thesis is organized in six chapters, including this first chapter that features an introduction and the context of this Thesis.

Chapter 2 provides a detailed review of the geometric transforms, including some predictive video coding solutions exploiting geometric transforms already available in the literature. Then, the distributed video coding paradigm is studied, as well as some relevant side information creation methods already available.

In Chapter 3, the first side information creation method using geometric transforms, called Unidirectional Warping Side Information (UWSI), is proposed. Initially, the UWSI architecture is presented, followed by a detailed description of the most relevant techniques used, notably bidirectional geometric motion estimation and the motion model decision.

Next, Chapter 4 proposes a second side information creation method using geometric transforms, called Bidirectional Warping Side Information (BWSI). After presenting the BWSI architecture, the novel techniques, comparing to the UWSI, are described in detail, notably the initial motion estimation in both directions and the GT vectors fusion.
In Chapter 5, both side information creation solutions, as well as the respective DVC codecs, are evaluated. After defining the test conditions, the solutions are evaluated in terms of SI quality and RD performance.

Finally, in Chapter 6, the conclusions, achievements and some possible future developments are presented.
Chapter 2

Geometric Transforms and Video Coding: Reviewing the Relevant Background

The main objective of this chapter is to review the relevant technical background for this Thesis, notably predictive video coding based on geometric transforms and distributed video coding. With this purpose in mind, the next subsections will review some of the most relevant concepts and available technical solutions for the objectives of this Thesis.

2.1 Reviewing Geometric Transforms based Predictive Video Coding

This subsection intends to review predictive video coding based on geometric transforms. To achieve this target, this subsection will first review the basics on geometric transforms and predictive video coding and after the most relevant solutions in the literature combining predictive video coding with geometric transforms.

2.1.1 Basics on Geometric Transforms

A geometric transform may be defined as a one-to-one mapping of a straight line, plane or space onto itself. In fact, the basis of geometric transforms is the mapping of one coordinate system onto another. With that purpose, a mapping function establishing a connection between an input plane, e.g. a reference image, and an output plane, e.g. the warped image, is used. Considering the purpose of this Thesis, geometric transforms for images will be mainly considered in the following (see some examples in Figure 2.1).
An alternative to the traditional block matching algorithm (BMA) method for motion estimation is the use of geometric transforms to better estimate motion compensated (MC) predictions [6]. With some additional computational complexity, the combination of a block matching algorithm with a spatial transform might lead to more efficient estimations and, thus, to a reduction of the final bitrate.

Regarding geometric transforms, there are some simpler mapping functions that can be used, such as affine, perspective, bilinear and polynomial, and also more complex geometric transforms due to the high number of parameters involved that will not be considered in this text [6]. As the polynomial transform also requires a rather high number of parameters, only the affine, perspective and bilinear transforms will be considered in the following as they are those typically used for the purposes of this Thesis [7].

Consider that each point in the input image has known coordinates. The output image is the observed warped image where the forward mapping function can be written as:

$$ [x, y] = [X(u, v), Y(u, v)] $$

where \([u, v]\) indicates the coordinates of the input image corresponding to the coordinates \([x, y]\) in the output image. For the inverse mapping function, it comes:

$$ [u, v] = [U(x, y), V(x, y)] $$

This means that, for the forward mapping, each pixel in the input image is copied to the output image through the \(X\) and \(Y\) mapping functions, while the same method is used for the inverse mapping (in this case, using the \(U\) and \(V\) functions). Due to the fact that the output grid is also discrete and the mapping functions might return any real coordinate, interpolation techniques may have to be used to avoid the phenomena of holes and overlapping of pixels in the output image.

Homogeneous coordinates is a coordinates system used in projective geometry; they have a wide range of applications in computer graphics as it is possible to represent geometric transforms by a simple matrix. With the use of homogeneous coordinates, it is possible to easily represent any \([x, y]\) Euclidean coordinates by simply adding one more coordinate; so, the 2D-points are represented by 3 coordinates vectors. In projective geometry, \([x, y]\) is represented by the homogeneous vector \([u, v, h]\), where \([u, v, h]\) represents a point at the infinity (as shown in Figure 2.2).
Figure 2.2 – Homogeneous vector \([u, v, h]\) and its projection on the plane \(h = 1\).

Note that \(h\) could be any non-zero number. Since \(u = xh\), \(v = yh\) and \([u, v, h] = [xh, yh, h]\), it is possible to conclude that the recovery of the original Euclidean coordinates is pretty simple. That is, with a simple division by the homogeneous component \(h\), it is possible to get \([x, y]\), since \(x = u/h\) and \(y = v/h\).

As only projections on the 2D-plane are interesting for the purposes of this Thesis, it is possible to ignore the homogeneous component by considering \(h = 1\). In this way, the focus is mainly on the mapping between \([x, y]\) and \([u, v]\). To do so, a \(3 \times 3\) matrix is used to specify the 2D coordinate transforms, in this case matrix \(T1\) in equation (2.3). This matrix works on the previous homogeneous coordinates system and provides a good representation for the affine and the perspective transforms. So, it comes

\[
[xw, yw, w] = [u, v, 1] \ast T1
\]  

(2.3)

where

\[
T1 = \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]  

(2.4)

To better explain the function of these variables, matrix \(T1\) is decomposed, starting with the following sub-matrix:

\[
T2 = \begin{bmatrix}
a_{11} & a_{12} \\
a_{21} & a_{22}
\end{bmatrix}
\]  

(2.5)

Matrix \(T2\) specifies a linear transform for scaling, reflection, shearing and rotation. Moreover, the vector \(\begin{bmatrix} a_{13} \\ a_{23} \end{bmatrix}\) generates the perspective transform, while the vector \(\begin{bmatrix} a_{11} & a_{12} \end{bmatrix}\) is responsible for the translation of the \(x\) and \(y\) coordinates, respectively, and the component \(a_{33}\) is a scaling factor for the plane \(h\).

### 2.1.1.1 Affine Transform

The affine mapping transform include zooms, rotations, translations and shears\(^1\) \([6] [7]\). The main characteristic of the affine transform is that it preserves the parallel lines. For the affine transform, \(a_{13} = 0, a_{23} = 0\) and \(a_{33} = 1\).

\[
[xw, yw, w] = [u, v, 1] \begin{bmatrix}
a_{11} & a_{12} & 0 \\
a_{21} & a_{22} & 0 \\
a_{31} & a_{32} & 1
\end{bmatrix}
\]  

(2.6)

---

\(^1\) A shear mapping leaves all points on one axis fixed, while the other points are shifted parallel to the axis by a distance proportional to their perpendicular distance from that axis.
This means that matrix $T_1$ has six degrees of freedom $(a_{11}, a_{12}, a_{22}, a_{23}, a_{31}, a_{32})$. To find these 6 transform coefficients, the coordinates correspondence of three non-collinear points in both images $([x_k, y_k])$ and $([u_k, v_k])$, where $k \in [0,2]$, are needed. Note that here $w = 1$ due to the fact that $h = 1$. Solving the six linear equations in (2.7), it is possible to find the six coefficients defining the affine transform.

$$
\begin{bmatrix}
x_0 & y_0 & 1 \\
x_1 & y_1 & 1 \\
x_2 & y_2 & 1
\end{bmatrix}
= 
\begin{bmatrix}
u_0 & v_0 & 1 \\
u_1 & v_1 & 1 \\
u_2 & v_2 & 1
\end{bmatrix}
\begin{bmatrix}
a_{11} & a_{12} & 0 \\
a_{21} & a_{22} & 0 \\
a_{31} & a_{32} & 1
\end{bmatrix}
$$

(2.7)

The affine transform can easily map any input triangle into any output triangle, or any input rectangle into any output parallelogram. However, when it comes to warp a rectangle into any quadrilateral, the affine transform will not work; this happens because the values $a_{13}$ and $a_{23}$ were considered to be zero. Since these two coefficients are associated to the perspective transform (defined below), naturally the affine transform cannot map any quadrilateral. Due to this fact, a more efficient transform, which means more complex too, may be needed.

### 2.1.1.2 Perspective Transform

The perspective transform is also known as the projective mapping or homography transform. Formally, a perspective transform in a plane is a transformation used in projective geometry: it is the composition of a pair of perspective projections [6] [7]. It describes what happens to the perceived positions of observed objects when the point of view of the observer changes. Perspective transforms do not preserve sizes or angles but do preserve incidence and cross-ratio, two properties which are important in projective geometry. Incidence is the property that guarantees that all points in a line on the input plane lie on the correspondence line in the output image, i.e. it ensures that the perspective transform preserves all the lines. The cross-ratio property states that any four collinear points have a given cross-ratio value defined by equation (2.8), where $A, B, C$ and $D$ are the chosen points [8].

$$
(A, B; C, D) = \frac{AC}{BD}
$$

(2.8)

In perspective transforms, the cross-ratio of collinear points does not change under any projective line (as shown in Figure 2.3). This means that $(A, B, C, D)$ is equal to $(A', B', C', D')$.

![Figure 2.3 – Perspective transform of points $A, B, C$ and $D$ onto $A', B', C'$ and $D'$ [9].](image)

The perspective transform differs from the affine transform in the two coefficients that are responsible for the perspective transform; this also means that the perspective transform is able to do everything that the affine transform can do. The perspective transform has the following general representation

\[...\]
Since \( w = ua_{13} + va_{23} + a_{33} \) and \( xw = ua_{11} + va_{21} + a_{31} \), it comes

\[
x = \frac{ua_{11} + va_{21} + a_{31}}{ua_{13} + va_{23} + a_{33}}
\]

\[
y = \frac{ua_{12} + va_{22} + a_{32}}{ua_{13} + va_{23} + a_{33}}
\]

Considering \( a_{33} = 1 \), as \( a_{33} \) is a scaling factor for the plane \( h \) and \( h = 1 \), these equations become

\[
x = ua_{11} + va_{21} + a_{31} - xu_{13} - xv_{a_{23}}
\]

\[
y = ua_{12} + va_{22} + a_{32} - yu_{13} - yv_{a_{23}}
\]

The perspective transform has eight degrees of freedom. To find these coefficients, the coordinates correspondence of four non-collinear points in both images \((x_k, y_k)\) and \([u_k, v_k]\), where \(k \in [0,3]\), are needed. With this, it is possible to find the eight coefficients by solving the following system of equations:

\[
\begin{bmatrix}
u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0x_0 & -v_0x_0 & a_{11} & x_0 \\
u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 & a_{21} & x_1 \\
u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 & a_{31} & x_2 \\
u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3x_3 & -v_3x_3 & a_{41} & x_3 \\
0 & 0 & 0 & 0 & u_0 & v_0 & 1 & 0 & -u_0y_0 & -v_0y_0 & a_{22} & y_0 \\
0 & 0 & 0 & 0 & u_1 & v_1 & 1 & 0 & -u_1y_1 & -v_1y_1 & a_{32} & y_1 \\
0 & 0 & 0 & 0 & u_2 & v_2 & 1 & 0 & -u_2y_2 & -v_2y_2 & a_{43} & y_2 \\
0 & 0 & 0 & 0 & u_3 & v_3 & 1 & 0 & -u_3y_3 & -v_3y_3 & a_{43} & y_3
\end{bmatrix}
\]

Although the perspective transform is more complex than the affine transform, it has the advantage of mapping any input quadrilateral into any output quadrilateral; this is a very important characteristic. Typically, the map used is a quadrilateral to square mapping and the four non-collinear points are the vertices of each quadrilateral.

### 2.1.1.3 Bilinear Transform

The bilinear transform is used to get even more complex mappings, like quadrilaterals onto non-planar quadrilaterals [6][7]. This mapping technique is based on linear interpolation between two edges of the quadrilateral. In this way, this transform preserves lines which are horizontal or vertical in the output quadrilateral and preserves equispaced points along such lines. However, it does not preserve equispaced points along diagonal lines. The bilinear transform has the following representation

\[
[x, y] = [uv, u, v, 1] \begin{bmatrix} a_3 & b_3 \\
a_1 & b_1 \\
a_2 & b_2 \\
a_0 & b_0 \end{bmatrix}
\]

Then, it is possible to conclude that

\[
x = a_0 + a_1u + a_2v + a_3uv
\]

\[
y = b_0 + b_1u + b_2v + b_3uv
\]

Note that, if \( a_3 = 0 \) and \( b_3 = 0 \), the bilinear mapping becomes the affine mapping.
To find the eight coefficients, the coordinates correspondence of four non-collinear points in both images are needed, as for the perspective transform. For this, it is possible to solve the following system of equations and get the eight coefficients:

\[
\begin{bmatrix}
    x_0 & y_0 \\
    x_1 & y_1 \\
    x_2 & y_2 \\
    x_3 & y_3 \\
\end{bmatrix}
= 
\begin{bmatrix}
    1 & u_0 & v_0 & u_0v_0 \\
    1 & u_1 & v_1 & u_1v_1 \\
    1 & u_2 & v_2 & u_2v_2 \\
    1 & u_3 & v_3 & u_3v_3 \\
\end{bmatrix}
\begin{bmatrix}
    a_0 & b_0 \\
    a_1 & b_1 \\
    a_2 & b_2 \\
    a_3 & b_3 \\
\end{bmatrix}
\]

(2.15)

2.1.1.4 Geometric Transforms Comparison

The affine mapping is the simplest of the three transforms presented above as it is defined by the smallest number of coefficients. Notwithstanding their simplicity, perspective and bilinear mappings are very useful as they can map any quadrilateral (while the affine transform can only map parallelograms). Between the perspective and bilinear mappings, the first is preferable because it preserves all the lines, while the later only preserves the horizontal and vertical lines in the input. With respect to computational complexity, bilinear is better because it requires fewer operations when compared to the perspective mapping [7].

2.1.2 Predictive Video Coding: Basics and State-of-the-Art

Considering its relevance for this Thesis, this section intends to review the basics and the state-of-the-art on predictive video coding. After a few standards developed in the past twenty years, the H.264/AVC standard provides nowadays the best video compression performance and is largely used in real applications.

2.1.2.1 Basics

Predictive video coding is a powerful approach to remove redundancy and irrelevancy in moving pictures, this means video; its main objective is to achieve high compression ratios when compared to the Pulse-Coded Modulation (PCM) solution for a specific target quality. The simplest way to represent an image is with the so-called PCM format. Typically, a PCM colour image requires $M \times N$ samples of the luminance signal and the corresponding two sets of chrominances samples which resolution depends on the subsampling format; moreover, video data is composed by a sequence of pictures of similar spatial resolution and a certain temporal rate. Considering $P$ bits per sample and a video with 25 images per second, the total number of bits per second in the PCM format is $25 \times M \times N \times P \times 3$ if the three components have the same resolution; this number may be huge. Thus, the need of reduce this bitrate is crucial, if practical conditions are addressed as the transmission and storage capacities are always limited. This reduction may be done by eliminating the spatial, temporal and statistical (all mathematical) redundancies, and the irrelevancy, also called perceptual redundancy, as they do not contribute to increase the mathematical and perceptual qualities.

Predictive video coding solutions, also called hybrid video coding solutions, exploit the temporal redundancy by means of temporal predictions and the spatial redundancy using a transform which may have a frequency interpretation (leading to the hybrid label associated to the time+frequency combination). Moreover, also the statistical redundancy is exploited by means of entropy coding and the irrelevancy by quantizing the transform coefficients. This type of video coding solution has been
adopted by the very popular and largely adopted ITU-T H.26x and MPEG-x families of standards which sponsored the explosion of digital video applications and services like digital TV, mobile and Internet video streaming, DVD, Blu-ray, YouTube, etc. The state-of-the-art on predictive video coding is nowadays represented by the H.264/AVC (Advanced Video Coding) standard which provides a compression gain of about 50% regarding the previously available video coding standards, notably the MPEG-1, MPEG-2 and MPEG-4 Visual standards.

In the following, the basics and most important tools of predictive video coding will be briefly described [1].

- **Exploiting the temporal redundancy** – In a video scene, nearby images are typically similar. Due to this fact, it is possible to exploit the temporal redundancy between adjacent frames that are highly correlated. To do this, temporal prediction typically combined with motion estimation and compensation are the main tools. These tools predict the current frame based on some previously and/or future coded frames. For this, three types of frames are often used:
  - **Intra (I)** – Intra frames are coded without exploiting the temporal redundancy and, thus, without temporal prediction. In live streaming, Intra frames allow the users to have a starting point, and thus random access, as well as to recover from potential transmission errors.
  - **Predicted (P)** – Predicted frames can use forward prediction from one previously coded Intra or Predicted frames.
  - **Bidirectional (B)** – Bidirectional predictive frames have the ability to use both forward and backward temporal prediction from two reference frames.

Using B frames causes an increase of complexity and delay, but they may also significantly increase the RD performance. On the other hand, the use of Intra frames reduces the complexity and stops the error propagation; however, the compression factor is much lower for the same quality. Naturally, the syntactic details of the three types of frames mentioned above differ for the various standards defined along the time.

Regarding motion compensation, translational motion estimation is typically performed. This operation is executed for each macroblock (MB) (corresponding to 16 x 16 luminance samples) or MB partition, which means that pixels within a block have the same motion. In order to find the best motion match, there are many methods and criteria to search for the best motion. While full search methods perform a full search in a pre-defined window around the block, thus leading to the optimal solution, partial search saves complexity at the cost of RD performance and there many fast motion search algorithms available, like Three Step Search (TSS) and Orthogonal Search Algorithm (OSA), which keep the motion search complexity lower.

- **Exploiting the spatial redundancy** – Spatial redundancy regards the fact that nearby pixels in a giving frame are often correlated with each other. To exploit this spatial redundancy, the Discrete Cosine Transform (DCT) is commonly used. This technique works on blocks and characterizes each block in the frequency domain using a limited number of transform coefficients. In this way, it is possible to achieve a strong energy compression, because typically lower frequencies have
higher energy and higher frequencies have lower energy. The $8 \times 8$ DCT basis functions are shown in Figure 2.4.

![DCT basis functions](image)

Figure 2.4 – $8 \times 8$ DCT basis functions used in image and video coding standards [10].

- **Exploiting the irrelevancy** – The main tool to exploit the irrelevancy is quantization. Quantization consists in dividing the DCT coefficients by a quantization parameter to reduce the precision of each coefficient. This quantization parameter may not be the same for each coefficient, i.e. each coefficient might have a quantization parameter that is more appropriate, notably due to the different sensitivity of the human visual system (HVS). Quantization removes the irrelevant information, i.e. information that the HVS cannot detect, to obtain a greater reduction of the final bitrate without any perceptual quality impact. Since this process eliminates information to which the HVS is not sensitive, quantization eliminates perceptually redundant information; although this is not a mathematically lossless process, it is possible to speak about a perceptually lossless process.

- **Exploiting the statistical redundancy** – To exploit the statistical redundancy, an entropy encoder is used to exploit the statistics of the coding symbols, notably quantized transform coefficients and motion vectors. A common example of an entropy coder is Huffman coding which assigns to the most used symbols shorter length codes and vice-versa. Huffman coding is mostly used for the motion vectors, DCT coefficients and MB coding classes. As it only exploits the statistical redundancy, entropy coding is a lossless process.

Combining all the tools mentioned above, Figure 2.5 presents a basic predictive video coding architecture.

![Predictive video coding architecture](image)

Figure 2.5 - Basic predictive video coding architecture [6].

The basic processing flow in predictive video coding proceeds as follows: First, the image is split in MBs with $16 \times 16$ pixels. Then, if the frame is not Intra coded, motion estimation is performed. Then,
the motion compensation module uses the motion vector information and previous decoded frame(s) to generate the prediction for the current frame, MB by MB. Finally, the prediction frame is subtracted from the original frame to obtain the MB level prediction error that will be transformed, quantized, entropy coded and transmitted to the receiver. To ensure that the receiver can use the predicted error correctly, the encoder must obtain and store the decoded frame, as the receiver does not have access to the original frame.

2.1.2.2 State-of-the-Art

The video coding standard representing nowadays the state-of-the-art is the H.264/AVC standard which is largely used in many applications domains, e.g. digital TV, mobile and Internet streaming, DVD and Blu-ray [1] [10]. Some of the main tools which allow the H.264/AVC standard achieving substantial gains regarding the previously available predictive video coding standards are described in the following.

- **Variable block-size prediction** – In H.264/AVC, each macroblock may be divided into four blocks of 8x8, and each block may be divided into four 4 x 4 blocks (considering the luminance component). The size of the coding blocks is chosen according to the degree of motion in the MB. Blocks like 4 x 8 and 16 x 8 may also be considered. This tool allows a very close adaptation of the used motion field to the video content characteristics and thus more efficient temporal predictions.

- **Quarter-pixel accuracy** – The H.264/AVC standard allows quarter-pixel accuracy for the motion vectors. This means that the motion search is performed with a higher accuracy (in this case, fractional precision motion vectors) considering interpolated sub-samples.

- **Multi-reference frames** – Unlike MPEG-2, where it is only possible to use a maximum of 2 reference frames, in H.264/AVC it is possible to use multiple reference frames. In other words, a list of reference pictures (up to 16) is created in the encoder, so they can be searched for better predictions at the cost of additional memory and computational power. With this purpose, the bitstream has to include some memory control commands, so that the decoder can store in its memory the same frames that will be needed for future predictions.

- **Generalized B frames and Weighted prediction** – In H.264/AVC, generalized B frames were introduced where B frames may be chosen as reference frames for predicting other frames. It is up to the encoder to decide which frames should be used as reference in a very flexible way. B-slice MBs may also be coded with a weighted prediction between two reference frames.

- **Intra-frame coding** – In the H.264/AVC standard, it is possible to use more efficient intra coding, i.e. it is possible to code the current macroblock based on adjacent blocks in the same frame, to better exploit the spatial redundancy. The main impact of this change happens for Intra frames where temporal predictions are forbidden for random access purposes.

- **Flexible slice-size and Flexible Macroblock Ordering** – Flexible slice-size and Flexible Macroblock Ordering (FMO) are two versatile tools providing a lot of flexibility to the encoder. For example, with flexible slice-size, it is possible to group the foreground of a picture using the slice
group concept. Flexible Macroblock Ordering can enhance the robustness to data losses by controlling the spatial relationship between the regions that are coded in each slice.

