A Novel Multi-View Image Coding Scheme based on View-Warping and 3D-DCT

M. Zamarin[1], S. Milani[2], P. Zanuttigh[1], G.M. Cortelazzo[1]

Department of Information Engineering
University of Padova, Italy
Via Gradenigo 6/B
35131 Padova - Italy

Abstract

Efficient compression of multi-view images and videos is an open and interesting research issue that has been attracting the attention of both academic and industrial world during the last years. The considerable amount of information produced by multi-camera acquisition systems requires effective coding algorithms in order to reduce the transmitted data while granting good visual quality in the reconstructed sequence. The classical approach of multi-view coding is based on an extension of the H.264/AVC standard, still based on motion prediction techniques. In this paper we present a novel approach that tries to fully exploit the redundancy between different views of the same scene considering both texture and geometry information. The proposed scheme replaces the motion prediction stage with a 3D warping procedure based on depth information. After the warping step, a joint 3D-DCT encoding of all the warped views is provided, taking advantage of the strong correlation among them. Finally, the transformed coefficients are conveniently quantized and entropy coded. Occluded regions are also taken into account with ad-hoc interpolation and coding strategies. Experimental results performed with a preliminary version of the proposed approach show that at low bitrates it outperforms the H.264 MVC coding scheme on both real and synthetic datasets. Performance at high bitrates are also satisfactory provided that accurate depth information is available.

Key words:
multi-view image coding, 3D warping, 3D-DCT, multi-view plus depth, 3DTV, 3D spatial prediction

Email addresses: `zamarinm@dei.unipd.it` (M. Zamarin), `simone.milani@dei.unipd.it` (S. Milani), `pietro.zanuttigh@dei.unipd.it` (P. Zanuttigh), `corte@dei.unipd.it` (G.M. Cortelazzo)