- **Hierarchical transform with Integer DCT** – Integer DCT is a new way to compute the transform coefficients only based on sums and shifts, thus eliminating the mismatch issues typical of the inverse DCT. In H.264/AVC, a hierarchical block transform is used with two layers: in the first layer, the integer DCT is applied to the sixteen $4 \times 4$ blocks of a MB; in the second layer, a Hadamard transform is applied to the $4 \times 4$ DC coefficients to exploit the correlation between those DC coefficients. This process is executed for both luminance and chrominances.

- **In-the-loop deblocking filtering** – The main objective of the in-the-loop deblocking filtering is to smooth the so-called block effect, thus improving the subjective and also objective compression performance. This technique is applied to all $4 \times 4$ block edges with an adaptive filtering strength depending on the video characteristics, thus improving the perceptual video quality without blurring the image.

- **Improved and adaptive entropy coding** – In H.264/AVC, the CAVLC (Context-Adaptive Variable-Length Coding) and CABAC (Context-Adaptive Binary Arithmetic Coding) entropy codecs were introduced. The use of context adaptive entropy coding proved to increase the H.264/AVC compression performance with rate reductions at no quality penalty.

The basic H.264/AVC coding architecture is presented in Figure 2.6.

![Figure 2.6 – Typical architecture of a H.264/AVC predictive video codec [1].](image)

The H.264/AVC profiles were designed to facilitate the interoperability between various applications of the standard that have similar functional requirements while limiting the complexity. At this stage, there are 7 profiles: Baseline, Main, Extended, High, High 10, High 4:2:2 and High 4:4:4. Each profile has the proper set of tools to provide certain functionalities, notably a certain compression capability, at an acceptable complexity. To complement the profiles, levels are defined which specify the parameters to be used, e.g. bitrate and spatial resolution. All decoders conforming to a specified profile must support all features in that profile. On the contrary, the bitstreams associated to a certain profile do not have to use all the tools in that profile. Encoders are not normative as this is not
requested for interoperability; however, naturally, encoders have to produce bitstreams compliant with a specific compliance point this means a profile@level combination.

2.1.3 Most Relevant Predictive Video Coding with Geometric Transforms Solutions

This subsection intends to review two solutions available in the literature combining predictive video coding with geometric transforms.

2.1.3.1 Picture-Level Parametric Motion Representation for Efficient Motion Compensation

The predictive coding solution with geometric transform described in this section has been proposed by Sung et al. in 2011 [11].

A. Objectives and Basic Approach

The objective of the proposed picture-level parametric motion representation solution is to be integrated in a predictive codec that improves the RD performance regarding an available reference codec. Besides the usual predictions, the improved codec searches for good warped reference pictures based on a geometric transform to include in the reference picture list. To achieve this purpose, a tracking algorithm is used to match correspondence points between a chosen reference picture and the current picture. Based on these correspondence points, a set of homography transforms are calculated. Each homography transform produces a warped reference picture that is compared to the current frame. The warped reference picture which proves to be most similar to the current frame is chosen. Then, an early decision rule is checked to determine if the chosen warped reference picture is added to the reference picture list used for the coding process.

B. Architecture and Walkthrough

The basic architecture of the proposed picture-level parametric motion representation process is presented in Figure 2.7. As explained, this process only eventually provides an additional reference picture for the proposed video codec.

![Figure 2.7 - Basic architecture of the proposed picture-level parametric motion representation algorithm.](image)

To better understand the proposed picture-level parametric motion representation process, its walkthrough is presented in the following:

- KLT algorithm – Initially, a tracking algorithm is used to match correspondence points between the current frame and a reference picture in the decoded picture buffer. In this case, the authors choose a KLT algorithm (Kanade-Lucas-Tomasi) that detects and tracks corner-like feature points, as shown in Figure 2.8.
After acquiring a set of correspondence points, groups of more than 4 points are created, based on a region growing approach.

- **Homography transform** – For each group of correspondence points, a warped reference picture is generated based on the homography transform calculated with those correspondence points. Thus, multiple warped reference pictures \( W_i \), with \( i = [1, 2, ..., M] \), are created and posteriorly compared to the current frame. Two warped reference pictures examples \( (W_1 \) and \( W_2 \)) are shown in Figure 2.9, along with the reference picture \( R \) and the current picture \( C \). Naturally, the selection of the type of homography to be used and the determination of its parameters are the key steps in this process which are detailed in the next subsection.

- **Selection of the best warped reference picture** – After obtaining all the warped reference pictures, a test algorithm is performed. This test is based on the Sum of Absolute Differences (SAD) which is calculated between each \( k \)-th block of the \( i \)-th warped reference picture and the same \( k \)-th block of the current picture. This test is very sensitive to motion, i.e. the KLT algorithm must deliver good correspondence points so that the homography transform can estimate the proper motion and the SAD is not too high.

Now, it is time to assess the warped frames by computing an expected gain for each block of each warped reference picture. Then, for each \( k \)-th block of each \( i \)-th warped reference picture \( (W_i) \), considering the reference picture \( R \) and the current frame \( C \), a SAD difference, \( DS_i^k \), is computed as:

\[
DS_i^k = SAD(R^k, C^k) - SAD(W_i^k, C^k)
\]  

(2.16)

The first SAD value is measured between the reference picture and the current picture and it is the same for all warped reference pictures based on a certain reference picture while the second SAD value is related to each warped frame. Then, a block-level gain expressing the SAD benefits of each warped block is computed as:

\[
G_i^k = \begin{cases} 
    DS_i^k, & \text{if } DS_i^k > 0 \\
    0, & \text{if } DS_i^k \leq 0 
\end{cases}
\]  

(2.17)
Considering that $DS^k_i$ values below zero lead to a worse motion compensation than the normal block matching algorithm, $DS^k_i$ values larger than zero are set to $G^k_i$ and $DS^k_i$ values below zero set to zero on $G^k_i$. To find the total SAD gain for each warped reference picture ($G_i$), the sum of all $G^k_i$ values is computed:

$$G_i = \sum_k G^k_i$$  \hspace{1cm} (2.18)

At this stage, the warped reference picture with the largest SAD gain is selected as the best one to be brought to the coding process.

After obtaining the best warped reference picture, the associated homography transform is applied to the 4 corners of the current picture to code to obtain their correspondence points and through them the 4 associated displacement vectors. The reason why this is done with the 4 picture corners is because it is computationally simpler to calculate the 8 parameters of the homography transform when using correspondence points that contain coordinates with the value zero and fewer bits are required to be transmitted.

- **Parameter Quantization** – Then, a quarter-pel quantization of the homography parameters is performed using the 4 displacement vectors. This is done because it has been proven to be better to transmit the 4 displacement vectors with quantization than the 8 parameters of the homography transform with quantization.

- **Decision Rule** – Finally, an early decision rule is applied to check if inserting the warped frame in the coding buffer has the potential to bring enough RD performance gains. The rule states that the best warped reference picture should only be added to the reference picture list if a large enough SAD gain is obtained. To do this, the blocks with the larger gains, i.e. those where $SAD(R^k, C^k)$ is two times larger than $SAD(W^k, C^k)$, are marked. Then, for the warped reference picture to be effectively considered in the coding process, a minimum of 15% of the blocks must be marked; otherwise, the warped frame is discarded. Finally, if the best warped reference picture fulfills this rule, it is placed in the reference picture list.

Considering the increase of processing time, another decision rule is tested in an attempt to reduce the overall complexity of this algorithm. This rule states that if a certain warped reference picture is not included in the reference picture list, it means that the parametric motion is not good enough; then, any frame between the given reference frame and the current frame is forced not to use this process.

C. **Main Novel Coding Tool**

The most relevant coding tool present in this picture-level parametric motion representation algorithm is the homography transform function. In this case, the perspective transform has been selected, i.e. a mapping function that can map a point $p(x, y)$ to a point $p'(x', y')$ using 8 parameters $(h_1, ..., h_8)$ to characterize the parametric motion of the picture as follows:

$$x' = \frac{h_1x + h_2y + h_3}{h_7x + h_8y + 1}$$  \hspace{1cm} (2.19)
The perspective transform parametric motion can be computed using a minimum of four correspondence points between the two involved frames. In this solution, the authors adopted a normalized direct linear transform algorithm, which requires more than 4 correspondence points, to calculate the 8 parameters [11].

D. Performance Evaluation

The proposed picture-level parametric motion representation algorithm has been implemented in the context of the HEVC Test Model under Consideration (TMuC) 0.7.3. The High Efficiency Video Coding (HEVC) standard is currently under joint development by MPEG and ITU-T and targets providing improved compression efficiency regarding the H.264/AVC standard. For the tests performed, the low delay and random access conditions, which are among those approved by the JCT-VC joint MPEG and ITU-T committee, have been used. Besides the defined sequences, some additional sequences, notably Flowervase, City, Jets, Bluesky_1080p and Station2_1080p, with more complex motion were tested to check the performance of this algorithm in those conditions; the spatial resolution is specified below in Table 2.1.

<table>
<thead>
<tr>
<th>Video Sequence</th>
<th>Spatial Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flowervase_WQVGA</td>
<td>416x240</td>
</tr>
<tr>
<td>Flowervase_WVGA</td>
<td>832x480</td>
</tr>
<tr>
<td>City_720p</td>
<td>1280x720</td>
</tr>
<tr>
<td>Jets_720p</td>
<td>1280x720</td>
</tr>
<tr>
<td>Bluesky_1080p</td>
<td>1920x1080</td>
</tr>
<tr>
<td>Station2_1080p</td>
<td>1920x1080</td>
</tr>
</tbody>
</table>

The results obtained for all the sequences tested are presented in Table 2.2 using the well-known Bjontegaard metric for two set of RD points; the Bjontegaard metric enables the comparison of RD curves in terms of the average PSNR improvement or the average per cent bitrate saving. In Table 2.2, BD expresses the bitrate savings with the positive values indicating an increase of the rate and vice-versa.
As shown in Table 2.2, the most significant gains are obtained for the sequences added by the authors, i.e. those containing more complex motion. Beside those, the Cactus sequence also shows a good gain. The average BD gain is 3.1% and 3.5% for the low delay and random access conditions, respectively.

In terms of complexity, the encoder spends 33.6% more time and the decoder spends 20.8% more time with the proposed coding solution regarding the original TMuC as additional motion estimation and interpolation are needed at the encoder and decoder.

In summary, the proposed solution integrates a new coding technique, which proved to reduce the average bitrate (although a rather low auxiliary information rate has to be added) compared to the pure block matching solution. The main drawbacks of the proposed solution regard the selection of only one best warped reference picture which might lead to a conflict between the foreground and background motions; moreover, the algorithm to select the best warped reference picture compares

<table>
<thead>
<tr>
<th>Class</th>
<th>Video Sequence</th>
<th>Low Delay</th>
<th>Random Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Traffic</td>
<td>-</td>
<td>0.0 %</td>
</tr>
<tr>
<td></td>
<td>PeopleOnStreet</td>
<td>-</td>
<td>0.0 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>-</td>
<td>0.0 %</td>
</tr>
<tr>
<td>B</td>
<td>BasketballDrive</td>
<td>0.1 %</td>
<td>-0.2 %</td>
</tr>
<tr>
<td></td>
<td>BQTerrace</td>
<td>0.7 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>Cactus</td>
<td>-3.1 %</td>
<td>-8.4 %</td>
</tr>
<tr>
<td></td>
<td>Kimono</td>
<td>0.1 %</td>
<td>0.0 %</td>
</tr>
<tr>
<td></td>
<td>ParkScene</td>
<td>0.2 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>Bluesky_1080p</td>
<td>-8.4 %</td>
<td>-14.2 %</td>
</tr>
<tr>
<td></td>
<td>Station2_1080p</td>
<td>-41.1 %</td>
<td>-28.9 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>-7.4 %</td>
<td>-7.4 %</td>
</tr>
<tr>
<td>C</td>
<td>BasketballDrill</td>
<td>-0.2 %</td>
<td>0.0 %</td>
</tr>
<tr>
<td></td>
<td>BQMall</td>
<td>0.6 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>PartyScenes</td>
<td>-0.1 %</td>
<td>-0.2 %</td>
</tr>
<tr>
<td></td>
<td>RaceHorses</td>
<td>0.5 %</td>
<td>0.0 %</td>
</tr>
<tr>
<td></td>
<td>Flowervase</td>
<td>-0.7 %</td>
<td>-0.4 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>0.0 %</td>
<td>-0.1 %</td>
</tr>
<tr>
<td>D</td>
<td>BasketballPass</td>
<td>0.6 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>BlowingBubbles</td>
<td>0.2 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>BQSquare</td>
<td>0.3 %</td>
<td>-0.2 %</td>
</tr>
<tr>
<td></td>
<td>RaceHorses</td>
<td>0.8 %</td>
<td>0.1 %</td>
</tr>
<tr>
<td></td>
<td>Flowervase</td>
<td>-6.1 %</td>
<td>-2.3 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>0.0 %</td>
<td>-0.1 %</td>
</tr>
<tr>
<td>E</td>
<td>Vidyo1</td>
<td>0.0 %</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Vidyo3</td>
<td>-0.9 %</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Vidyo4</td>
<td>-0.1 %</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>City</td>
<td>-0.2 %</td>
<td>-3.6 %</td>
</tr>
<tr>
<td></td>
<td>Jets</td>
<td>-12.4 %</td>
<td>-16.4 %</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>-2.7 %</td>
<td>-10.0 %</td>
</tr>
<tr>
<td>Total</td>
<td>Average</td>
<td>-3.1 %</td>
<td>-3.5 %</td>
</tr>
</tbody>
</table>
the \( k \)-th block of one reference picture with the same \( k \)-th block of the current picture, making it very sensitive to motion. An improved algorithm to select the best warped picture may consider a small window around each block to choose the best block match, thus leading to a better warped reference picture as less sensitive to the motion compensation performed.

2.1.3.2 A Block-Adaptive Skip Mode for Inter Prediction Based on Parametric Motion Models

The predictive video codec integrating a geometric transform tool presented in this section has been proposed by Glantz et al. in 2011 [12].

A. Objectives and Basic Approach

The objective of this novel coding solution is to improve the RD performance regarding a reference codec by including an additional zero-residue SKIP coding mode regarding a reference picture generated based on an appropriate geometric transform. With this purpose in mind, the proposed algorithm estimates local translation motion models to extract good motion features of the picture to be coded regarding a reference picture using a tracking algorithm. Then, parametric motion models are calculated using the feature correspondence points with a modified version of a highly robust regression method based on the Helmholtz reciprocity principle. The Helmholtz reciprocity principle states how a ray of light and its reverse ray encounter matched optical adventures, such as reflections, refractions, and absorptions in a passive medium, or at an interface [13]. This means a set of parametric motion models are computed based on the correspondence points obtained above. Then, a selection algorithm chooses the best parametric motion estimation based on the so-called true motion vectors, in this case those obtained by the tracking algorithm, and the motion vectors associated to each parametric motion model. Next, the final parametric motion model is calculated. Herein, it is possible to code macroblocks using the parametric motion model obtained before by defining a new coding mode, labeled as PSKIP (Parametric Skip). The PSKIP mode uses the parametric motion model to reduce the sensitivity to complex motion and does not transmit residual data regarding the parametric motion model generated prediction frame.

B. Architecture and Walkthrough

The basic architecture of the proposed parametric motion model estimation algorithm is shown in Figure 2.10.

![Figure 2.10 – Basic architecture of the proposed parametric motion estimation](image)

To better understand the proposed parametric motion model estimation process, its walkthrough is presented in the following:
• **KLT algorithm** – As for the solution presented in the previous section, a KLT tracking algorithm is applied between two given frames of a video sequence to search for good feature correspondences, as shown in Figure 2.8. The associated motion vectors are also called *true motion vectors*. Then, \( m \) random true motion vectors are grouped to create multiple sets of motion vectors.

• **Motion Model Parameters Estimation** – Herein, a homography transform with only 4 parameters is calculated between those two given frames involved [14]. This means that this homography transform only needs 2 correspondence points to be determined, and thus can only map translations, rotations and scalings. For each subset of correspondence points, \( s \), defined above a homography transform \( H_s \) is thus computed.

• **Outliers removal** – New motion vectors are estimated by applying the obtained homography transform to the correspondence points in the reference picture. The distance between the *true motion vectors* and the estimated motion vectors is computed for each \( H_s \) previously computed. Then, the \( n \)-th percentile\(^2\) of those distances defines a threshold that will be used to divide the estimation motion vectors into inliers and outliers, where \( n \) defines the desired outlier tolerance.

• **Rating** – The \( n \)-th percentile of the distances between true motion vectors and estimated motion vectors allows to derive a standard deviation [14]. Based on this standard deviation, \( \sigma'_s \), and the amount of inliers, \( I_s \), a rating \( \phi_s \) is computed as:

\[
\phi_s = \frac{I_s}{\sigma'_s}
\]

(2.21)

This rating has been chosen because it expresses a relation between the amount of inliers and the standard deviation of those inliers, allowing the best homography transform to be chosen; the higher the rating \( \phi_s \), the lower the deviation between the true motion vectors and the estimated motion vectors, considering the amount of inliers.

• **Determining the final homography transform** – After choosing the subset of correspondence points with the highest rating, \( \phi_s \), a final homography transform with 8 parameters is calculated. This homography transform with 8 parameters can map any quadrilateral onto any quadrilateral, instead of only the translation, rotation and scaling mappings performed by the homography transform with 4 parameters.

• **PSKIP mode** – After obtaining the best homography transform, a new coding/prediction mode (PSKIP mode) can be defined. This PSKIP mode provides the main advantage of any SKIP mode, i.e. no residual data has to be transmitted, while using the estimated parametric motion model to better predict the motion information. It is expected that the PSKIP mode can be added together with both the SKIP and INTER modes in a reference codec HEVC test model HM 1.0, as it will be shown later.

To define the transform used in the novel PSKIP mode, additional information has to be transmitted to the receiver. In this solution, the authors choose to send the 8 uncompressed

\(^2\) A percentile is a value or variable below which a certain percent of observations fall [33]; for example, a 10th percentile represents the value which has below 10% of the observations.
parameters of the homography transform in floating point precision rather than 4 motion vectors with quarter-pel quantization, as done for the previous solution.

In Figure 2.11, the integration of the proposed PSKIP mode in a common hybrid video encoder is presented. As shown, the parametric motion model parameters are always transmitted as side information, whether the picture uses the PSKIP mode or not. The encoder chooses the best coding mode after assessing the PSKIP mode along with the standard INTER and INTRA coding modes.

**C. Main Novel Coding Tool**

The most relevant coding tool included in this algorithm is the homography transform. In this case, two different homography transforms are used: one with 4 parameters and another with 8 parameters. The homography transform with 4 parameters can only map translations, rotations and scalings and it is used because it is simpler to compute as it requires only two correspondence points. The 4 parameters homography transform, $H_x$, is represented as in (2.22) and the 4 coefficients can be calculated as (2.23), where $m_i$, with ($i = 0, ..., 3$), are the 4 parameters, and $(x_k, y_k) \leftrightarrow (x'_k, y'_k)$, with ($k = 1, 2$), are the two correspondences [14].

\[
H_x = \begin{bmatrix}
m_0 & m_1 & m_2 \\
-m_1 & m_0 & m_3
\end{bmatrix}
\]

(2.22)

\[
\begin{bmatrix}
x_1 & y_1 & 1 & 0 \\
y_1 & -x_1 & 0 & 1 \\
x_2 & y_2 & 1 & 0 \\
y_2 & -x_2 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
m_0 \\
m_1 \\
m_2 \\
m_3
\end{bmatrix}
= \begin{bmatrix}
x'_1 \\
y'_1 \\
x'_2 \\
y'_2
\end{bmatrix}
\]

(2.23)

The homography transform with 4 parameters is used for comparing all the extracted parametric motion models. After, when the best parametric motion model is selected, the corresponding 8 parameters homography transform is calculated and used.

**D. Performance Evaluation**

To evaluate the performance of the proposed solution, the new coding mode (PSKIP) was incorporated into the HEVC test model HM 1.0 in addition to all the already available coding modes. The HEVC test model has larger blocks sizes, better interpolation filters and larger transforms than the H.264/AVC, typically leading to bitrate saves between 30% and 40% [15].
A set of sequences were tested to evaluate the performance of the novel coding solution. Table 2.3 lists the sequences with the associated spatial resolution, number of frames, frame rate and the results for the low delay and random access test conditions (as defined by the JCT-VC group in charge of the HEVC development) using the BD-rate and BD-PSNR metrics.

Regarding the low delay and random access test conditions, a coding structure is presented in Figure 2.12, where the dashed boxes represent the current coded frame, the arrows the frames used as prediction and big B frame is a generalized B mode.

![Coding structure](image)

**Figure 2.12** – Coding structure of the test conditions used: (a) low delay and (b) random access (DO regards the display order and CO coding order) [12].

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Resolution</th>
<th>Frames</th>
<th>Frame rate</th>
<th>Low Delay</th>
<th>Random Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlueSky</td>
<td>1920x1080</td>
<td>218</td>
<td>25</td>
<td>-8.2%</td>
<td>-0.9%</td>
</tr>
<tr>
<td>BQSquare</td>
<td>416x240</td>
<td>600</td>
<td>60</td>
<td>-2.8%</td>
<td>-3.6%</td>
</tr>
<tr>
<td>BQTerrace</td>
<td>1920x1080</td>
<td>600</td>
<td>60</td>
<td>-2.0%</td>
<td>-1.4%</td>
</tr>
<tr>
<td>Cactus</td>
<td>1920x1080</td>
<td>500</td>
<td>50</td>
<td>-1.2%</td>
<td>0.1%</td>
</tr>
<tr>
<td>City</td>
<td>1280x720</td>
<td>600</td>
<td>60</td>
<td>-3.6%</td>
<td>-0.5%</td>
</tr>
<tr>
<td>Desert</td>
<td>720x400</td>
<td>240</td>
<td>25</td>
<td>1.2%</td>
<td>1.8%</td>
</tr>
<tr>
<td>Entertainment</td>
<td>720x576</td>
<td>250</td>
<td>25</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>PartyScene</td>
<td>832x480</td>
<td>500</td>
<td>50</td>
<td>-2.3%</td>
<td>-2.8%</td>
</tr>
<tr>
<td>Station2</td>
<td>1920x1080</td>
<td>250</td>
<td>25</td>
<td>-29.1%</td>
<td>-9.7%</td>
</tr>
</tbody>
</table>

As expected, the proposed codec performs better for the low delay test conditions, because the previous frame is always available as reference, leading to better parametric motion estimation. The Desert sequence results, where the performance shows a loss of 1.8%, can be explained by the heat waves present in the video sequence, which lead to poor global motion estimation; therefore, the codec does not use the parametric motion estimation and the associated PSKIP mode. The rate increase can be explained by the PSKIP mode side information, since the parametric motion model parameters are always sent, whether the PSKIP mode is used or not (and the 8 parameters are sent uncompressed). However, for the Station2 sequence, the gain reaches 29.1%, due to the complex motion, notably zoom and rotation, present in the video sequence. Considering that the HEVC test model is already bringing compression gains of about 30% to 40% regarding H.264/AVC, the total potential gain regarding H.264/AVC is significantly high. A good example of the improvements brought
by the PSKIP mode regarding HEVC test model HM 1.0 are presented in Figure 2.13, where it is possible to see that many SKIP and INTER mode coded blocks are replaced by the new PSKIP mode; as the HEVC reference software selects the PSKIP mode, this means it brings better RD performance. Also, the size of each coded macroblock as increased, which means that the global motion estimation is good in this case.

![Figure 2.13 – Example of the PSKIP mode usage: (a) without PSKIP mode; (b) with PSKIP mode available (orange = SKIP, green = INTER, blue = INTRA and magenta = PSKIP) [12].](image)

In summary, the proposed PSKIP mode codec shows a better RD performance than H.264/AVC, notably considering that the additional coding mode has been integrated into the HEVC test model HM 1.0, which already saves up to 40% bitrate regarding H.264/AVC. The selection algorithm to choose the best homography transform is based on a motion criterion, leading to better global motion estimation. However, although the final homography transform has 8 parameters, the selection of the parametric motion relies on a comparison between homography transforms with 4 parameters, which might not lead to the best motion model. Also, the parametric motion can only be used if the residual data is very low, because only a parametric SKIP mode (without residual coding) is available. Considering that the main target of the novel coding mode is to better characterize the background motion, this should not represent a major drawback, but future works might consider including a parametric coding mode with residual data, creating a parametric INTER (PINTER) mode. Another drawback is the fact that the side information containing the parametric motion is transmitted whether the frame uses it or not. So, to reduce the bitrate, the parametric motion estimation should only be transmitted when the parametric coding mode are effectively used or when a certain minimum gain is achieved.
2.2 Reviewing Distributed Video Coding

This subsection intends to briefly review the basics of Distributed Video Coding (DVC) [5] as this will be the coding paradigm adopted in this Thesis. To accomplish this target, this section will first review the main definitions and theorems, proceeding after to a description of one of the most efficient DVC codecs, the DISCOVER Wyner-Ziv (WZ) codec [16]. Finally, some of the most relevant side information creation methods available in the literature are reviewed as this is the core issue on this Thesis.

2.2.1 Basic Definitions and Theorems

Over the years, predictive coding schemes were adopted by the vast majority of the digital video applications. This coding architecture leads to rather complex encoders and simpler decoders, following the typical down-link model such as in digital TV. Nowadays, a set of emerging applications are not well characterized by the down-link model but rather follow an up-link model, which means that some simple devices deliver information to a central, eventually rather complex, receiver. These novel needs led to the emergence of a new video coding paradigm based on the Slepian-Wolf (SW) and the Wyner-Ziv (WZ) theorems, well known as distributed video coding (DVC). One of the DVC targets is to use light encoding systems while achieving the same compression efficiency as the best predictive video coding schemes available. The DVC paradigm also provides improved error resilience, codec independent scalability and a good exploitation of the multi-view correlation without the encoders communicating among them [5].

Consider two or more correlated random sequences, independently and identically distributed (i.d.d.), that are separately encoded and jointly decoded by a single decoder [5] [17], as shown in Figure 2.14. The Slepian-Wolf theorem states that the minimum rate to encode those correlated signals separately is the same as the minimum rate for joint encoding, considering a small error probability. With this, the Distributed Source Coding (DSC) paradigm can, in theory, match the traditional predictive coding paradigm in terms of rate-distortion performance. The separate encoders assume that the video signals are correlated, even though they don’t know the other video signals; this is relevant because the decoder will jointly decode all the video signals, after estimating the correlation between them. Regarding Figure 2.14, since $X$ and $Y$ are correlated but are independently encoded, each sequence represents a “noisy” version of the other sequence, and both will be later jointly decoded by Slepian-Wolf coding techniques.

Figure 2.14 – Slepian-Wolf coding architecture for two correlated signals, $X$ and $Y$, i.d.d. [17].

The Wyner-Ziv theorem states that when performing independent encoding of a sequence, $X$, with the side information, $Y$, available at the decoder, there is no coding efficiency loss, regarding the
predictive coding schemes, even if the coding process is lossy [5], assuming that \( X \) and \( Y \) are jointly Gaussian, memoryless and a mean square error (MSE) distortion measure is used. Therefore, Wyner-Ziv coding focus on the case where a lossy and independent coding of the \( X \) signal is performed, considering that the correlated sequence \( Y \) is available only at the decoder, and not at the encoder. That is, \( Y \) is independently lossy encoded and decoded, while \( X \) is independently encoded but jointly decoded, because it uses the \( Y \) information to conditionally decode \( X \), as shown in Figure 2.15. In other words, the \( Y \) signal is a “noisy” version of the original \( X \) signal, which will be later conditionally decoded. This architecture is named asymmetric coding, and \( Y \) is known as side information (SI).

![Wyner-Ziv coding architecture for two correlated signals, \( X \) and \( Y \), i.d.d. [17].](image)

### 2.2.2 DISCOVER WZ Codec: a Concise Description

The DISCOVER Wyner-Ziv video codec [16] is based on the early Stanford WZ video codec proposed around 2002 [5]. The DISCOVER coding architecture uses a frame-based approach, instead of a block-based approach as adopted by the alternative Berkeley WZ video codec [5]. The Stanford WZ architecture is also characterized by the use of a feedback channel, introducing some delay but allowing the decoder to perform rate control. The architecture of the DISCOVER WZ video codec is presented in Figure 2.16 and is explained below.

![DISCOVER Wyner-Ziv video codec architecture [5].](image)

**A. Encoder walkthrough:**

- **Frame Classification** – The video sequence is divided into Key frames and WZ frames. Key frames are coded as Intra frames, and WZ frames are coded using a WZ approach. The
frequency with which the Key frames are inserted defines the GOP size. Depending on the
temporal correlation in the video, an adaptive GOP size selection might be used. Typically, a GOP
size of 2 is used, leading to a video structure where odd frames are Key frames and even frames
are classified as WZ frames.

- **Discrete Cosine Transform** – An integer $4 \times 4$ block-based DCT is applied to the WZ frames.
Then, the so-called DCT coefficients bands are formed for each WZ frame. Each DCT coefficient
band contains all the DCT coefficients of the $4 \times 4$ blocks in the frame with a specific frequency.

- **Quantization** – A uniform quantization with $2^{M_k}$ levels is performed to compress the data
regarding a target quality, where $2^{M_k}$ depends on the DCT coefficients band $b_k$ as the human
visual system is not equally sensitive to all bands. For a given band, the quantized bits with the
same significance are grouped together, forming the so-called *bit-planes*. This allows the encoder
to separate properly and sort the most important bits in terms of perceptual relevance for a given
coded frame. Quantization of the AC coefficients also allows the encoder to adjust the dynamic
range of each AC band, leading to a smaller difference between each quantum level and, thus, a
more efficient use of the bits available.

- **Low-Density Parity-Check Accumulate (LDPCA) Encoding** – The encoding starts with the
most significant bit-plane of the first DCT band (the DC coefficient band). Once all the bit-planes of
that band are properly encoded, the same procedure is done by the remaining AC coefficient
bands. The encoded information is stored in a buffer waiting to be sent to decoder after request
through the feedback channel; in this way, no more processing is needed, just access to the
memory, after each decoder request. Once requested via the feedback channel, the information is
sent to the decoder in packets. For better decoder control, also the cyclic redundancy check
(CRC) code is computed for each bit-plane and sent to the decoder for error checking.

- **Encoder Rate Estimation** – At the encoder, the amount of information that is sent in the first rate
chunk is estimated to reduce the number of rate requests to be made by the decoder. This
number should be slightly underestimated to leave the decision of requesting more information to
the decoder and avoid the risk of sending useless information.

**B. Decoder walkthrough:**

- **Side Information creation** – The estimation of the side information performed at the decoder is
the most important tool of the DVC paradigm. At this stage, a motion-compensation frame
interpolation process is performed with the intention of finding an estimation of the WZ frame to
code based on some previously decoded frames (one in the past and another in the future), thus
already available. Initially, after both past and future reference frames are identified ($X_b$ and $X_f$,
respectively) and low-pass filtered, a full-search motion estimation with a modified matching
criteria is performed [5]. This algorithm takes into account the amplitude of the motion vectors,
benefiting those which are closer to the origin. Then, a hierarchical coarse-to-fine approach is
used, dealing with larger blocks in the first iteration for a sketch of the motion vector field, which
will be improved by the second iteration, achieving a more detailed motion field through smaller
blocks. A spatial motion smoothing algorithm is performed at this point, to smooth the motion
vector field according to a weighted median. Finally, the interpolated frame is ready to be computed based on bidirectional motion compensation, leading to the so-called side information.

- **DCT Estimation** – A block-based $4 \times 4$ integer DCT is performed over the side information, leading to a frequency band decomposition.

- **Correlation Noise Modeling** – The main target of the correlation noise model (CNM) is to estimate the correlation between the side information of a given WZ frame and the original WZ frame, although without having access to it. Assuming that the residual statistics between the WZ frame DCT coefficients and the side information DCT coefficients follows a Laplacian distribution, the Laplacian parameter can be estimated for both DCT coefficient and DCT band levels, leading to an adaptive estimation of the residual statistics [5]. The CNM is forward to the turbo decoder for more efficient decoding, this means less parity bits to correct the same amount of errors.

- **LDPCA Decoding** – The LDPCA decoder receives the parity information sent by the encoder and processes it together with the already known residual statistics and DCT-transformed side information. If needed after request stopping criterion checking, more information might be requested by the LPDCA decoder through the feedback channel.

- **Request Stopping Criterion** – The LPDCA decoder checks all the LPDC codes parity-check equations. If it is decided that no more bits are needed to decode a given bit-plane, the decoding process goes to the next bit-plane or the next band, until all the bands are properly decoded.

- **CRC check** – Then, a CRC code is used to check the decoded bit-planes and correct eventual residual errors that might appear in the decoded bit-planes. Considering that the data has already been corrected by the LDPCA decoder, a CRC-8 checksum was chosen for the target quality/bitrate trade-off.

- **Reconstruction** – Once all the bit-planes are grouped to obtain the quantized DCT bins for all the blocks of the WZ frame, an optimal reconstruction is performed to obtain all the DCT coefficients values while minimizing the MSE.

- **IDCT** – Herein, a block-based $4 \times 4$ inverse DCT is perform to obtain the reconstructed WZ frame in the pixel domain.

- **Frame Mixing** – The decoded frames are finally ready to be placed in the appropriate video sequence, according to the display order.

To evaluate the DISCOVER WZ codec, its performance was tested against three state-of-the-art standard solutions with an equivalent encoder complexity as no motion estimation is performed: H.263+ Intra, H.264/AVC Intra and H.264/AVC No Motion. Four sequences were chosen (Foreman, Coast Guard, Hall Monitor and Soccer), and the RD performance was obtained for QCIF resolution at 15 Hz and a GOP size of 2. The sequences have 299 frames, except for the Hall Monitor sequence, which has 329 frames. The key frames are encoded with the H.264/AVC Intra (Main profile), because this is considered one of the best image compression solutions. Several RD points are also considered to estimate the overall RD performance for a meaningful range of bitrate/quality; for more information, please see [5].
Figure 2.17 shows the RD performance for the four test sequences. Regarding the Coast Guard sequence, DISCOVER WZ codec shows great performance, gaining up to 1 dB, when compared to the other state-of-the-art standard solutions. With respect to the Hall monitor, the performance almost reaches H.264/AVC No Motion; considering the static background presented in this sequence, this is considered a good performance. For the Soccer sequence, the DISCOVER WZ codec performance is not as good when compared to H.264/AVC Intra and H.264/AVC No Motion, due to the high motion present which leads to lower quality side information. Finally, for the Foreman sequence, the DISCOVER WZ codec shows good performance for the lower bitrates, matching the performance of the H.264/AVC Intra when the bitrate increases. In summary, the DISCOVER WZ codec RD performance seems to be especially promising for video content with low and regular motion like is video surveillance applications.

Regarding the encoding complexity, the DISCOVER WZ encoder shows rather low encoding complexity when compared to the other coding solutions, as shown in Figure 2.18 for the Coast Guard sequence. This leads to a good trade-off between RD performance and encoder complexity [5].

Figure 2.18 – Encoding complexity measured in terms of encoding time for the Coastguard video sequence [5].

### 2.2.3 Most Relevant Side Information Creation Methods

As the side information creation process is the main topic of this Thesis, this subsection intends to review some of the main side information creation techniques available in the literature.
2.2.3.1 Advanced side information creation techniques and framework for Wyner-Ziv video coding

The side information creation solution described in this section has been proposed by Ascenso et al. in 2008 [18]. This SI creation solution was selected for presentation here because it is one of the best performing solutions in the literature, representing here the much adopted block-based interpolation side information creation approach.

A. Objectives and Basic Approach

The objective of the proposed side information creation framework is to allow achieving a better RD performance, after integration in a WZ video codec regarding a reference WZ video codec, by creating more reliable side information. The proposed SI creation solution relies on a set of techniques that, properly combined, achieve improved and more reliable side information. These techniques, described in the following, are: motion estimation with smoothness constraints, bidirectional motion estimation with linear trajectories, hierarchical motion estimation and spatial motion filtering.

B. Architecture and Walkthrough

The architecture of the proposed side information creation framework is presented in Figure 2.19. This framework uses available decoded past and future frames ($X'_p$ and $X'_f$, respectively) as input and delivers an interpolated side information frame.

![Figure 2.19 – Side information creation framework [18].](image)

To better understand the proposed side information creation process, an encoding walkthrough is presented, providing more details for the more interesting side information creation modules in the architecture.

- **Motion estimation with smoothness constraints** – This module performs motion estimation between $X'_p$ and $X'_f$, a past and a future decoded frame, considering the information obtained by the neighborhood blocks, which provides smoother motion vectors. Initially, it searches for the best motion vector $v_i^*$ for the $i$-th block of $X'_f$ on $X'_p$ in terms of distortion, $D_i$, just to have a starting point. The distortion $D_i$ is computed as the mean absolute difference (MAD). Then, to smooth the motion field, a Lagrangian cost function (2.24) is computed for each motion vector, regarding a predetermined search area, with the target to determine the best motion vector from a RD perspective. Note that $\bar{v}_i$ is the obtained motion vector, $\lambda'$ is the Lagrange multiplier, block $B_i$ is a neighbor of block $B_j$ and $U(B_j)$ are all the 8 neighbors of block $B_j$. The $\lambda'$ is adaptively computed and proportional to the sum of the distortions obtained before.
\[
\hat{v}_i = \arg\min_v \left\{ D_i(v) + \lambda' \sum_{j \in U(B_i)} [D_j(v) - D_i(v)] \right\}
\]  
(2.24)

This process uses the motion information of the 8 closest blocks (in a 3 × 3 window of blocks) and the distortion caused by those motion vectors in the chosen block to estimate a new and more reliable motion vector for that block. This process chooses blocks in a random order and it must be repeated a couple of times for all the blocks (authors suggest 10 times) in order to guarantee a stable solution.

- **Adaptive Search Range (ASR)** – This module intends to define a motion vector search area regarding the neighbor blocks. Based on the four closest blocks (top, bottom, left and right) to the block under processing, a search area \((d_x, d_y)\) is defined as in (2.25) for the motion vector of each block, as shown in Figure 2.20.

\[
x_u + L_k \leq d_x \leq x_b - L_k
\]
\[
y_i + L_k \leq d_y \leq y_r - L_k
\]
(2.25)

In (2.25), \((x_u, y_u)\) corresponds to the motion vector of block U, and the same holds for block R, L and B. \(L_k\) represents the block size for a given scale \(k\) (the scales will be detailed later) and \((d_x, d_y)\) represents the searching range.

- **Bidirectional motion estimation with linear trajectories** – Since the previous motion estimation with smoothness constraints is performed from \(X'_f\) to \(X'_b\), some block overlaps and holes might appear on the desired interpolated frame, \(Y_i\). To solve this problem, a regular grid containing the position of all blocks is used for the frame \(Y_i\) and a new motion estimation process is performed with a specific requirement: all the motion vectors must pass through the center of those blocks; this implies taking the interpolated frame, \(Y_i\), as the reference frame in terms of block boundaries.

To do this, the closest motion vector to the center of each block in the frame \(Y_i\) is displaced to the center of that block. Then, motion estimation with linear trajectory is performed, while fixing the center of a given block in frame \(Y_i\), as shown in Figure 2.21. The estimation is performed on both vertical and horizontal directions.
In (2.26), \( v_b \) is the motion vector from frame \( Y_i \) to frame \( X'_b \) and \( d_b \) is the temporal distance between those frames while the same stands for \( v_f \) and \( d_f \). \( s \) is the motion vector used to cover the search area and \([-M_y, M_y] \times [-M_x, M_x]\) represents the area obtained by the ASR algorithm above.

\[
\begin{align*}
    v'_b &= v_b + s \\
    v'_f &= v_b - s \frac{d_b}{d_f} , \quad (s * d_b) |d_f| , s \in [-M_y, M_y] \times [-M_x, M_x]
\end{align*}
\]  

(2.26)

Finally, motion estimation is performed again based on the results of the first module (motion estimation with smoothness constraints), and the best motion vector is chosen.

- **Spatial motion filtering (SMF)** – Herein, a spatial motion filtering is used with the intention to filter noisy motion vectors, leading to a spatially smoother motion vector field. The proposed algorithm uses weighted median filters based on neighboring blocks which might represent better the motion of the block under processing. For each block, the SMF module tests the distortion and the similarity of neighborhood motion vectors, and, combining those two criteria, finds the best motion vector. Therefore, the decision to replace a motion vector by another is based on both distortion and spatial criteria.

- **Hierarchical motion estimation with affine motion model** – To improve the motion estimation, the proposed codec adopts a hierarchical motion estimation approach. This means that the proposed codec starts with larger block sizes, to obtain a coarse and smooth motion estimation, and proceeds after to smaller block sizes, to obtain a finer motion estimation. With this purpose, a block size scale is defined, starting with the coarser scale \( 1 \), with block size \( L_1 \times L_1 \) (typically \( L_1 = 16 \)) until the finer scale \( k \), with block size \( L_k \times L_k \) (\( L_{\text{min}} = 1 \)), corresponding to the pixel-domain. Each block of a given scale \( j \) is split onto four blocks to obtain a finer scale \( j + 1 \).

After the first process of finding the best motion vector field, each block is divided into four blocks of equal size. Once this split happens, the new motion vector for each of the blocks obtained is computed using an affine motion model. This affine motion model uses the 3 closest motion vectors to compute a local vector field that will determine the new motion vector for a given block. The use of an affine transform is motivated by its low complexity and the possibility of compensating some additional complex motions, like zooms and rotation. After all the new blocks have their own motion vector, the algorithm repeats some of the previous steps, notably the ASR, Bidirectional motion estimation with linear trajectories and SMF are performed again, as shown in Figure 2.19. This procedure is performed until \( L_k = 1 \) or until it reaches a minimum block size.
- **Motion compensation** – Finally, when all the motion vectors regarding the smallest block size are computed, the motion vector field is complete and the interpolated side information frame can be constructed by means of motion compensation.

### C. Performance Evaluation

To evaluate the performance of the proposed side information creation solution properly integrated in a WZ video codec, called WZ codec with motion compensated frame interpolation (WZ codec with MCFI) in the following, which details are presented in [18], some video sequences were selected and test conditions defined as presented in Table 2.4. As it is one of the best performing Intra coding schemes available, the H.264/AVC Main profile in Intra mode has been selected to code the key frames.

**Table 2.4 – Test conditions [18].**

<table>
<thead>
<tr>
<th>Video Sequence</th>
<th>Frames</th>
<th>Framerate</th>
<th>Resolution</th>
<th>GOP size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreman</td>
<td>299</td>
<td>30</td>
<td>CIF: 352x288</td>
<td>2</td>
</tr>
<tr>
<td>Hall Monitor</td>
<td>329</td>
<td>15</td>
<td>QCIF: 176x144</td>
<td>2</td>
</tr>
<tr>
<td>Coastguard</td>
<td>299</td>
<td>15</td>
<td>QCIF: 176x144</td>
<td>2</td>
</tr>
<tr>
<td>Soccer</td>
<td>299</td>
<td>30</td>
<td>CIF: 352x288</td>
<td>2</td>
</tr>
</tbody>
</table>

For these experiments, eight rate-distortion points ($Q_d$) were defined with quantization matrices for the WZ frames and quantization parameters (QP) for the key frames; these RD points are defined in [18] and were chosen in order to reach a similar quality for both frame types to avoid major unpleasant quality variations along time.

The proposed codec was tested against H.264/AVC Intra, H.264/AVC zero-motion (Inter mode without motion compensation) and two WZ codecs, one with an extrapolation (and not interpolation) approach, using the previous two decoded frames (WZ codec with SE-1) and another with a bidirectional multi-reference approach, creating two motion vector fields, one forward and another backward, which are simply averaged to create the side information (WZ codec with SE-B) [19].

Initially, the contribution of each module was checked for all the sequences for the RD point $Q_8$ and different block sizes (with $L_{min} = 4$), as shown in Table 2.5. Although this test was performed for the RD point $Q_8$, the same behavior was observed for the other RD points.

**Table 2.5 – PSNR SI quality for each module of the SI creation framework (RD point $Q_8$) [18].**

<table>
<thead>
<tr>
<th>Sequence</th>
<th>$L_k=16$ [dB]</th>
<th>$L_k=8$ [dB]</th>
<th>$L_k=4$ [dB]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASR</td>
<td>ASR+BiME</td>
<td>SMF</td>
</tr>
<tr>
<td>Coastguard</td>
<td>30.25</td>
<td>30.5</td>
<td>30.39</td>
</tr>
<tr>
<td>Hall Monitor</td>
<td>35.46</td>
<td>35.48</td>
<td>35.22</td>
</tr>
<tr>
<td>Foreman</td>
<td>32.01</td>
<td>32.12</td>
<td>32.31</td>
</tr>
</tbody>
</table>

From Table 2.5, it is possible to conclude that the hierarchical approach shows good performance by comparing the RD performance while using different sets of the modules presented above; the performance is better for the lower resolution videos. The ASR+BiME solution also proved to be quite efficient (versus only ASR) for all sequences. Considering the SMF module, it does not perform well for low resolutions (QCIF), but it shows good improvements for CIF resolutions.
In Figure 2.22, some RD performance results for the proposed WZ codec are presented in comparison with the adopted benchmarks already defined.

*Figure 2.22 – RD results regarding the tested sequences [18].*

Considering Figure 2.22 (Coastguard sequence), the proposed codec performs very well regarding the benchmarks, due to the strong background motion, which can be properly estimated by the decoder but is not appropriately exploited by the standard benchmark codecs. With respect to the Hall Monitor sequence (Figure 2.22), the proposed algorithm is only surpassed by the H.264/AVC zero motion, due to the presence of a static background which is very efficient coded with this codec, notably using the SKIP mode. For the Foreman sequence, some gains can be observed for the low/medium bitrates. Considering the Soccer sequence (Figure 2.22), the SI motion vector field is not properly estimated due to the high number of moving objects which leads to RD performance losses when compared to the H.264/AVC Intra and H.264/AVC zero motion solutions which basically uses intra coded macroblocks. Regarding the other WZ codecs tested, the proposed codec outperforms them largely, gaining up to 2 dB for the Foreman sequence, for example.

In terms of the encoder complexity, the proposed codec outperforms both H.264/AVC solutions, concluding that the WZ coding has a great development potential as a video coding solution providing a good trade-off between RD performance and encoder complexity [18].

In conclusion, the proposed advanced side information creation framework is a powerful side information creation solution obtaining encouraging RD performance when compared to both H.264/AVC ‘low complexity’ encoding solutions and WZ coding solutions available in the literature. As such, it will be an important benchmark for the solution to be proposed in this Thesis.
2.2.3.2 Mesh-Based Motion-Compensated Interpolation for Side Information Extraction in Distributed Video Coding

The hybrid block based and mesh-based solution described in this section has been proposed by Kubasov et al. in 2006 [20]. This SI creation solution was selected for presentation here due to the original hybrid motion model it adopts.

A. Objectives and Basic Approach

The objective of the proposed solution is to use a mesh-based approach in order to obtain a better frame interpolation and, thus, a better RD performance regarding a reference codec. Initially, the proposed algorithm calculates a mesh for the past reference frame and the deformation of that mesh regarding the future reference frame. Then, the algorithm compares the MSE obtained with this mesh-based approach and the typical block-based approach (as in the previous section) to decide, pixel by pixel, which one will be used to compute the final interpolated frame. These results in a hybrid solution where the inclusion of the mesh based side information creation branch should improve the available block based side information creation branch performance trying to combine the best of two different motion models.

B. Architecture and Walkthrough

The basic architecture of the proposed hybrid side information creation solution is presented in Figure 2.23.

Mesh construction – Initially, to construct the 2D mesh, a Delaunay triangular regular mesh is applied to the past reference frame; this mesh is created as starting point (see Figure 2.24 (a)).

Mesh position estimation – Herein, a motion estimation algorithm computes a mesh-based optical flow based on the differences between the two reference frames, one in the past and another in the future. The displacement \( \vec{u}(s) = (dx(s), dy(s)) \) for a given point \( s \) of a certain triangle \( T \) is calculated, based on the vertices of that triangle, as in (2.27).

---

3 In mathematics and computational geometry, a Delaunay triangulation for a set \( P \) of points in a plane is a triangulation \( DT(P) \) such that no point in \( P \) is inside the circumcircle of any triangle in \( DT(P) \). Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation; they tend to avoid skinny triangles.
\[
\begin{align*}
    dx(s) &= \sum_{j \in I} w_j(s) \cdot dx_j \\
    dy(s) &= \sum_{j \in I} w_j(s) \cdot dy_j
\end{align*}
\]

(2.27)

In (2.27), \((dx_j, dy_j)\) are the coordinates of the three vertices of the triangle and \(w_j(s)\) represents the weights associated to each coordinate, calculated as the barycentric coordinates of point \(s\) with respect to each vertex \(j\). To obtain the optimal vectors \(\left\{ (dx_j, dy_j) \right\}_{j=1}^{N} \) for each vertex present in the mesh, a MSE approach is used.

\[
    MSE = \sum_{s \in \Omega_1, (s + \bar{u}(s)) \in \Omega_2} \left[ I_1(s) - I_2(s + \bar{u}(s)) \right]^2
\]

(2.28)

In (2.28), \(I_1(s)\) and \(I_2(s)\) are the original and reference key frames, and \(\Omega\) defines the support for the motion estimation. Since the vector \(\bar{u}(s)\) is calculated via (2.27) and \(\left\{ (dx_j, dy_j) \right\}_{j=1}^{N} \) are calculated using (2.28), a Newton-Gauss differential algorithm and a Levenberg-Marquard algorithm are used to obtain the motion vectors minimizing the MSE. Finally, based on the motion vector field associated to the vertices of the regular triangular grid in the past reference frame a deformed mesh is created for the future reference frame (see Figure 2.24 (b)).

![Figure 2.24](image)

Figure 2.24 – Mesh deformation: (a) starting mesh for the past reference frame and (b) deformed mesh for the future reference frame [20].

- **Frame Interpolation** – At this point, a frame interpolation process is performed to obtain an estimation for the frame to be coded; this interpolation is based on the mesh-based motion vector field already obtained for the mesh vertices.

- **Block-based motion estimation** – In parallel, a block-based side information creation solution is also applied, in this case using an algorithm proposed by Ascenso et al., detailed in [21], and which software was provided by the IST and DISCOVER software development teams; the output is an interpolated frame as in the previous step but, in this case, using an available interpolation solution.

- **Hybrid Interpolation** – This module targets the creation of the final WZ interpolated frame, \(Y\), using the two interpolated frames already estimated. Initially, this algorithm computes the MSE between the two reference frames regarding the mesh-based model and the block-based model.
with a penalty term proportional to the amplitude of the motion vector, since large motion vectors usually do not represent well the block motion. The interpolated frame is estimates as:

\[
Y(x,y) = \begin{cases} 
I_{\text{comp}}(x,y,\vec{b}) & \text{if } \text{MSE}(x,y,\vec{b}) \cdot f(Pen(\vec{b})) < \text{MSE}(x,y,\vec{m}) \\
I_{\text{comp}}(x,y,\vec{m}) & \text{otherwise}
\end{cases}
\] (2.29)

In (2.29), a pixel domain choice is made considering the MSE for each approach, where \(\vec{b}\) and \(\vec{m}\) are the motion vectors regarding the block-based and mesh-based approaches, respectively. \(Pen(\vec{b})\) is the average Euclidean distance between the vector \(\vec{b}\) of the block containing \((x,y)\) and the vectors from the nine neighboring blocks, and \(f(d)\) is a monotonous function applied to normalize the obtained penalty. Once all the pixels are estimated, the final interpolated frame \(Y\) is created; examples of the final interpolated frames are presented in Figure 2.25.

**Figure 2.25** – Frame interpolated with: (a) block based approach, (b) mesh-based approach and (c) proposed hybrid solution, (d) signals the use of the block-based approach (light pixels) versus the mesh-based approach (dark pixels) for the proposed hybrid SI creation solution [20].

### C. Performance Evaluation

To evaluate its performance, the proposed hybrid SI creation solution was integrated in a transform-domain distributed video codec. As detailed earlier, the block-based hierarchical full search motion estimation was based on the solution proposed in [21]. The RD performance was obtained using the Foreman sequence, QCIF-30Hz, and a GOP size of 2. Along with the block-based, mesh-based and hybrid approaches, a state-of-the-art transform-domain WZ codec from [22], denoted as Aaron, and the H.263+ codec with a I-P-I-P mode are also considered. Four RD points were adopted to estimate the overall RD performance. The RD performance for the Foreman sequence is presented in Figure 2.26.
Considering the mesh-based and the block-based approaches, it is possible to conclude that the mesh-based motion estimation solution provides better results, gaining up to 1 dB. Regarding the proposed hybrid solution, it is possible to see that the RD performance is quite similar to the mesh-based approach, concluding that the proposed hybrid solution relies mainly on the mesh-based motion estimation branch. The H.263+ solution leads to the best RD performance, probably due to the encoder higher complexity which uses P frames. It is also worth noticing the large gains of the proposed hybrid solution when compared to the state-of-the-art transform-domain WZ codec (Aaron). In conclusion, the proposed mesh-based approach proved to bring significant gains regarding the traditional block-based approach. This mesh-based approach showed great potential, which might be improved in the future; however, the block-based WZ approach has improved significantly since these experiments.

2.2.3.3 High Order Motion Interpolation for Side Information Improvement in DVC

The high order motion interpolation solution for side information creation described in this section has been proposed by Petrazzuoli et al. in 2010 [23].

A. Objectives and Basic Approach

The main objective of the proposed side information creation method based on high order motion interpolation is to improve the side information estimation process through the use of more reference key frames and more complex motion models, targeting an RD performance increase regarding the DISCOVER WZ codec benchmarking. The basic idea is here to obtain better SI quality by better modeling the motion field using more decoded frames; naturally some additional decoder complexity is involved in this process.

B. Architecture and Walkthrough

This method consists in three main steps: i) motion estimation considering two additional key frames (one in the past and another in the future); ii) motion vector interpolation in order to achieve a more complex estimation for each block; and iii) a final adjustment of the vectors to the center of the blocks of the interpolated frame. The basic architecture of the proposed high order interpolation method is presented in Figure 2.27.
The walkthrough for the proposed high order interpolation SI creation method is presented in the following:

- **DISCOVER Side Information Creation** – Considering the frame to be interpolated, $I_k$, and a GOP size of 2, the DISCOVER side information creation process provides motion vectors based on the key frames $I_{k-1}$ and $I_{k+1}$, as shown in Figure 2.28, and described in Section 2.2.2.

  ![Figure 2.28](image)

  **Figure 2.28** – Bidirectional motion estimation as performed in the DISCOVER WZ codec [23].

  In Figure 2.28, $v$ is the motion vector field from $I_{k+1}$ to $I_{k-1}$, $u$ is the motion vector field from $I_k$ to $I_{k-1}$ and $w$ is the motion vector field from $I_k$ to $I_{k+1}$, the points $p$ represent the centers of given blocks in frame $I_k$ and points $q$ refer to the points where the motion vectors $v$ cross the $I_k$ frame. In the DISCOVER SI creation method both the backward and forward motion vector fields are generate in the process leading to the creation of the interpolated WZ frame.

- **Additional Motion estimation** – After obtaining the motion vector fields regarding the two closest key frames as in the DISCOVER SI creation process, this module intends to estimate better motion vectors considering the four closest key frames. Considering a given block centred in $p$ of frame $I_k$, $B_k^p$, the estimated block in $I_{k-1}$ is $B_{k-1}^{p+u(p)}$. Initially, this method intends to find and estimation for that block $B_{k-1}^{p+u(p)}$ in frame $I_{k-3}$ using a simple block matching approach. To perform this, the following equation is minimized to obtain the best motion estimation:

  \[
  f(\tilde{u}) = \sum_{q \in \text{Window}} \left| B_{k-1}^{p+u(p)}(q) - B_{k-3}^{p+\tilde{u}}(q) \right|^n + \lambda \| \tilde{u} - 3u \| \tag{2.30}
  \]

  where $\tilde{u}$ is the best motion vector between $I_k$ and $I_{k-3}$ and the last term penalizes the vectors which deviate too much from the motion vector field $u$. The sum of absolute differences (SAD) or the sum of squared differences (SSD) can be used to compute $\tilde{u}$ by simply setting $n = 1$ or $n = 2$, respectively. The same procedure is performed to find the other motion vector field, $\tilde{w}$, considering frames $I_{k+1}$ and $I_{k+3}$.
- **Motion vector interpolation** – This module intends to estimate the position of a given block in the interpolated frame considering the backward and forward motion vectors, leading to an eventual complex (non-linear) motion. For that, the instants of the four frames \((k-3, k-1, k+1, k+3)\) and their respective motion vectors \((p+u, p+w)\) are interpolated to obtain the value at instant \(k\) \((\hat{y})\), as shown in Figure 2.29.

- **Motion vectors adjustment** – This last module intends to adjust the motion trajectory to the center of a given block in the interpolated frame. The motion trajectory chosen is the closest to each block center.

![Diagram of motion vector interpolation and adjustment](image)

**Figure 2.29 – Proposed high order interpolation method for motion estimation [23].**

### C. Performance Evaluation

Before evaluating the proposed method performance, some experiments were conducted in order to define some metrics and parameters that were not defined before, such as the selection between the SAD and SSD metrics, the \(\lambda\) parameter referring to the penalty for the deviation of the motion vectors, and the fractional motion vectors precision. Additionally, the best GOP size was also studied. The selected test sequences were *Book arrival*, *Ballet*, *Jungle* and *Breakdance* [23].

- **SAD and SSD metrics** – Some SI quality tests were performed with these two alternative metrics, which led to quite similar results, with a slightly advantage for the SAD metric. Since the SAD is computationally lighter, this was the selected error metric for the remaining experiments.

- **Fractional motion vectors precision** – The performance for full, half and quarter pixel motion vector precision was tested against the DISCOVER solution, considering the SAD metric and \(\lambda = 0\). The results are presented in Table 2.6; it is possible to conclude that it is not worth going more than half pixel precision as this increases the complexity without RD performance gains.

**Table 2.6 – \(\Delta_{PSNR}\) [dB] for full, half and quarter pixel precisions regarding DISCOVER WZ codec [23].**

<table>
<thead>
<tr>
<th>Precision</th>
<th>Ballet</th>
<th>Book arrival</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lossless keyframes</td>
<td>QP=31 for keyframes</td>
</tr>
<tr>
<td>Full pixel</td>
<td>0.318</td>
<td>0.277</td>
</tr>
<tr>
<td>Half pixel</td>
<td>0.319</td>
<td>0.274</td>
</tr>
<tr>
<td>Quarter pixel</td>
<td>0.320</td>
<td>0.269</td>
</tr>
</tbody>
</table>
- **λ parameter** – To define the λ parameter, 10 values between 0 and 100 were considered and the optimal results associate to the maximization of the PSNR over the full test sequences are presented in Table 2.7.

<table>
<thead>
<tr>
<th>GOP size</th>
<th>2</th>
<th>4</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>λ_{opt}</td>
<td>50</td>
<td>20</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2.7 – Optimal value for λ for different GOP sizes [23].

- **GOP size** – Additionally, the performance was also tested for different GOP sizes in terms of SI quality. The results in Table 2.8 reveal that the highest SI quality gains regarding DISCOVER are achieved with a GOP size of 4, reaching 0.5 dB gain in the most favorable cases. With a GOP size of 2, the linear interpolation performed by DISCOVER is good, so the gains obtained are not too high. Considering a GOP size of 8, both methods estimate poor side information, so the gains presented are not too good either.

Table 2.8 - ΔPSNR [dB] for different GOP sizes regarding DISCOVER [23].

<table>
<thead>
<tr>
<th>QP for keyframes</th>
<th>Book arrival</th>
<th>Ballet</th>
<th>Jungle</th>
<th>Breakdance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GOP size of 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lossless</td>
<td>0.356</td>
<td>0.460</td>
<td>0.165</td>
<td>0.055</td>
</tr>
<tr>
<td>31</td>
<td>0.326</td>
<td>0.348</td>
<td>0.150</td>
<td>0.053</td>
</tr>
<tr>
<td>34</td>
<td>0.291</td>
<td>0.313</td>
<td>0.139</td>
<td>0.054</td>
</tr>
<tr>
<td>37</td>
<td>0.252</td>
<td>0.238</td>
<td>0.123</td>
<td>0.049</td>
</tr>
<tr>
<td>40</td>
<td>0.204</td>
<td>0.204</td>
<td>0.101</td>
<td>0.044</td>
</tr>
<tr>
<td><strong>GOP size of 4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lossless</td>
<td>0.523</td>
<td>0.301</td>
<td>0.387</td>
<td>0.133</td>
</tr>
<tr>
<td>31</td>
<td>0.471</td>
<td>0.290</td>
<td>0.369</td>
<td>0.127</td>
</tr>
<tr>
<td>34</td>
<td>0.464</td>
<td>0.270</td>
<td>0.360</td>
<td>0.121</td>
</tr>
<tr>
<td>37</td>
<td>0.422</td>
<td>0.236</td>
<td>0.341</td>
<td>0.116</td>
</tr>
<tr>
<td>40</td>
<td>0.392</td>
<td>0.202</td>
<td>0.314</td>
<td>0.104</td>
</tr>
<tr>
<td><strong>GOP size of 8</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lossless</td>
<td>0.226</td>
<td>0.060</td>
<td>0.037</td>
<td>0.036</td>
</tr>
<tr>
<td>31</td>
<td>0.234</td>
<td>0.045</td>
<td>0.028</td>
<td>0.041</td>
</tr>
<tr>
<td>34</td>
<td>0.230</td>
<td>0.045</td>
<td>0.010</td>
<td>0.032</td>
</tr>
<tr>
<td>37</td>
<td>0.230</td>
<td>0.033</td>
<td>0.000</td>
<td>0.028</td>
</tr>
<tr>
<td>40</td>
<td>0.198</td>
<td>0.027</td>
<td>0.000</td>
<td>0.025</td>
</tr>
</tbody>
</table>

Finally, a complete end-to-end test was performed. The proposed high order interpolation SI creation method was integrated in the DISCOVER WZ codec and the RD performance for the two WZ codecs was compared using a Bjontegaard metric; the results are presented in Table 2.9.
Table 2.9 – RD Bjontegaard metrics comparing the proposed and the DISCOVER SI creation processes [23].

<table>
<thead>
<tr>
<th></th>
<th>Book arrival</th>
<th>Ballet</th>
<th>Jungle</th>
<th>Breakdance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GOP size of 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta_R$ (%)</td>
<td>-3.218</td>
<td>-0.878</td>
<td>-1.929</td>
<td>-1.904</td>
</tr>
<tr>
<td>$\Delta_{PSNR}$ [dB]</td>
<td>0.141</td>
<td>0.044</td>
<td>0.076</td>
<td>0.076</td>
</tr>
<tr>
<td><strong>GOP size of 4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta_R$ (%)</td>
<td>-2.637</td>
<td>-1.072</td>
<td>-1.989</td>
<td>-2.257</td>
</tr>
<tr>
<td>$\Delta_{PSNR}$ [dB]</td>
<td>0.116</td>
<td>0.054</td>
<td>0.078</td>
<td>0.093</td>
</tr>
<tr>
<td><strong>GOP size of 8</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta_R$ (%)</td>
<td>-3.333</td>
<td>-1.073</td>
<td>-1.953</td>
<td>-1.323</td>
</tr>
<tr>
<td>$\Delta_{PSNR}$ [dB]</td>
<td>0.054</td>
<td>0.150</td>
<td>0.077</td>
<td>0.059</td>
</tr>
</tbody>
</table>

Table 2.9 shows that it is possible to reach a rate reduction up to 3.33% and a quality improvement up to 0.15 dB, regarding the DISCOVER WZ codec.

In conclusion, the proposed high order interpolation method provides significant gains in terms of side information quality (up to 0.52 dB in Table 2.8). The novel SI creation solution requires a small increase on the overall decoding complexity, leading to a good trade-off between complexity and RD performance; regarding the encoder, no changes are needed in comparison with the DISCOVER WZ encoder.
Chapter 3

Unidirectional Warping Side Information Creation: Architecture and Tools

Geometric transforms are a powerful and efficient way to characterize complex motion and deformations like zooms, rotations and shears. However, geometric transforms are not widely used to model motion in predictive video coding solutions, notably due to the high rate overhead necessary to transmit the motion model parameters to the decoder. This overhead may lead to a significant increase of the coding rate when geometric transforms are used, thus negatively affecting the compression efficiency gains that may be obtained. However, in the distributed video coding paradigm, the temporal redundancy of the video sequence is exploited at the decoder by using appropriate motion estimation and compensation tools and, thus, no motion data is transmitted from the encoder. In this context, geometric transforms can be used at the decoder to model the motion, without the negative impact of the overhead rate incurred in predictive video coding solutions. In this chapter, geometric transforms are applied to generate the side information at a distributed video decoder, with the expectation of obtaining high quality side information and overall compression efficiency gains when compared to state-of-the-art alternative solutions.

The side information creation is one of the most important tools in a distributed video codec due to its major impact in the final RD performance. The side information is an estimation of the original frame to code created at the decoder, usually an interpolated frame between two previously decoded reference frames, one temporally in the past and another in the future. The interpolation approach, in opposition to the extrapolation approach, leads to some coding delay (associate to the transmission of the future reference frame) which is compensated with better overall RD performance. In the available literature, side information creation mostly uses a translational motion model to estimate the interpolated frame. However, the translational motion model has shortcomings when it is necessary to represent complex motions such as rotations, zooms and other deformations. Therefore, a geometric motion model might overcome some of the translational motion model drawbacks and provide better adaptations to complex moving regions.
As described in Section 2.2.2, the side information must be estimated at the decoder without the availability of the original WZ frames to be coded. As shown for the DISCOVER WZ video coding solution in the same section, side information creation is rather important to the final RD performance as, typically, an increase of the overall SI quality leads to fewer rate requests by the decoder and higher decoded quality; this improves the RD performance and reduces the decoding complexity (and delay).

Considering the objectives of this Thesis, two main WZ video coding solutions are proposed depending on the number of directions (one or two) considered for the motion estimation process performed in the context of the side information creation process: a Unidirectional Warping Side Information (UWSI) creation solution and a Bidirectional Warping Side Information (BWSI) creation solution. These solutions intend to estimate motion trajectories between the reference frames, targeting the provision of coherent and reliable characterizations of the true motion in the sequence. The motion estimation is performed without any help from the encoder, i.e. using only the previously decoded reference frames which may be both key frames and WZ frames, depending on the GOP size. After, the interpolated SI frame \( Y_s \) is created by motion compensation using the extracted motion vector field(s). Both the UWSI and BWSI solutions can provide higher SI quality than the MCFI SI creation solution adopted in the context of the DISCOVER WZ codec which architecture is shown in Figure 16. It is also worth noticing that the proposed tools are only included in the decoder, i.e. without changing the encoder, thus, keeping the encoding complexity low.

While the UWSI solution is presented in this chapter, the BWSI solution will be presented in the next chapter, with both solutions finally evaluated in Chapter 5. In Section 3.1, the architecture and the walkthrough of the proposed UWSI creation method is presented, while Section 3.2 will present some useful details on the geometric transforms processing. Next, Section 3.3 will describe all the procedures and tools adopted with Section 3.4 providing some final remarks.

### 3.1 UWSI Creation Framework

The proposed UWSI creation solution makes use of geometric transforms to estimate the decoder side information for each block by warping appropriate regions from the reference frames. In Figure 3.1, the high-level architecture of the proposed UWSI creation solution is presented.

![Figure 3.1 – Architecture of the proposed UWSI creation solution.](image)

Since there is currently no SI creation solution in the literature based on geometric transforms (GT), the proposed UWSI creation technique is mostly novel; in Figure 3.1, the shaded (blue) blocks correspond to the new proposed techniques, namely the Reference Frames Up-sampling, Bidirectional Geometric ME, Block Size Adaptation, Motion Model Decision and Motion Compensation...
modules. UWSI creation also includes a Backward Motion Estimation (Backward ME) module and a Bidirectional Translational ME module, implemented using conventional solutions, notably as proposed in [24]. These methods provide a starting point for the estimation of the GT motion vector field. The UWSI architecture includes two SI estimation branches, one corresponding to the Motion Compensated Frame Interpolation (MCFI) solution adopted in the DISCOVER WZ codec and a novel one based on geometric transforms. The proposed architecture should exploit the best of these two SI creation approaches by selecting, at block level, one of the two ways to generate the side information frame. The UWSI creation framework proceeds as follows:

1. **Backward ME** – As performed in MCFI, the backward and forward reference frames (\(X_b\) and \(X_f\), respectively) are used to perform backward motion estimation (ME) using a weighted MAD criteria [24] to obtain a starting point for the estimation of the motion vector field. Note that reference frames correspond to key frames or WZ frames (depending on the GOP size) already decoded. In this way, a motion vector is obtained for each 16 × 16 block of the forward reference frame. Then, translational motion vectors are determined for each SI block based on the estimated backward motion vector field in order to have a motion vector for each SI block, thus avoiding holes or overlapping of blocks in the SI frame.

2. **Bidirectional Translational ME (16 × 16)** – After, bidirectional translational motion estimation is performed for 16 × 16 block sizes as described in MCFI [24] to refine the block level translational motion vectors obtained in the previous step. This module uses the techniques proposed in [24], such as the Adaptive Search Range and the Spatial Motion Filtering. This algorithm assumes a linear trajectory between the reference and SI frames and guarantees that every translational motion vector must cross the center of the corresponding SI block.

3. **Bidirectional Translational ME (8 × 8)** – To obtain a more dense and detailed translational motion field, bidirectional translational ME is performed again, now with a reduced block size which goes 16 × 16 to 8 × 8 [24]. So, the algorithm defined in Step 2 is repeated for the 8 × 8 block size.

4. **Reference Frames Up-sampling** – The backward and forward reference frames are up-sampled by a factor of 4 to provide a more accurate estimation for the possible geometric deformations; this up-sampling factor is related to the fact that quarter-pel motion estimation will be performed. Thus, two up-sampled reference frames are created with the H.264/AVC quarter-pel up-sampling filter [1], so that the next tools can benefit from the increased precision of the reference frames.

5. **Bidirectional Geometric ME** – Using the translational motion vectors estimated in Step 2, a perspective geometric transform for each 16 × 16 block is estimated. This GT estimation aims to better characterize the motion between the reference frames when compared to the typical MCFI solution [24].

6. **Block Size Adaptation** – Then, to refine the GT vectors obtained for 16 × 16 blocks, Step 5 is applied again but now for smaller block sizes of 8 × 8, thus obtaining more detailed geometric transforms. So, the block size adaptation module transforms the GT vectors for 16 × 16 blocks into GT vectors for 8 × 8 blocks, i.e. a scale change is applied to all GT vectors. Only then, Step 5 can be repeated for 8 × 8 block sizes.
7. **Motion Model Decision** – To obtain a more reliable SI estimation, the novel GT mode corresponding to the GT vectors obtained in Step 6 is compared with the traditional translational mode corresponding to the MCFI solution [24]. So, the Motion Model Decision step compares and selects, for each block, the best motion estimation model using a MAD criterion, thus identifying at block level the best of the two SI creation alternative approaches. To perform this decision, the residual error for the translational mode obtained in the Step 3 and the residual error for the GT mode obtained in Step 6 are used.

8. **Motion Compensation** – Finally, using the translational motion field or the GT motion field data selected for each block in Step 7, bidirectional motion compensation is performed to create the SI frame.

### 3.2 Geometric Transforms

As mentioned in Chapter 2, there are many available geometric transforms and, thus, the first decision for this Thesis is to select the geometric transform to be used. Since it allows to significantly improve the motion estimation accuracy and there was past experience in the multimedia signal processing research group, it was decided to adopt the perspective geometric transform as the geometric motion model. In this context, some details on this geometric transform are presented in the following because this is important for several modules in the architecture. The perspective transform provides a quadrilateral to quadrilateral mapping with the following representation:

\[
[x, y, w] = [u, v, 1] \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]

(3.1)

where \(a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}, a_{31}, a_{32}, a_{33}\) are the GT parameters, \((u, v)\) are the input quadrilateral positions and \((x, y)\) are the corresponding output warped positions. From (3.1), it is possible to obtain:

\[
x = \frac{u a_{11} + v a_{21} + a_{31}}{u a_{13} + v a_{23} + a_{33}}
\]

\[
y = \frac{u a_{12} + v a_{22} + a_{32}}{u a_{13} + v a_{23} + a_{33}}
\]

(3.2)

To calculate the GT parameters, four pair of positions and their correspondence positions are needed. So, the vertices of both input and output quadrilaterals are used, i.e. \((u_i, v_i)\) for the input and \((x_i, y_i)\) for the output, with \(i \in [0,1,2,3]\).

As usual, it is assumed that \(a_{33} = 1\) (see Section 2.1.1) and the eight GT parameters are determined by solving the linear system in (3.3), using the four vertices of the input quadrilateral and the four vertices of the output quadrilateral.

\[
\begin{bmatrix}
u_0 & 0 & 0 & 0 & -u_0 x_0 & -v_0 x_0 & a_{11} & x_0 \\
u_1 & 1 & 0 & 0 & -u_1 x_1 & -v_1 x_1 & a_{21} & x_1 \\
u_2 & 0 & 1 & 0 & -u_2 x_2 & -v_2 x_2 & a_{31} & x_2 \\
u_3 & 1 & 0 & 0 & -u_3 x_3 & -v_3 x_3 & a_{12} & x_3 \\
0 & 0 & 0 & u_0 & v_0 & 1 & -u_0 y_0 & -v_0 y_0 & a_{22} & y_0 \\
0 & 0 & 0 & u_1 & v_1 & 1 & -u_1 y_1 & -v_1 y_1 & a_{32} & y_1 \\
0 & 0 & 0 & u_2 & v_2 & 1 & -u_2 y_2 & -v_2 y_2 & a_{13} & y_2 \\
0 & 0 & 0 & u_3 & v_3 & 1 & -u_3 y_3 & -v_3 y_3 & a_{23} & y_3
\end{bmatrix}
= \begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
x_3 \\
y_0 \\
y_1 \\
y_2 \\
y_3
\end{bmatrix}
\]

(3.3)
Since the UWSI creation solution proposed in this chapter only uses square to quadrilateral mappings, as shown in Figure 3.2, the linear system in (3.3) can be simplified.

Figure 3.2 – Quadrilateral to square mapping [6].

Assuming that \((u_0, v_0)\) are the vertices of a square block, where the upper left vertex \((u_0, v_0)\) is the reference point, and taking \((x_i, y_i)\) as the vertices of the quadrilateral (warped positions), the linear system presented in (3.3) can be used to calculate the GT parameters with lower complexity obtaining:

\[
a_{11} = \frac{x_1 - x_0}{N-1} + a_{13} * x_1
\]

\[
a_{12} = \frac{y_1 - y_0}{N-1} + a_{13} * y_1
\]

\[
a_{13} = \frac{(N-1) * a_{21} + (N-1) * a_{31} - x_2 - (N-1) * a_{23} * x_2}{(N-1) * x_2}
\]

\[
a_{21} = \frac{x_3 - x_0}{N-1} + a_{23} * x_3
\]

\[
a_{22} = \frac{y_3 - y_0}{N-1} + a_{23} * y_3
\]

\[
a_{23} = \frac{(N-1) * a_{22} + (N-1) * a_{32} - y_2 - (N-1) * a_{13} * y_2}{(N-1) * y_2}
\]

\[
a_{31} = x_0
\]

\[
a_{32} = y_0
\]

where \(N\) is the block size. To easily obtain all GT parameters, coefficients \(a_{13}\) and \(a_{23}\) are calculated first; the following terms \(\Delta x_j\) and \(\Delta y_j\), where \(j \in [0,1,2]\), are defined as in (3.5) to compute \(a_{13}\) and \(a_{23}\) GT parameters.

\[
\Delta x_1 = x_1 - x_2
\]

\[
\Delta x_2 = x_3 - x_2
\]

\[
\Delta x_3 = x_2 - x_3 + x_0 - x_1
\]

\[
\Delta y_1 = y_1 - y_2
\]

\[
\Delta y_2 = y_3 - y_2
\]

\[
\Delta y_3 = y_2 - y_3 + y_0 - y_1
\]

So, the GT parameters \(a_{13}\) and \(a_{23}\) can be obtained from:

\[
a_{13} = \frac{1}{(N-1)} * \begin{vmatrix} \Delta x_3 & \Delta x_2 \\ \Delta y_3 & \Delta y_2 \\ \Delta x_1 & \Delta x_2 \\ \Delta y_1 & \Delta y_2 \end{vmatrix}
\]

(3.6)
After computing both $a_{13}$ and $a_{23}$, the other six GT parameters $\{a_{11}, a_{12}, a_{21}, a_{22}, a_{31}, a_{32}\}$ can be computed with (3.4). Finally, the warped coordinates $(x, y)$ associated to the correspondence $(u, v)$ points can be computed using (3.2) and the GT parameters previously calculated.

### 3.3 UWSI Creation Tools

The tools associated to the various modules in the architecture presented in Figure 3.1 are now described in the following sections. More detail is provided for the tools related to geometric motion modeling as they are the major contribution of this Thesis.

#### 3.3.1 Backward Motion Estimation

This module receives as input the two reference frames and estimates the translational motion field between those reference frames without having access to any information about the original frame. Backward motion estimation is performed as described in the MCFI solution [24] and proceeds as follows:

1. Initially, the two relevant reference frames are identified. For a GOP size of two, the reference frames are the two neighboring key frames of the interpolated frame, one in the past and another in the future. If a larger GOP size is used, previously decoded WZ frames are also used as reference frames.

2. Then, the two reference frames are low-pass filtered to obtain later a more spatially coherent motion vector field. Motion estimation is then performed from $x_f$ to $x_b$, i.e. in a backward direction. For this, the well-known block matching algorithm (BMA) employing a modified matching criterion is used [24]; the modified criterion adds weights to the MAD criterion to favor translational motion vectors closer to the block center in order to regularize the translational motion field. The block size is 16 x 16 and full pixel accuracy is used.

3. Then, a motion vector field is obtained for the SI frame, i.e. from the translational motion vectors calculated in Step 2, some translational motion vector is selected for each SI frame block. In this case, a simple algorithm that selects the closest motion vector to the center of each SI frame block is used. Formally, the main target is to minimize (3.7) to obtain the closest translational motion vector, $v_i$.

$$v_i = \min \left\{ \left( (p_{b,x} + (p_{f,x} - p_{b,x}) \cdot w - p_{s_i,x})^2 + (p_{b,y} + (p_{f,y} - p_{b,y}) \cdot w - p_{s_i,y})^2 \right) \right\}$$  \hspace{1cm} (3.7)

$$w = \frac{d_b}{d_b + d_f}$$  \hspace{1cm} (3.8)

In (3.7), $(p_{b,x}, p_{b,y})$ and $(p_{f,x}, p_{f,y})$ represent the two points that define the vector estimated in Step 2 above, $(p_{s_i,x}, p_{s_i,y})$ represents each SI block center, $w$ is a weight computed as in (3.8) and $d_b$ and $d_f$ represent the temporal distance between the SI frame and the backward and forward reference frames, respectively.
3.3.2 Bidirectional Translational Motion Estimation

Bidirectional motion estimation is performed between the two reference frames, considering as starting point the motion vector estimated with the previous technique for each SI frame block [24]. The objective is to refine the translational motion field, and eventually ‘correct’ any errors in the motion vector selection (Step 3 from Section 3.1). The weighted approach used for the backward motion estimation targets allows to obtain a more coherent motion vector field. As in [24], this technique preserves the linear trajectory of the translational motion vectors when it is considered the SI frame as reference. Thus, motion estimation is performed with both reference frames and the translational motion vector that goes from $X_a$ to $X_f$ crosses the center of the SI block in a linear trajectory. That means that every translational motion vector evaluated must intersect the center of the respective SI block. This technique was described in detail in Section 2.2.3.1 (Bidirectional motion estimation with linear trajectories). This module is applied twice, one for block sizes of $16 \times 16$, to obtain a good starting point, and another for block sizes of $8 \times 8$, to refine the $16 \times 16$ translational motion field and to characterize with enough detail the motion of the sequence and to obtain the same translational motion field as the MCFI [24].

3.3.3 Reference Frames Up-sampling

This tool up-samples both reference frames to quarter pixel accuracy with the objective of providing reference frames with an increased resolution. This allows to better estimate the warped blocks with the geometric transforms tools presented in the next sections. Since the true motion of the sequence cannot be fully described with the full pixel samples, up-sampled reference frames should allow to achieve a better precision for the estimation of the best geometric transform.

In this case, the 6-tap H.264/AVC up-sampling filter, nowadays used by the motion estimation and compensation tools in the H.264/AVC video codec to obtain motion vectors with 1/4 pixel precision, was selected (see Figure 3.3).

![Figure 3.3 – Half pixel and quarter pixel positions definition. Capital letters represent the integer pixels and lower case letters represent the half and quarter pixel positions [1].]
In Figure 3.3 and Figure 3.4, the capital letters, like $G, H, M, N$, are the integer pixel position, the positions $b, h, j, m, s, a, a, b, b, c, d, d, e, e, f, f, g, g$ and $h$ represent the half pixel positions and the $a, c, d, e, g, i, k, n, p$ and $r$ represent the quarter pixel positions. To obtain the up-sampled pixel values for 1/2 and 1/4 pixel positions, the following steps are performed:

1. The horizontal and vertical half pixel samples (orange pixels in Figure 3.4) are calculated with (3.9). For example, pixels $b$ and $h$ are calculated with:

$$b = \frac{E - 5F + 20G + 20H - 5I + J}{32} + 0.5 \quad (3.9)$$

$$h = \frac{A - 5B + 20G + 20M - 5R + T}{32} + 0.5$$

2. Then, the middle half pixel samples (red pixels in Figure 3.4) are calculated with (3.10). For example, the pixel $j$ is calculated with:

$$j = \frac{cc - 5dd + 20h + 20m - 5ee + ff}{32} + 0.5 \quad (3.10)$$

3. After, quarter pixel samples that are between half pixel samples, i.e. green pixels in Figure 3.4, are estimated with a bilinear filter. Thus, estimation is performed by averaging, followed by rounding the two closest half pixel samples, as shown in (3.11).

$$a = \frac{G + b}{2} + 0.5 \quad (3.11)$$

4. Finally, the diagonal quarter pixel samples, i.e. the black pixels in Figure 3.4, are calculated by averaging, followed by rounding the two closest diagonal half pixel samples, as in (3.12).

$$e = \frac{b + h}{2} + 0.5 \quad (3.12)$$

Figure 3.4 - (a) Full pixel and (b) Half and quarter pixel samples illustration.

### 3.3.4 Bidirectional Geometric Motion Estimation

The main objective of this tool is to estimate the best geometric transform to characterize the deformation of each SI block. This module is applied twice, first with a block size of $16 \times 16$ and after with a block size of $8 \times 8$; it receives as input the translational motion vectors for a block size of $16 \times 16$ and the GT vectors for a block size of $8 \times 8$. This hierarchical approach was adopted so that the $16 \times 16$ GT motion field can provide a good starting point for the final $8 \times 8$ GT motion field.

As this is a rather complex module, the detailed architecture of the proposed Bidirectional Geometric Motion Estimation module is shown in Figure 3.5. This architecture as two inputs and they
are used separately: when this algorithm is applied to $16 \times 16$ blocks, the input is the translational motion vectors field that will be converted into the corresponding GT vectors; when the algorithm is applied to $8 \times 8$ blocks, a GT motion fields was already obtained, and thus the initial vectors conversion is not needed. The output of this module is a refined GT motion field.

Figure 3.5 - Architecture of the proposed Bidirectional Geometric ME.

To estimate a geometric transform a possible approach is to evaluate every possible combination of GT vectors, for example with a $32 \times 32$ pixels search window, it would imply estimating, for each block, more than $10^{16}$ perspective transforms. Since the full-search search complexity is rather high, a novel GT search algorithm was defined to find the best GT vectors for each block, providing a good trade-off between SI quality and decoder complexity. The approach followed in the estimation of the best GT vector for each vertex of a given block in the backward reference frame is illustrated in Figure 3.6.

Figure 3.6 - (a) Initial GT vectors positions and (b) Possible best GT vector positions after performing the search for the first vertex (considering only the backward direction).

Note that, to preserve the linear trajectory of the GT vectors of each block vertex, the forward GT vector position needs to be symmetric to the backward GT vector position considering the $(u_i, v_i)$ position in the SI frame, as seen in Figure 3.7. With this constraint, it is guaranteed that each GT vector from $X_b$ to $X_f$ crosses its respective vertex $(u_i, v_i)$ in the interpolated SI frame.
To obtain the final set of four GT vectors for each SI block, the following techniques are applied according to the architecture in Figure 3.5:

1. **GT Vectors Creation** – To have an initial starting point for the estimation of the best geometric deformation, four translational GT vectors are required for each SI block. These four initial translational GT vectors correspond to a geometric deformation that is equivalent to the translation obtained with just one motion vector. Thus, to create the initial translational GT vectors, it is only necessary to displace the input motion vector into the four vertices of the block, as shown in Figure 3.8.

![Figure 3.8](image)

*Figure 3.8 – (a) Input translational motion vectors and (b) translational GT vectors.*

2. **Vertex Selection** – A block vertex is selected starting with the upper left vertex and following a clockwise rotation. The estimation of the best geometric transform is done by searching one GT vector at a time while the remaining GT vectors are kept fixed, as shown in Figure 3.6.

3. **GT Vectors Selection** – Then, the best GT vector for a given vertex should be selected. Initially, for a given search window, the GT vector of the selected vertex points to the positions \((x_{b,i}, y_{b,i})\) and \((x_{f,i}, y_{f,i})\) in the backward and forward reference frames, respectively, where \(i\) represents the respective vertex. The \((x_{b,i}, y_{b,i})\) and \((x_{f,i}, y_{f,i})\) correspond to the reference GT vector positions, i.e. the central positions where the search windows are located for the defined vertex \(i\). For each possible GT vector in the search window, the following operations are made:

   a. **GT Parameters Determination**: The GT parameters for both the backward and forward directions are calculated. Thus, using the four SI block vertices \((u_i, v_i)\) and the corresponding GT vector positions \((x_{b,i}, y_{b,i})\) in the backward reference frame, the GT parameters are calculated with (3.6) and (3.4). The same procedure is repeated for the GT vector positions \((x_{f,i}, y_{f,i})\) in the forward reference frame.
b. **Blocks Warping:** Then, two warped blocks are generated using the two sets of GT parameters calculated in the previous step, one for each up-sampled reference frame. This process is illustrated in Figure 3.9.

![Blocks warping scheme](image)

**Figure 3.9 – Blocks warping scheme.**

The mapping of a regular grid of pixels into a quadrilateral usually leads to points lying in non-integer positions. Since the \((x, y)\) coordinates calculated with (3.2) can have fractional precision, it is necessary to have a reliable method to estimate the pixel values for these fractional positions. The reference frames, which were up-sampled with the quarter pixel H.264/AVC interpolation filter, provide pixel samples at quarter pixel position. However, since any fractional position may be obtained, these quarter pixel samples are not enough to estimate fractional pixel values with arbitrary precision. Therefore, a bilinear interpolation method is used to estimate the pixel value at any position based on the quarter pixel samples. Formally, equation (3.13) corresponds to the solution presented in Figure 3.10, i.e. the equation expresses a combination of two linear interpolations, one in each direction. Then, the final equation (3.14) is obtained by manipulating (3.13).

\[
P(x, y) = (1 - p_x)(1 - p_y)p_a + p_x(1 - p_y)p_b + (1 - p_x)p_y p_d + p_xp_y p_c \quad (3.13)
\]

\[
P(x, y) = (p_b - p_d)p_x + (p_d - p_c)p_y + (p_c - p_d)p_b + p_d p_x p_y + p_c \quad (3.14)
\]

![Bilinear pixel interpolation for a regular square grid](image)

**Figure 3.10 – Bilinear pixel interpolation for a regular square grid [6].**

As seen in (3.13), (3.14) and Figure 3.10, the estimation for a given pixel value \(P(x, y)\) in position \((x, y)\) is based on the weights \(p_x\) and \(p_y\), representing the distances to the reference position in the square grid, \(p_a\), both horizontally and vertically.

c. **Residual Error Calculation:** The MAD metric is now applied to calculate the residual error between the two warped blocks (3.15), where \(N\) is the block size and \(W_b\) and \(W_f\) are the
warped blocks for the set of GT vectors corresponding to the geometric transform for the backward and forward reference frames.

\[
MAD = \frac{1}{N \times N} \sum_{i,j} |W_{f,i} - W_{b,j}|
\]

(3.15)

d. **GT Vectors Regularization:** To obtain a more accurate geometric transform estimation, it is important to find the true motion of the objects/blocks in the video sequence [24]. Thus, it is not enough to minimize the MAD residual error as many motion estimation solutions do but it is also needed to find a smoother GT motion field. In this case, a simple regularization criterion, favoring the GT vectors closer to the origin and avoiding GT vectors with large magnitudes (that most likely do not follow the true motion), is used. This GT vector regularization consists in applying a penalty to the MAD obtained in the previous step, directly proportional to the distance between the GT vector and the reference GT vector, i.e. the GT vector available as input to this module. In the first iteration of the GT vector search process, a *local weighted approach* where every vertex is considered independently is used. For the following iterations, a *global weighted approach* is applied, where the GT vectors obtained for all vertices are globally considered. This two-level approach allows to independently estimate the GT vectors (in the first iteration) without any constraint, i.e. the penalty applied to the MAD is directly proportional to the distance between the GT vector and the reference GT vector of that vertex. To calculate the MAD value for the local weighted approach \((MAD_l)\) the following equations are used:

\[
d_l = \sqrt{(x - x_c)^2 + (y - y_c)^2}
\]

(3.16)

\[
MAD_l = MAD \times (1 + k \times d_l)
\]

(3.17)

where \((x_c,y_c)\) represents the reference GT vector position for a given vertex, \(d_l\) represents the distance between the GT vector position \((x,y)\) and \((x_c,y_c)\) for the local weighted approach, and \(k\) is a scaling factor.

In the following iterations, the GT vectors are globally refined, i.e. the penalty applied to the MAD is directly proportional to the sum of the distances between the GT positions and the corresponding reference GT positions for all vertices. The equations to calculate the MAD value for the global weighted approach \((MAD_g)\) are shown in (3.18) and (3.19):

\[
d_g = \sum_{i=1}^{4} d_{l_i}
\]

(3.18)

\[
MAD_g = MAD \times (1 + k \times \frac{d_g}{4})
\]

(3.19)

where \(d_g\) represents the sum of the distances of the GT vectors to \((x_c,y_c)\) calculated for each vertex independently. It was found experimentally that \(k\) could be the same for both the global and local approaches, since no benefits were obtained with different \(k\) values for the approach used. Then, steps a to d are repeated until all positions inside the search range are tested.

e. **GT Vector Decision:** From all the GT vectors evaluated inside the search window, the GT vector leading to the minimum MAD is selected and kept fixed when performing these steps.
for the other vertices. Then, the next vertex is selected and all techniques from step a are repeated, until all the four vertices are evaluated.

When the four vertices have been processed for a SI block and the corresponding GT vectors are obtained, a full refinement iteration is completed, and the algorithm proceeds to Step 4.

4. **Refinement Stopping Criterion** – If the GT vectors do not change during a complete refinement iteration, the algorithm stops the search process, because the search algorithm has successfully converged to a solution (i.e. the best GT vectors have been found); in most cases, the computational complexity can be reduced when compared to an approach where a fixed number of iterations are executed. Otherwise, the number of iterations is incremented and the algorithm goes back to Step 2. To stop the GT search algorithm for the cases where convergence is difficult to obtain, the maximum number of iterations is limited to five.

### 3.3.5 Block Size Adaptation

The main objective of this module is to obtain more precise GT vectors (finer scale) based on GT vectors already calculated for a coarser scale. Here, the geometric deformation is calculated using the process explained in Section 3.2. Thus, the GT vectors for the four $8 \times 8$ blocks in each $16 \times 16$ block are calculated by applying the geometric deformation of the respective $16 \times 16$ block. This procedure is done for both backward and forward GT vectors positions. The final result for one of the directions is shown in Figure 3.11.

![Figure 3.11](image)

**Figure 3.11** – Example of the block size adaptation for one direction: (a) GT vectors positions for the larger block and (b) obtained GT vectors positions for the four smaller blocks.

### 3.3.6 Motion Model Decision

The motion model decision process selects, for each SI block, between the translational and the GT motion modeling modes using a MAD criterion, i.e. the mode with the minimum MAD residual error value for a given SI block is chosen. The residual error for the translational mode was already obtained in the Bidirectional Translational ME module for $8 \times 8$ blocks and intends to replicate MCFI [24], while the MAD residual error for the GT mode was calculated in the Step c of the Bidirectional Geometric ME module for $8 \times 8$ blocks. Regarding this decision, it is possible to conclude that, for a significant number of blocks, the GT mode has a slightly better MAD value than the translational mode but, in fact, when the geometric transform SI estimation is compared to the original block, a worse SI.
estimation may be obtained. Based on these observations, it is proposed to apply a penalty offset to the GT mode to ensure it is only selected when is significantly better than the translational mode. For each block, the decision $D_{si}$ follows (3.20):

$$D_{si} = \begin{cases} B_w, & \text{if } \text{MAD}_w < \text{MAD}_t - \alpha \\ B_t, & \text{if } \text{MAD}_w \geq \text{MAD}_t - \alpha \end{cases}$$

(3.20)

where $\alpha$ is the penalty offset, $D_{si}$ represents the chosen mode for each SI block and $B_w$ and $B_t$ are the SI estimations for the GT and translational modes, respectively.

### 3.3.7 Motion Compensation

Finally, bidirectional motion compensation is performed to obtain the side information frame. This module uses the calculated GT parameters, the bilinear interpolation method and the up-sampled reference frames to obtain the warped pixel values for each SI block depending on the selection made in the previous module. The pixel values are obtained by averaging, followed by rounding, of the pixels values indicated in both reference frames, as in (3.21):

$$P_{si} = \left\lfloor P_f \ast w + P_b \ast (1 - w) + 0.5 \right\rfloor$$

(3.21)

where $P_b$ and $P_f$ are the pixel values in the backward and forward frames, $w$ is the weight defined in (3.8) and $\lfloor \cdot \rfloor$ is the floor operator.

### 3.4 Summary

The main idea of the proposed Unidirectional Warped Side Information creation framework is to exploit a geometric motion model to obtain side information with better quality than with a typical translation motion model. The proposed UWSI solution makes two estimations for each SI block, a default one using the usual translational MCFI method [24] and a novel one using a geometric transform approach; then, for each block, the best approach is selected and the SI frame is created by appropriately combining the two motion models for all blocks. With respect to the geometric approach, several tools were proposed to obtain a more coherent GT motion field and avoid random deviations.

Since the UWSI solution has some weaknesses, notably regarding some SI frame regions that might not appear in both reference frames, another side information creation solution is proposed in Chapter 4. This solution also includes two branches: the MCFI and the GT motion model. The GT motion model is exploited by estimating geometric transforms into two possible directions (from forward to backward reference frames and vice versa) and a unidirectional motion compensation mode (only one reference frame is used). So, in the next Chapter, the Bidirectional Warped Side Information Creation solution is proposed in detail.
Chapter 4

Bidirectional Warping Side Information Creation: Architecture and Tools

In the previous chapter, a side information creation technique using an advanced motion model based on geometric transforms to characterize the true motion of the visual scene has been proposed. However, the unidirectional warping approach followed in the previous UWSI framework has some drawbacks, notably side information cannot be reliably estimated when occlusions occur. In fact, for some camera motions, like panning, some regions of one reference frame (such as frame borders) are occluded in the other reference frame; thus, the side information estimated with the proposed UWSI module has lower quality. To solve this problem, it is not enough to estimate a motion field from one reference frame to the other but it is rather necessary to estimate two motion fields, one in each direction, i.e. a bidirectional approach is needed. Thus, in this chapter, a new side information creation technique is proposed: the Bidirectional Warping Side Information (BWSI) creation solution.

This BWSI solution performs backward and forward motion estimation and geometric motion search for the reference frames, $X_b$ and $X_f$, to create the side information, with $b = i - N$ and $f = i + N$. In the backward direction, a motion model is estimated from the future reference frame ($i + N$) to the past reference frame ($i - N$) while, in the forward direction, a motion model is estimated from the past reference frame ($i - N$) to the future reference frame ($i + N$). These two motion models provide different estimations of the motion between the frames $X_b$ and $X_f$; thus, they must be fused together to obtain a unified GT motion field leading to better SI quality and better RD performance. In addition, this solution allows the use of a unidirectional motion compensation technique for some specific SI blocks, where only one reference frame is used to create the side information. This is similar to an extrapolation SI creation technique (or the traditional P mode in predictive video coding), but in this case allowing to estimate a SI block, not only from the past reference frame, $X_b$, but also from the future reference frame, $X_f$. Also, to obtain a more reliable GT motion field, outlier GT vectors are eliminated, which leads to a more coherent GT motion field and a better overall SI quality.
In Section 4.1, the architecture and the walkthrough of the proposed BWSI creation method is presented, while in Section 4.2 all the procedures and tools proposed are described. In Section 4.3, the final remarks are presented.

4.1 BWSI Creation Framework

The proposed BWSI solution estimates, for each SI block, two geometric transforms that warp regions from the reference frames to obtain an improved SI estimation. As in the previous UWSI solution, motion estimation is performed to obtain an initial translational transform (the starting point for the estimation of the geometric transform) and two block sizes are used: 16 x 16 and 8 x 8. The architecture of the proposed BWSI solution is shown in Figure 4.1, where the modules in blue correspond to the modules proposed in the context of this Thesis which are different from the modules already proposed for the previous UWSI solution presented in Chapter 3.

**Figure 4.1 - Architecture of the proposed BWSI creation solution.**

The walkthrough of the proposed BWSI method is explained in the following:

1. **Reference Frames Up-sampling** – Initially, both reference frames are up-sampled by a factor of 4 to provide better estimations for all possible geometric transforms, i.e. warped blocks with higher quality. The up-sampling factor is 4 to obtain frames with quarter-pel accuracy. This technique was already used for the UWSI solution and is described in Section 3.3.3.

2. **Forward ME** – Using both reference frames \(X_b\) and \(X_f\), motion estimation with 16 x 16 block sizes and full pixel accuracy is performed for the forward direction, i.e. from the past reference frame \(i - N\) to the future reference frame \(i + N\). This process generates a forward translational motion field and provides a good starting point for the forward geometric search performed in Step 4. This BWSI module is similar to the corresponding UWSI module described in Section 3.3.1; however, in this case, the selected motion vector for each SI block is not the translational motion vector that is closest to its center. Thus, this module provides the forward translational motion field, \(MV_{b,f}\), characterizing the motion from the reference frame \(X_b\) to the reference frame \(X_f\).

3. **Backward ME** – Here, Step 2 is repeated for the backward direction, i.e. estimating the motion from the future reference frame \(i + N\) to the past reference frame \(i - N\). Thus, the backward translational motion field, \(MV_{f,b}\), characterizing the motion from the reference frame \(X_f\) to the reference frame \(X_b\), is obtained to provide the starting point for the backward geometric search to be performed in Step 5.

4. **Forward Geometric ME** – This module receives as input the forward translational motion field \(MV_{b,f}\) estimated in Step 2 and starts by generating a set of forward GT vectors for each \(X_b\) block. Then, the estimation of the best geometric transform for each block is performed with a weighted
matching criterion using the up-sampled reference frames; in this case, a block size of $16 \times 16$ and half-pel accuracy is used for the geometric transform vectors.

5. **Backward Geometric ME** – Then, the process performed in Step 4 is repeated in the opposite direction (from the future to the past), thus obtaining a backward GT motion field.

6. **GT Vectors Fusion** – This step aims at obtaining a reliable GT motion field for the SI frame is created while avoiding holes and block overlappings when the SI frame is created. This module receives as input both the forward and backward GT motion fields and generates a single GT motion field for the SI frame, along with the motion compensation mode that should be used. Initially, the GT vectors considered unreliable are eliminated. Then, for each SI block is selected one of the transforms previously calculated (in steps 4 and 5). Finally, to better estimate the regions where occlusions occur, the best motion compensation mode (and respective GT field) for each SI block is chosen.

7. **Bidirectional Geometric ME** – To refine the GT vectors obtained in the previous step, bidirectional geometric ME is performed. However, only the blocks previously classified with the bidirectional motion compensation mode are refined. The GT vectors for the remaining blocks (using a unidirectional motion compensation mode) do not change. This BWSI module is the same as the UWSI module described in Section 3.3.4.

8. **Block Size Adaptation** – This module intends to create a GT motion field for $8 \times 8$ blocks using as input the GT motion field obtained in Step 7 for $16 \times 16$ blocks. A hierarchical approach was adopted because it has been experimentally proved that obtaining a $8 \times 8$ GT motion field refined from a $16 \times 16$ GT motion field brings more coherent results that directly estimating the motion field with $8 \times 8$ blocks. After generating this GT motion field for $8 \times 8$ blocks, the motion compensation mode of each block is reevaluated. Then, it is possible to refine the GT vectors with smaller block sizes by performing bidirectional geometric ME again (i.e. repeating Step 7 for block sizes of $8 \times 8$). This operation is performed only for blocks that were classified with the bidirectional motion compensation mode.

9. **Bidirectional ME ($16 \times 16$ and $8 \times 8$)** – Besides the block-level warping tools presented in steps 2 to 8, bidirectional ME is performed twice, the first time with a block size of $16 \times 16$ and then with a block size of $8 \times 8$, to replicate the MCFI approach [24]. By performing also this pure translational approach, it is possible to perform next a motion model decision (Step 10) that chooses between the conventional MCFI approach and the novel geometric transform approach. This module is described in Section 3.3.2.

10. **Motion Model Decision** – This step aims to choose the best motion model. To do so, it uses the MAD values calculated for the warping mode in Step 8 and for the MCFI in Step 9. Then, the best motion model is chosen with a residual error criterion. The blocks that were classified with the unidirectional mode are not submitted to this motion model decision.

11. **Motion Compensation** – Finally, motion compensation is performed to obtain the final SI frame. Depending on the motion compensation mode selected in Steps 6 and 8, a SI block might be compensated using the bidirectional mode as described in Section 3.3.7 (two reference frames are used), or the unidirectional approach (one reference frame is used).
4.2 BWSI Creation Tools

In this section, the tools specifically proposed for the BWSI solution are described in detail. Since the Reference Frames Up-sampling, Forward ME, Bidirectional geometric ME, Block Size Adaptation and Motion Model Decision modules were already described in Section 3.3, their description is not repeated here.

4.2.1 Backward Geometric ME

The main target of this module is to model the motion with geometric transforms using as starting point the translational motion vectors obtained with the previous Backward ME module. Since this technique is the same for the backward and forward directions, only the backward geometric motion estimation is described. This technique is similar to the Bidirectional Geometric ME proposed for the previous UWSI solution with the difference that now the estimation process is performed between $X_b$ and $X_f$, i.e. without the SI frame. The backward geometric ME consists in the following steps:

1. **GT Vectors Creation** – In this step, a translational geometric transform (a set of four GT vectors for each block) is created with the received translational motion vectors as previously explained in Section 3.3.4.

2. **Vertex Selection** – A block vertex is selected starting with the upper left vertex and following a clockwise direction. As in the previous solution, the estimation of the best geometric transform is done by searching one GT vector at a time while fixing the other GT vectors.

3. **GT Vectors Selection** – In this step, the geometric transform of each block is found by evaluating several possible deformations, i.e. the quality of several warped blocks. For each block, the search for the best GT vectors is made with the following techniques:

   a. **GT Parameters Determination**: After selecting the vertex and setting each initial search point within the search window, the GT parameters for the quadrilateral deformation of the backward reference frame are obtained considering each block on the forward reference frame. This process is quite similar to the GT parameters determination described in the previous UWSI solution, although in this case only a set of GT parameters representing the motion between the reference frames $X_b$ and $X_f$ is obtained. In the previous UWSI solution, two sets of GT parameters were obtained to represent the motion from $Y_i$ to $X_b$ and $Y_i$ to $X_f$.

   b. **Block Warping**: In this step, one warped block is generated with the parameters calculated in Step a. This process is illustrated in Figure 4.2. The procedure for block warping is described in Section 3.3.4.

![Figure 4.2 – Backward geometric motion estimation model.](image-url)
c. **Residual Error Calculation:** To evaluate each warped block, the same MAD metric used for the USWI solution is adopted but now with different reference frames. In this case, the residual error calculation is performed between the warped block in frame $X_b$ obtained in Step b and the current block in frame $X_f$.

d. **GT Vectors Regularization:** To regularize the GT vectors, the two-level approach previously proposed for the UWSI solution is used, notably the same regularization criterion is employed, i.e. the local weighted approach in the first iteration, and the global weighted approach in the remaining iterations. Then, the algorithm goes back to Step a to repeat this process until all positions inside the search range are tested.

e. **GT Vectors Decision:** The best warped block is selected from the set of all warped blocks evaluated; this block corresponds to the block leading to the minimum MAD value. The set of GT vectors leading to the best warped block are stored as well as the residual error, as done for the previous UWSI solution. Then, Steps 2 and 3 are repeated for the each of the other three block vertices, using a clockwise rotation.

4. **Refinement Stopping Criteria** – When all the four vertices are tested, a full refinement iteration is completed. As for the previous solution, if the GT vectors do not change during an iteration, the algorithm stops the GT vectors search process; otherwise, another iteration is performed, until a maximum of five iterations.

4.2.2 **GT Vectors Fusion**

The main objective of this module is to provide a reliable GT motion field for the SI frame creation using the GT motion fields previously estimated [25], this means for both the forward and backward directions. This process includes three main steps: initially, the transforms $T'$ considered unreliable are eliminated; then, the two closest transforms $T'$ to each SI block are tested and the best one is selected, with $T'$ being a group of four GT vectors, one for each vertex, representing the deformation of a given block; finally, the motion compensation mode is chosen for each SI block. This module proceeds as follows:

1. **GT Vectors Elimination** – This step compares the two residual errors obtained when the forward transform $T_f$ and the backward transform $T_b$ obtained for the same block position, but in their respective reference frames. This comparison is performed to decide if both GT vectors are kept or one of them is excluded [25]. The decision $D_i$, for each block, is performed as follows:

$$
D_i = \begin{cases} 
0, & \text{if } |MAD_b - MAD_f| < \tau \\
1, & \text{if } |MAD_b - MAD_f| \geq \tau \land MAD_b < MAD_f \\
2, & \text{if } |MAD_b - MAD_f| \geq \tau \land MAD_f < MAD_b 
\end{cases}
$$

(4.1)

where $MAD_b$ is the residual error calculated between a given block in $X_f$ and the respective warped block obtained by the backward transform $T_b$ (the same for the $MAD_f$), and $\tau$ is a threshold. If $D_i = 0$, both the backward transform, $T_b$, and the forward transform, $T_f$, are considered reliable; thus, both are kept. When $|MAD_b - MAD_f|$ is larger than $\tau$, that means that one of the transforms is considered more reliable than the other. So, if $D_i = 1$, $T_b$ is kept ($MAD_b$...
has the lowest value) and \( T_f \) is dropped, while if \( D_i = 2 \), \( T_f \) is kept (\( MAD_f \) has the lowest value) and \( T_b \) is dropped.

2. **GT Motion Field Creation** – Then, the best \( T \) for each SI block is selected between the \( T_b \) and \( T_f \) considered reliable in the previous step, i.e. the \( T_b \) and \( T_f \) transforms that were not eliminated. For this selection, a simple criterion based on the distance between the position where each GT vector intersects the SI frame and the corresponding vertex of each SI block is used. Thus, the two closest \( T \), one for each direction, are chosen and evaluated, and the best one is selected. To calculated the distance of every transform of the backward GT motion field to a given SI block, the following steps are repeated for all the \( T_b \) transforms:

a. For each GT vector of a given transform \( T_b \), the distance \( d_i \) between the vertex and the point where the GT vector intersects the SI frame (see Figure 4.3) is calculated as:

\[
\begin{align*}
    d_i = & \sqrt{(x_{b,i} + (x_{f,i} - x_{b,i}) \ast w - u_i)^2 + (y_{b,i} + (y_{f,i} - y_{b,i}) \ast w - v_i)^2} \\
\end{align*}
\]

In (4.2), \((x_{b,i}, y_{b,i})\) and \((x_{f,i}, y_{f,i})\) are the GT vector positions of vertex \( i \) in \( X_b \) and \( X_f \), respectively, \( w \) is the weight already defined in (3.8) and \((u_i, v_i)\) are the vertices of the SI block under consideration.

b. Then, the overall distance \( d_{T_b} \) is calculated for \( T_b \) using (4.3), to define the overall distance between the position where the transform \( T_b \) intersects the SI frame and the corresponding SI block (for which no geometric transform is available).

\[
\begin{align*}
    d_{T_b} = & \sum_{i=1}^{4} d_i \\
\end{align*}
\]

c. After obtaining the overall distance \( d_{T_b} \) for each \( T_b \) transform, the \( T_b \) that leads to the minimum overall distance \( d_{T_b} \) is selected.

Then, the previous steps a to c are repeated for the forward direction, to find the closest transform \( T_f \) to the SI block. Finally, the GT vectors of the two closest transforms are displaced, so that the GT vectors can cross the respective vertex on the SI block. After obtaining the warped blocks, the residual error is calculated for both transforms \( T_b \) and \( T_f \), and the one leading to the minimum MAD value is selected. From the transform selected between the reference frames, it is now...
possible to obtain two transforms, $T_{si,b}$ and $T_{si,f}$, as seen in Figure 4.4, that represent the motion between the SI frame and the corresponding reference frame $X_b$ and $X_f$.

![Graphical representation of the two transforms $T_{si,b}$ and $T_{si,f}$ obtained for an SI block.](image)

**Figure 4.4** – Graphical representation of the two transforms $T_{si,b}$ and $T_{si,f}$ obtained for an SI block.

3. **Motion Compensation Mode Decision** – Since some regions of the image might not appear in both reference frames, three motion compensation modes are needed. This procedure intends to classify every $16 \times 16$ SI block in terms of the best motion compensation mode: interpolation with two reference frames or extrapolation with just one reference frame (from the past or the future reference frames). As shown in Figure 4.5, the use of the traditional bidirectional motion compensation does not perform well when occlusions are present even if padding of the pixels in frame boundaries is performed [25].

![An example of a typical occlusion](image)

**Figure 4.5** – An example of a typical occlusion [25].

In some cases, the bidirectional approach followed in the BWSI wrongly assumes that some objects or regions of the frame are available in both reference frames, when they might appear only in one of the reference frames, just in the past or in the future. Thus, two unidirectional motion compensation modes have been introduced to allow the creation of a SI block using only one reference frame. The three motion compensation modes defined are: bidirectional, backward unidirectional and forward unidirectional. The bidirectional mode is the one presented in Section
3.3.7, where a given SI block is compensated using both reference frames. In the backward and forward unidirectional mode, the blocks are compensated using only the backward and forward reference frames, respectively.

The decision of the best motion compensation mode for each SI block is based on the percentage of pixels of the warped blocks laying outside the reference frame boundaries. Thus, to decide if a SI block is classified as unidirectional or bidirectional, both backward and forward deformations obtained by the GT vectors are evaluated, to evaluate how much the quadrilaterals obtained are inside (or outside) the corresponding reference frame. So, the decision for the flag $D_b$, which is rather 0 or 1, is performed as follows (for the forward transform $T_{si,f}$, the decision for the flag $D_f$ process the same way):

$$D_b = \begin{cases} 0, & \text{if } P_b > \rho \\ 1, & \text{otherwise} \end{cases}$$

(4.4)

where $P_b$ is the percentage of the pixels obtained from the $T_{si,b}$ transform laying outside $X_b$ and $\rho$ is a fixed percentage threshold. If $D_b = 0$, $P_b$ is larger than $\rho$, so $T_{si,b}$ is considered to be outside $X_b$; otherwise, if $D_b = 1$, $T_{si,b}$ is considered to be outside $X_b$. After deciding if $T_{si,b}$ and $T_{si,f}$ are inside or outside the respective frame, the motion compensation mode $D_{si}$ can be selected for each SI block as:

$$D_{si} = \begin{cases} U_b, & \text{if } D_b = 1 \land D_f = 0 \\ U_f, & \text{if } D_b = 0 \land D_f = 1 \\ B, & \text{otherwise} \end{cases}$$

(4.5)

From (4.5), it is possible to decide which motion compensation mode should be used for each SI block. So, if $T_{si,b}$ is considered to be inside $X_b$ ($D_b = 1$) and $T_{si,f}$ is considered to be outside $X_f$ ($D_f = 0$), the forward unidirectional compensation mode $U_f$ is chosen; however, if $D_b = 0$ and $D_f = 1$, the backward unidirectional compensation mode $U_b$ is chosen instead. If both $T_{si,b}$ and $T_{si,f}$ are considered to be inside $X_b$ and $X_f$ ($D_b = 1 \land D_f = 1$), the traditional bidirectional compensation mode $B$ is used. In the rare case, of both transforms considered outside ($D_b = 0 \land D_f = 0$), the bidirectional compensation mode $B$ is also used. The typical scenario for both backward and forward unidirectional compensation modes is presented in Figure 4.6, where two GT vectors of a SI block are represented.

![Figure 4.6 – Typical scenario for (a) backward unidirectional and (b) forward unidirectional compensation modes.](image-url)
4.2.3 Motion Compensation

This module aims to create the final SI frame with the estimated GT motion field and the motion compensation mode decisions obtained in the previous steps. In the previous UWSI solution, the motion compensation module only performed bidirectional motion compensation. However, in this BWSI solution, two additional motion compensation modes are used. Thus, if a given SI block is classified with the bidirectional mode, the motion compensation described in Section 3.3.7 for the USWI solution is performed. In case a SI block is classified with the backward unidirectional mode, the GT vectors of the backward reference frame are used to obtain the warped pixel values corresponding to the created SI block. For the forward unidirectional mode, the same procedure is performed, but with the corresponding GT vectors and the forward reference frame. An example of backward unidirectional compensation is presented in Figure 4.7.

![Figure 4.7 – Backward unidirectional mode compensation example.](image)

4.3 Summary

To summarize, the main idea of this proposed BWSI creation solution is to better estimate the final motion model based on a geometric motion transform and a bidirectional search approach. With the BWSI creation framework, several tools were proposed, notably the GT vectors elimination, to exclude transforms that are considered unreliable, and the unidirectional motion compensation mode, to allow the possibility of performing motion compensation using only one reference frame. By combining the new tools with some of the tools proposed in the previous UWSI creation solution, it is expected that further gains can be obtained towards outperforming the MCFI translational approach proposed in [24] and the UWSI creation solution proposed in Chapter 3. In the next chapter, the performance of both the UWSI and BWSI solutions will be assessed for meaningful test conditions, notably in terms of SI quality and overall RD performance.
Chapter 5

Warping Side Information Creation Methods: Performance Evaluation

The main target of this chapter is to present the performance evaluation of the UWSI and BWSI solutions proposed in the previous chapters. To obtain a solid assessment, precise and relevant test conditions are first defined in Section 5.1. After, results will be presented using the most relevant performance metrics to evaluate the proposed SI creation solutions, notably the quality of the side information and the final RD performance. Thus, in Section 5.2, the two proposed solutions are compared among them and with the state-of-the-art MCFI SI creation solution in terms of SI quality. To assess the RD performance of the proposed SI creation solutions, they were integrated in a state-of-the-art DVC codec adopting the Stanford architecture described in Chapter 2. In this way, two DVC codecs were obtained, notably DVC-UWSI and DVC-BWSI, which include the UWSI and BWSI SI creation solutions to generate the side information, respectively. This chapter is organized as follows: in Section 5.1, the test conditions, metrics and benchmarks are presented. In Sections 5.2 and 5.3, the SI quality and the RD performance are evaluated in detail. Finally, Section 5.4 provides some final remarks.

5.1 Test Conditions

To evaluate the SI quality and the overall RD performance, precise test conditions must be first defined. The DISCOVER DVC codec evaluation [16] provides a detailed, clear and complete set of test conditions that are currently widely used in the literature; thus, the test conditions used for the evaluation of the proposed SI creation methods are rather similar to the DISCOVER DVC codec test conditions; the differences regarding the DISCOVER test conditions will be highlighted in the following.

In the following subsections, the test conditions adopted to obtain the experimental results presented in Sections 5.2 and 5.3 are described. Thus, video sequences, coding conditions, coding benchmarks and evaluation metrics are presented next in detail.
5.1.1 Video Sequences

To evaluate the two proposed DVC codecs, nine video sequences were selected, with different characteristics, notably in terms of motion and texture, thus leading to a rather complete and meaningful set of sequences and results. The selected video sequences are: Container, Hall Monitor, Foreman, Mobile and Calendar, Coastguard, Table Tennis, Bus, Stefan and Soccer. In Figure 5.1, a sample frame of each sequence is presented.

![Video sequences selected: (a) first frame of Container; (b)frame 63 of Hall Monitor; (c) first frame of Foreman; (d) first frame of Mobile and Calendar; (e) frame 50 of Coastguard; (f) frame 99 of Table Tennis; (g) frame 29 of Bus; (h) first frame of Stefan; (i) first frame of Soccer.](image)

The Foreman, Hall Monitor, Soccer and Coastguard video sequences were already selected in the DISCOVER test conditions. However, more sequences were included to obtain a more representative set of results. The video sequences can be divided into 3 categories, notably low, medium and high motion activity, as explained in the following:

- **Low motion activity** – The Container and Hall Monitor video sequences can be classified as low motion sequences, mainly due to the static camera and low object motion. Both sequences were acquired with a fixed surveillance camera. The Container video sequence shows a scene at the harbor with a container cargo ship moving slowly, while the Hall Monitor video sequence is a typical video surveillance sequence showing an office corridor, with two persons walking.

- **Medium motion activity** – The Foreman, Mobile and Calendar, Coastguard and Table Tennis video sequences can be classified as medium motion sequences. The well-known Foreman videotelephony sequence shows a worker talking on a mobile phone, then moving the camera to show a building under construction. The Mobile and Calendar shows a toy train moving, while the background is composed by highly detailed drawings and a calendar moving up and down. The Coastguard video sequence is composed by a pan-left, following one small boat, and then a pan-right following a bigger boat. The Table Tennis sequence shows a match between two table tennis players and contains, initially, a zoom out followed by a scene cut, before the camera finally becomes static.

- **High motion activity** – The Bus, Stefan and Soccer video sequences can be classified as high motion sequences. The Bus sequence contains a fast panning following a big bus, while the Stefan sequence contains fast panning camera motion to follow a tennis player in a tennis court.
The Soccer sequence shows people playing football, while the camera attempts to follow the ball; the camera motion and object (players) motion is rather high for this sequence.

In Table 5.1, the characteristics of each video sequence are presented, notably the spatial and temporal resolutions as well the number of frames for each sequence.

Table 5.1 – Test video sequences characteristics.

<table>
<thead>
<tr>
<th>Motion Activity</th>
<th>Video Sequence</th>
<th>Spatial Resolution</th>
<th>Temporal Resolution [Hz]</th>
<th>Number of frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Container</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>Hall Monitor</td>
<td>176 x 144</td>
<td>15</td>
<td>165</td>
</tr>
<tr>
<td>Medium</td>
<td>Foreman</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>Mobile and Calendar</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>Coastguard</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>Table Tennis</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td>High</td>
<td>Bus</td>
<td>176 x 144</td>
<td>15</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>Stefan</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>Soccer</td>
<td>176 x 144</td>
<td>15</td>
<td>149</td>
</tr>
</tbody>
</table>

5.1.2 Coding Conditions

In this section, all the coding parameters and configurations that were used to evaluate the DVC-UWSI and DVC-BWSI video codecs performance are presented. The following conditions were adopted:

- **GOP Size** – For the evaluation of the proposed solutions, a fixed GOP size of 2 was adopted for all sequences, since this is the GOP size most often selected in the DVC literature.

- **Keyframe Coding** – The key frames are always encoded with H.264/AVC Intra in the Main profile [26] since this is one of the best Intra coding schemes in terms of RD performance; the H.264/AVC reference software JM9.6 with RD optimization on and all Intra modes enabled has been used.

- **Quantization Parameters** – To perform the experimental evaluation, eight RD points ($Q_i$) were defined in terms of the H.264/AVC Intra key frames quantization parameter ($QP_i$) and quantization matrix for the WZ frames. The WZ frames quantization matrices define for which DCT bands parity bits are transmitted and the number of bitplanes encoded for each DCT band; the decoded quality (after reconstruction) depends on the quantization matrix chosen, i.e. the amount of errors that are corrected, and the corresponding key frame quantization step. The eight RD points are presented in Figure 5.2 by defining the number of bitplanes coded for each DCT band.
The QP values were defined using an iterative process, which stopped when the average WZ frames quality (PSNR) was similar to the average key frames quality to avoid significant temporal quality variations which may have a negative user impact. The RD point Q1 corresponds to the lower bitrate/quality point, and the bitrate/quality increases as higher RD points are selected. Note that the QPs for the Hall Monitor, Foreman, Coastguard and Soccer video sequences were obtained from the DISCOVER test conditions that used the same QP definition process [16]. In Table 5.2, the QP values used for each RD point of each video sequence are presented.

**Table 5.2 - Key frames quantization parameters for each RD point, Q1.**

<table>
<thead>
<tr>
<th>Video sequences</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container</td>
<td>27</td>
<td>27</td>
<td>26</td>
<td>25</td>
<td>25</td>
<td>24</td>
<td>23</td>
<td>21</td>
</tr>
<tr>
<td>Hall Monitor</td>
<td>37</td>
<td>36</td>
<td>36</td>
<td>33</td>
<td>33</td>
<td>31</td>
<td>29</td>
<td>24</td>
</tr>
<tr>
<td>Foreman</td>
<td>40</td>
<td>39</td>
<td>38</td>
<td>34</td>
<td>34</td>
<td>32</td>
<td>29</td>
<td>25</td>
</tr>
<tr>
<td>Mobile and Calendar</td>
<td>38</td>
<td>37</td>
<td>36</td>
<td>35</td>
<td>35</td>
<td>34</td>
<td>32</td>
<td>28</td>
</tr>
<tr>
<td>Coastguard</td>
<td>38</td>
<td>37</td>
<td>37</td>
<td>34</td>
<td>33</td>
<td>31</td>
<td>30</td>
<td>26</td>
</tr>
<tr>
<td>Table Tennis</td>
<td>43</td>
<td>42</td>
<td>41</td>
<td>38</td>
<td>37</td>
<td>35</td>
<td>33</td>
<td>29</td>
</tr>
<tr>
<td>Bus</td>
<td>45</td>
<td>44</td>
<td>43</td>
<td>39</td>
<td>39</td>
<td>37</td>
<td>34</td>
<td>30</td>
</tr>
<tr>
<td>Stefan</td>
<td>45</td>
<td>44</td>
<td>44</td>
<td>40</td>
<td>39</td>
<td>38</td>
<td>35</td>
<td>30</td>
</tr>
<tr>
<td>Soccer</td>
<td>44</td>
<td>43</td>
<td>41</td>
<td>36</td>
<td>36</td>
<td>34</td>
<td>31</td>
<td>25</td>
</tr>
</tbody>
</table>

Note that the quantization parameters adopted for the H.264/AVC video codecs providing the key frames for the SI creation correspond to the RD points Q2 and Q7.

- **MCFI Parameters** – In the Forward ME, Backward ME and Bidirectional Translational ME modules, the parameters are chosen according to [24].

- **GT Estimation Parameters** – As explained before, to perform the estimation of the geometric transform, a hierarchical approach is adopted with two block sizes: first a 16 × 16 block size and, after, a 8 × 8 block size. The geometric modeling of the 16 × 16 blocks is performed with GT
vectors with half pixel accuracy, while the 8 × 8 blocks use GT vectors with quarter pixel accuracy. The Bidirectional Geometric ME included in both the USWI and BWSI solutions uses a 7 × 7 search window for each vertex of the 16 × 16 blocks, while for the 8 × 8 blocks use a 5 × 5 search window. In the BWSI solution, the Forward (or Backward) Geometric ME is performed with a 9 × 9 search windows. The search range values were empirically selected to obtain the best SI performance.

- **GT Regularization Parameters** – To regularize the GT motion field, both solutions use a local and global weighted approach including a scaling factor. For the Bidirectional Geometric ME included in the UWSI solution, the scaling factor for the 16 × 16 blocks, \( k_{b,16\times16} \), is 0.04, while for the 8 × 8 blocks, \( k_{b,8\times8} \), it is 0.21. For the BWSI solution, \( k_{b,16\times16} \) is 0.05 and \( k_{b,8\times8} \) is 0.21. Since the BWSI solution includes the Forward/Backward Geometric ME, another scaling factor, \( k_u \), taking the value 0.05 was defined. Again, all the scaling factor values were empirically selected to obtain the best SI performance.

- **Motion Model Decision Parameter** – To perform the motion model decision described in Section 3.3.6, a penalty offset, \( \alpha \), was applied to the MAD value obtained with the GT motion model. Several possible values were evaluated and it was concluded that \( \alpha \) equal 1 would lead to the best SI creation performance for both solutions.

- **GT Vectors Fusion Parameters** – This module described in Section 4.2.2 defines two thresholds: \( \tau \) and \( \rho \). The first threshold decides, in terms of MAD value, if two blocks are considered reliable or not, while the second threshold defines a percentage of pixels of a certain transform \( T \) that are allowed to be outside the frame. If the percentage of pixels outside the frame is greater than \( \rho \), the transform is considered to be outside the frame. These parameters are only used in the BWSI solution. After testing several possibilities, \( \tau \) was made equal to 4 and \( \rho \) equal to 25%.

- **Correlation noise model** – Besides the parameters necessary for SI creation, other parameters are necessary to perform DVC decoding. The correlation noise model corresponds to a Laplacian model where the Laplace parameter was offline defined for each band [27]. This unusual (and unrealistic) choice requires the transmission to the decoder of the Laplacian parameter for each DCT band and the creation of an SI estimate at the encoder side. This choice was necessary since the residuals obtained with the UWSI and BWSI methods are not reliable enough for the decoder estimation of the Laplacian parameter. An online adaptive correlation noise model that learns an accurate value of the Laplacian parameter [28] along the decoding process seems a rather promising approach to follow to solve this problem with a more realistic approach.

The IST-DVC codec used in this Thesis to integrate the proposed SI creation solutions follows the Stanford DVC architecture originally proposed in [22], but with several techniques to improve the RD performance while still maintaining a low encoder complexity. The IST-DVC codec also includes state-of-the-art DVC coding techniques available in the literature, such as an improved reconstruction method [29], to reach a rather powerful DVC solution. In summary, the IST-DVC codec includes at the encoder, the H.264/AVC 4 × 4 DCT transform, a uniform scalar quantizer and a LDPC syndrome code
as the Slepian-Wolf codec. At the decoder, IST-DVC uses a CRC code for error detection and a minimum mean-square error reconstruction method.

The proposed DVC-UWSI and DVC-BWSI codecs correspond to the IST-DVC codec proposed in [29] with the UWSI and BWSI creation solutions integrated, respectively. These two DVC codecs are compared to a IST-DVC codec using the MCFI SI creation solution (DVC-MCFI), i.e. using a simple translational motion model for the SI creation. For the performance evaluation, the IST-DVC codec proposed in [29] will be mentioned as DVC-MCFI to clearly distinguish which SI creation module is being used.

5.1.3 Coding Benchmarks

To evaluate the performance of the proposed DVC-UWSI and DVC-BWSI video codecs, the RD performance is compared to some benchmark solutions, notably the H.264/AVC Intra, the H.264/AVC Zero Motion and the DVC-MCFI codec. These video coding solutions share an important characteristic: all the encoders under evaluation have low encoding complexity as they do not use motion estimation at the encoder. The video coding solutions used as benchmarks are:

- **H.264/AVC Intra** – An H.264/AVC video codec using only the Intra mode is one of the most powerful and efficient codec following an Intra coding solution. This codec uses the Main profile and exploits the spatial redundancy with several $4 \times 4$ and $16 \times 16$ Intra prediction modes. Although it increases the encoding complexity, the CABAC arithmetic encoding is also employed.

- **H.264/AVC Zero Motion** – The H.264/AVC Zero Motion video codec that exploits some temporal redundancy is also a relevant benchmark; in this case, no motion estimation is performed at the encoder. The H.264/AVC Zero Motion encoder uses a motion search range of zero, i.e. only the collocated blocks in the previous and/or future reference frames are used for prediction. The same video structure adopted for the DVC-UWSI and DVC-BWSI codecs was used, i.e. a GOP size of 2, with an IBI GOP structure and two reference frames, one in the past and another in the future. The Main profile is used, and since no motion estimation is performed, the encoding complexity is kept low.

- **DVC-MCFI codec** – To also assess the DVC-UWSI and DVC-BWSI performances regarding an available DVC codec, the DVC-MCFI codec performance is also taken as benchmark. The only difference between the DVC-MCFI video codec and the proposed codecs is the side information module; all the remaining techniques are the same. By comparing the proposed DVC-UWSI and DVC-BWSI codecs with the DVC-MCFI codec, it is possible to assess the RD performance benefits of adopting a geometric motion model for the SI creation process, with respect to the rather popular translational motion model.

The coding conditions are the same for all DVC codecs. For all codecs under evaluation, only the luminance component was coded, meaning that the SI quality and the RD performance consider only the luminance rate and quality (both key frames and WZ frames).
5.1.4 Performance Evaluation Metrics

The quality metric used for the performance evaluation is the Peak Signal-to-Noise Ratio (PSNR). The well-know PSNR metric is still the most common metric used for video quality evaluation, despite its shortcomings to reliably express the perceptual (subjective) quality. The PSNR metric is a full reference metric as it measures the quality of a given decoded frame, \( D \), with respect to the corresponding original frame, \( O \), as follows:

\[
PSNR = 10 \log_{10} \left( \frac{L_{\text{max}}^2}{MSE} \right)
\]  

(5.1)

In (5.1), \( L_{\text{max}} \) is the maximum luminance sample value, which in this case is 255 as 8-bit samples are used, and MSE is the mean square error calculated between the decoded and corresponding original frames as:

\[
MSE = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [O(i,j) - D(i,j)]^2
\]  

(5.2)

where \( O(i,j) \) and \( D(i,j) \) are the luminance values of the original and decoded frames at position \((i,j)\) and \( m \times n \) is the spatial resolution of the video sequence.

5.2 Side information Performance Evaluation

The main target of this Section is to evaluate the side information quality obtained with the proposed UWSI and BWSI solutions standalone, i.e. before integration in any DVC codec. Thus, the UWSI and BWSI overall SI qualities are only compared to the MCFI SI quality [24] for all the test sequences. Also, the temporal evolution of the SI quality (for each frame) is presented to better understand the strengths and weaknesses of the proposed solutions in terms of quality temporal variation.

The average SI quality for the whole sequence for the UWSI, BWSI and MCFI solutions is shown in Table 5.3. Also, \( \Delta_{UWSI} \) and \( \Delta_{BWSI} \) are the average SI gains obtained for the UWSI and BWSI solutions, respectively, and are computed as:

\[
\Delta_{UWSI} = UWSI_{dB} - MCFI_{dB}
\]  

(5.3)

\[
\Delta_{BWSI} = BWSI_{dB} - MCFI_{dB}
\]  

(5.4)

The SI quality depends on the key frames quality and the accuracy of the SI creation method. This evaluation will allow comparing later the gains obtained in terms of SI quality with the complete RD performance gains, for the proposed DVC-UWSI and DVC-BWSI codecs.
Table 5.3 – SI PSNR quality for each video sequence tested.

<table>
<thead>
<tr>
<th>Motion Activity</th>
<th>Video Sequence</th>
<th>$Q_1$</th>
<th>MCFI [dB]</th>
<th>UWSI [dB]</th>
<th>BWSI [dB]</th>
<th>$\Delta_{\text{UWSI}}$ [dB]</th>
<th>$\Delta_{\text{BWSI}}$ [dB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Container</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>37.81</td>
<td>37.76</td>
<td>37.78</td>
<td>-0.05</td>
<td>-0.03</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>40.08</td>
<td>39.95</td>
<td>39.99</td>
<td>-0.14</td>
<td>-0.09</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hall</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>31.35</td>
<td>31.54</td>
<td>31.54</td>
<td>0.18</td>
<td>0.19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>34.51</td>
<td>34.90</td>
<td>34.92</td>
<td>0.38</td>
<td>0.40</td>
<td></td>
</tr>
<tr>
<td>Average (Low)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>34.58</td>
<td>34.65</td>
<td>34.66</td>
<td>0.06</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>37.30</td>
<td>37.42</td>
<td>37.46</td>
<td>0.12</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>Medium</td>
<td>Foreman</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>26.63</td>
<td>26.79</td>
<td>26.70</td>
<td>0.17</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>28.66</td>
<td>29.02</td>
<td>28.93</td>
<td>0.36</td>
<td>0.27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mobile and Calendar</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>26.21</td>
<td>26.87</td>
<td>26.93</td>
<td>0.67</td>
<td>0.72</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>28.06</td>
<td>28.96</td>
<td>29.06</td>
<td>0.90</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Coastguard</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>28.51</td>
<td>28.57</td>
<td>28.55</td>
<td>0.06</td>
<td>0.04</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>30.67</td>
<td>30.75</td>
<td>30.81</td>
<td>0.09</td>
<td>0.14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table Tennis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>26.01</td>
<td>25.99</td>
<td>26.06</td>
<td>-0.02</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>28.51</td>
<td>28.63</td>
<td>28.64</td>
<td>0.12</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>Average (Medium)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>26.84</td>
<td>27.06</td>
<td>27.06</td>
<td>0.22</td>
<td>0.22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>28.97</td>
<td>29.34</td>
<td>29.36</td>
<td>0.37</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>High</td>
<td>Bus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>21.90</td>
<td>21.80</td>
<td>22.32</td>
<td>-0.10</td>
<td>0.42</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>23.92</td>
<td>23.84</td>
<td>24.58</td>
<td>-0.09</td>
<td>0.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stefan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>20.51</td>
<td>20.71</td>
<td>20.67</td>
<td>0.20</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>22.16</td>
<td>22.51</td>
<td>22.53</td>
<td>0.36</td>
<td>0.37</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Soccer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>21.53</td>
<td>21.65</td>
<td>21.34</td>
<td>0.12</td>
<td>-0.19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>21.98</td>
<td>22.12</td>
<td>21.72</td>
<td>0.14</td>
<td>-0.26</td>
<td></td>
</tr>
<tr>
<td>Average (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_2$</td>
<td>21.32</td>
<td>21.39</td>
<td>21.44</td>
<td>0.07</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$Q_7$</td>
<td>22.69</td>
<td>22.82</td>
<td>22.94</td>
<td>0.14</td>
<td>0.26</td>
<td></td>
</tr>
</tbody>
</table>

From the results in Table 5.3, the following conclusions may be derived:

- The most significant SI quality gains are obtained when the key frames quality is better, i.e. when a lower QP is used (the QPs corresponding to the RD point $Q_2$). In fact, some results obtained for RD point $Q_2$, can be easily explained by the poor quality of the key frames, which restricts the
gains of the geometric approach, i.e. when the quantization error of the reference frames is high, it is difficult to appropriately model the motion.

- As expected, the average SI PSNR quality is better for the low motion activity sequences (36 dB), then for the medium motion activity sequences (around 28 dB) and, at last, for the high motion activity sequences (around 22 dB). When more complex motion occurs, it is more difficult to accurately model the motion and worst SI quality results are obtained.

- For the low motion activity sequences, the results obtained for both the Container and Hall Monitor sequences are quite different. In fact, although both video sequences have a static camera, the Hall Monitor sequence has a static background, while the Container sequence, mainly due to the sea waves, does not have. This characteristic is rather important, since in the Container sequence, the proposed solutions constantly choose geometric deformations that do not fit the true motion of the sequence. So, different results are obtained for both sequences, despite being both considered with low activity.

- Regarding the medium motion activity sequences, the proposed UWSI and BWSI solutions show significant gains, in particular for the Mobile and Calendar sequence. This can be easily explained by the camera motion present in this sequence, which contains a zoom out and a slow left-pan, because these rather complex motions can be better described by a geometric motion model, instead of a translational motion model. For the other medium motion activity sequences, the gains are also significant. For the Foreman sequence, some improvements are obtained when some complex motion related to the facial expressions occurs. For the Coastguard sequence, the pan-left and pan-right are very slow and can be well described by the translational motion model, as may be seen by the high PSNR value obtained, leaving short space for geometric motion model improvements. However, with a temporal resolution of 15Hz, the initial zoom out present in the Table Tennis sequence is rather fast, which difficults an accurate estimation of the motion in the sequence.

- For the high motion activity sequences, the Bus sequence is the one allowing the BWSI solution to show its benefits by providing larger SI quality. In fact, the most significant BWSI gain regarding the UWSI solution is obtained for this sequence due to the two unidirectional motion compensation modes available, that properly exploit the fast left-pan presented in this sequence.

In Figure 5.3 to Figure 5.5, the temporal evolution of the SI quality is shown for the Container, Mobile and Calendar and Bus sequences.
From the results presented in Figure 5.3 to Figure 5.5, the following conclusions may be drawn:

- For the Container sequence, the UWSI and BWSI solutions clearly loose regarding the MCFI. The SI is generated with rather stable quality except for a certain period, where the birds moving in the visual scene lead to a reduction in quality due to the difficulty to estimate their movements. From Figure 5.3, it is easy to conclude that the proposed solutions have a constant gap in terms of SI quality. In fact, although the Container sequence has a static camera, it doesn't have a constant
background, mainly due to the sea waves, that are constantly moving. This characteristic is rather important because the geometric motion model is constantly used to obtain deformations that worsen the SI quality. On this subject, the penalty offset introduced in the Motion Model Decision in Section 3.3.6 plays an important role: without it, the losses would go up to 0.8 dB on the Container video sequence, instead of 0.14 dB.

- Regarding the Mobile and Calendar sequence, significant gains of up to 0.9 dB for the UWSI and 1 dB for the BWSI are obtained when compared to the MCFI solution. As shown in Figure 5.6, this sequence has a zoom out from the beginning until frame 74, where a slow panning starts to stay until the end of the sequence. This difference can be observed in Figure 5.4, where the larger gains are obtained when the zoom out camera motion occurs, i.e. when the translational model cannot capture the true motion in the video sequence.

![Figure 5.6 – Evolution of the camera motion in the Mobile and Calendar sequence.](image)

- For the Bus sequence, Figure 5.5 shows that the proposed BWSI solution provides significant SI quality gains for some parts of the sequence. The strong camera panning in this sequence is exploited by the two unidirectional motion compensation modes available in the BWSI solution. Also, the UWSI solution shows small losses for the Bus sequence when compared to the MCFI, mainly because the geometric motion model obtains deformations that worsen the final SI quality. As shown in Figure 5.5, this allows significantly improving the SI quality when compared to the MCFI and UWSI solutions.

### 5.3 RD Performance Evaluation

In the previous section, the SI quality obtained with the two proposed SI creation frameworks was presented and compared to the state-of-the-art MCFI method. The higher the side information quality, the fewer the (parity) bits necessary to transmit from the encoder to the decoder to achieve a certain target quality. Thus, the SI quality plays a key role in the overall RD performance. The main objective of this section is to evaluate the RD performance of the two proposed DVC codecs: DVC-UWSI and DVC-BWSI. Figure 5.7 to Figure 5.15 illustrate the RD performance of both the DVC-UWSI and DVC-BWSI codecs for the eight RD points, according to the test conditions defined in Section 5.1. The RD performance of the two proposed solutions is compared against the benchmark video coding solutions described in Section 5.1.3: DVC-MCFI, H.264/AVC Intra and H.264/AVC Zero Motion.
Figure 5.7 – DVC-UWSI and DVC-BWSI codecs RD performance for the Container sequence.

Figure 5.8 – DVC-UWSI and DVC-BWSI codecs RD performance for the Hall Monitor sequence.

Figure 5.9 – DVC-UWSI and DVC-BWSI codecs RD performance for the Foreman sequence.
Figure 5.10 – DVC-UWSI and DVC-BWSI codecs RD performance for the Mobile and Calendar sequence.

Figure 5.11 – DVC-UWSI and DVC-BWSI codecs RD performance for the Coastguard sequence.

Figure 5.12 – DVC-UWSI and DVC-BWSI codecs RD performance for the Table Tennis sequence.
Figure 5.13 – DVC-UWSI and DVC-BWSI codecs RD performance for the Bus sequence.

Figure 5.14 – DVC-UWSI and DVC-BWSI codecs RD performance for the Stefan sequence.

Figure 5.15 – DVC-UWSI and DVC-BWSI codecs RD performance for the Soccer sequence.
From the results obtained, the following conclusions may be drawn:

- **DVC-UWSI and DVC-BWSI codecs versus H.264/AVC Intra codec** – In this case, the two proposed DVC codecs are compared to an efficient Intra codec, the H.264/AVC Intra codec. For many sequences, the proposed DVC codecs outperform the H.264/AVC Intra codec. In fact, certain trends may be found, depending on the motion activity in the video sequences:
  
  - For low motion sequences, the DVC-UWSI/BWSI video codecs achieve significant RD performance gains (up to 5 dB for the Container sequence) when compared to H.264/AVC Intra, due to the static camera and low amount of object motion.
  
  - For medium motion sequences, the proposed video codecs also achieve significant improvements, which go from 0.3 dB for the Foreman sequence to 4 dB for the Mobile and Calendar sequence.
  
  - For high motion sequences, the proposed DVC-UWSI and DVC-BWSI codecs do not perform better than H.264/AVC Intra, since the proposed motion models are not able to capture the complex motion of those sequences. Although the DVC-BWSI codec achieves gains of 0.2 dB against the H.264/AVC Intra codec for the Bus sequence, the RD performance of the two proposed DVC codecs is generally worse than H.264/AVC Intra, with losses up to 2 dB for the Soccer sequence.

- **DVC-UWSI and DVC-BWSI codecs versus H.264/AVC Zero Motion codec** – In this case, the proposed DVC codecs are compared to a low complexity predictive Inter encoder (at least regarding the full H.264/AVC Inter codec) which exploits the temporal correlation with a DPCM-like scheme. In general, the two proposed DVC codecs do not outperform the H.264/AVC Zero Motion codec. However, the proposed DVC codecs show gains up to 1.5 dB for the Coastguard, Mobile and Calendar and Bus sequences. These gains can be explained by the slow panning presented in Coastguard and Mobile and Calendar sequences, which the proposed DVC-UWSI/BWSI codecs can properly estimate (at the decoder) and the H.264/AVC Zero Motion codec cannot, as there is no motion search. For the Bus sequence, due to the fast panning, only the DVC-BWSI codec can outperform the H.264/AVC Zero Motion codec (up to 0.2 dB). However, the H.264/AVC Zero Motion codec has higher encoding complexity when compared to the proposed DVC codecs, as shown in the encoding complexity evaluation presented in [29]. Note that no changes were made to the encoders of the proposed DVC codecs and thus, the DVC encoding complexity is kept low.

- **DVC-UWSI and DVC-BWSI codecs versus DVC-MCFI codec** – For most cases, the two proposed DVC solutions outperform the state-of-the-art DVC-MCFI codec. In fact, the proposed DVC codecs show gains up to 0.6 dB for the Mobile and Calendar sequence, with an average gain of 0.2 dB for all sequences. The Container sequence is the only sequence where the proposed codecs do not outperform the DVC-MCFI codec, with losses up to 0.1 dB, which is consistent with the losses observed in the SI quality. The warping technique used for the SI creation process is the main responsible for the improvements in the RD performance, since all the remaining techniques are kept the same. This is a rather important result since it validates the assumption that geometric motion models can more accurately track the motion in a video sequence.
• **DVC-BWSI codec versus DVC-UWSI codec** – The only sequence where the DVC-BWSI codec outperforms the DVC-UWSI codec is the Bus video sequence, which confirms the SI quality results in Section 5.2. The bidirectional approach combined with the two unidirectional motion compensation modes can properly exploit the fast left-pan presented in this sequence, leading to significant improvements in the DVC-BWSI RD performance. For the other sequences, the DVC-BWSI and DVC-UWSI RD performance curves are very similar. The DVC-BWSI has a higher decoding complexity, since the geometric search, which is the most complex process, is performed four times for each SI frame, while the DVC-UWSI only performs it two times.

**5.4 Conclusions**

This chapter aimed at evaluating the RD performance of the proposed DVC-UWSI and DVC-BWSI codecs. After defining the test conditions, the SI quality for both UWSI and BWSI solutions was assessed for each video sequence. Then, the RD performance results for the conducted tests were presented for both DVC-UWSI and DVC-BWSI codecs.

The two proposed DVC-UWSI and DVC-BWSI codecs have better RD performance than the previous state-of-the-art DVC-MCFI video codec. The obtained gains are mainly due to the improvements in the motion modeling process; in this case, a high-order geometric model was able to better capture camera motions (such as zooms) that are not enough accurately described with a translational motion model, such as the one used by the DVC-MCFI video codec. The improvements in the SI quality lead to improvements in the overall RD performance, since fewer requests are made by the decoder to reach the target decoded quality. The most significant gains were obtained for the medium and high motion sequences, notably for the ones with pannings and zooms, which are better characterized using a geometric motion model instead of the translational motion model.
Chapter 6

Final Remarks

In this chapter, the work performed in this Thesis is briefly summarized, the main achievements identified and some possibilities for future work presented.

6.1 Summary

The first chapter of this Thesis introduces the context of this research, highlighting the importance of the distributed video coding paradigm for some emerging application scenarios. This chapter also focus on the importance of advanced motion models (such as geometric transforms) to further obtain compression gains. Then, in Chapter 2, practical coding schemes (such as the DISCOVER DVC codec) that exploit the potential of the distributed video coding paradigm are explained; the geometric transforms are analyzed more in-depth since they hold the promise of better modeling of the motion (temporal correlation) of a video sequence. In the same chapter, several work available in the literature is reviewed to better understand the most relevant background technologies for this Thesis, notably the strengths and weaknesses of both geometric transforms and side information creation techniques. This review of the state-of-the-art work motivates the main idea of this Thesis: to exploit advanced motion models, such as geometric transforms, to obtain advances in side information creation. In Chapter 3, the first SI creation solution is proposed: the Unidirectional Warping Side Information (UWSI) solution. First, a new architecture for side information creation that exploits geometric transforms is proposed, followed by a detailed description of all techniques integrated in this UWSI solution. With this UWSI solution, each block can be estimated with a perspective motion model (several parameters are estimated for each block) or with a simple translational motion model (one motion vector per block). In a similar way, Chapter 4 presents the second SI creation solution: the Bidirectional Warping Side Information (BWSI) solution. After presenting the BWSI creation architecture, the new coding tools are described in detail, notably regarding the previous UWSI solution. The BWSI is able to improve the UWSI for the cases where occlusions occur, since there are motion models in two opposite directions and a unidirectional motion compensated mode (just one reference frame) is used to create the side information. In Chapter 5, the performance of both SI
creation solutions was evaluated. After defining the adopted test conditions, the SI quality of both solutions was evaluated in comparison with previous state-of-the-art, a MCFI solution that follows a pure translational approach. Then, the proposed solutions were integrated in a DVC codec, creating the DVC-UWSI and DVC-BWSI codecs, which were properly evaluated in terms of RD performance.

6.2 Achievements

The main objective of this Thesis is to improve the overall RD performance of a DVC codec that uses a 'guess' side information creation approach, i.e. when the side information frame is estimated just with the reference frames (no additional information is transmitted by the encoder). Since the RD performance of a distributed video codec strongly depends on the SI creation module, it is rather important to improve the SI quality obtained at the decoder. The improvement of the SI quality has two important consequences: first, it reduces the bitrate, because fewer requests are made by the Slepian-Wolf decoder; secondly, it improves the final reconstructed WZ frame quality.

To improve the SI quality, it was proposed on this Thesis, for the first time, to exploit geometric motion models. In fact, the SI quality has been enhanced since the use of geometric deformations to estimate the SI frame leads to a better characterization of complex motions that cannot be well estimated by a traditional translational approach. However, some techniques, such as the hierarchical approach and the unidirectional motion compensation mode, were also helpful to achieve significant gains. Using these techniques, two DVC codecs that only differ on the SI creation module were obtained: DVC-UWSI and DVC-BWSI. With these two proposed codecs gains up to 1 dB were obtained in terms of SI quality and 0.6 dB in terms of RD performance when compared to the previous state-of-the-art DVC-MCFI codec. Regarding the H.264/AVC Intra and H.264/AVC Zero Motion, RD gains of up to 5 dB and 1.5 dB are obtained, respectively. Considering the promising experimental results obtained, it was decided that the solutions proposed in this Thesis should be submitted to a relevant journal for publication.

It is also important to note that all of the objectives defined for this Thesis in Chapter 1 have been accomplished.

6.3 Future work

This section discusses some of the work that naturally follows from the techniques proposed here, and that may be pursued in the future. Naturally, the focus is still to obtain better RD performance improvements since it is still difficult to obtain gains over H.264/AVC Intra for high motion sequences and RD gains over H.264/AVC zero-motion. To address the challenge of creating high quality side information, the following research ideas can be explored:

- **Better unidirectional compensation** – As described in the BWSI solution in Section 4.2.2, the decision for the unidirectional compensation mode is based only on the deformation obtained for the SI block. A possible improvement might be achieved if the decision of using the unidirectional compensation mode is based also on the neighboring SI blocks. This way, this mode is not often incorrectly used, i.e. SI block classified with the unidirectional compensation mode should not be surrounded by SI blocks with a different mode.
Global and local geometric transforms – To obtain a more coherent GT motion field, it is important to avoid random deviations of the deformations between neighboring SI blocks. A possible approach is to develop an algorithm that merges similar geometric transforms that follow a certain criteria. In this way, some wrongly chosen deformations can be eliminated, thus providing a more coherent GT motion field. This solution could provide benefits for smooth regions. However, the trade-off between the GT motion field smoothness and the good characterization (accuracy) of singular objects must be carefully considered to avoid ruining the characterization of small objects with complex motion.

Following the same reasoning, a global geometric transform might be obtained to improve the initial starting point of the GT motion field. The main target is to obtain a good estimation for the background global motion first.

Geometric continuous method – The proposed UWSI and BWSI solutions use a discontinuous method, i.e. the best deformation for every SI block is obtained without considering the neighbor GT vectors. So, the use of a continuous method for the GT vectors is also a good idea (see Figure 6.1).

So, a continuous method might be applied to further increase the smoothness of the GT motion field, but still obtaining geometric transforms that accurately represent the motion.

As in previous case, there is a trade-off between the GT motion field smoothness and the good characterization of singular objects. Another possibility is to employ both continuous and discontinuous methods, creating an algorithm that initially uses the discontinuous method and, later, decides whether the four GT vectors of a given vertex, one of each block, are close enough to consider the continuous method or far enough to keep the discontinuous method.

Hierarchical motion estimation approach – Regarding the hierarchical motion estimation approach, a further motion estimation layer at 4 × 4 block size might be considered, since the DVC decoder also uses a 4 × 4 DCT.

Bilinear geometric transform – Although the geometric transform used in this work was the perspective transform, there are possibilities such as the affine and the bilinear transforms. Although the affine transform is not recommended due to its limitations in terms of mapping quadrilaterals, the bilinear transform provides the same mapping possibilities as the perspective
transform. Therefore it might be useful to evaluate which geometric transform suits best this coding scenario.

- **DVC Multi-view video coding** – The two proposed DVC-UWSI and DVC-BWSI codecs can also be useful when applied to a multi-view video scenario. Typically, in a multi-view video scenario, several cameras capture the same scene from different viewpoints, which means there is a lot of correlation between the several views of the video scene. The great advantage of the distributed video coding paradigm for multi-view video is that the cameras do not need to communicate between each other to exploit the temporal, spatial and inter-view correlations as this is done at the decoder.

In conclusion, the objectives of this Thesis have been accomplished. It has been proven that generating side information using a geometric motion model can improve the RD performance of a DVC codec. Despite some future improvements already mentioned, major technical advances have been achieved with this work, since there is currently no SI creation solution in the literature based on geometric transforms.
References


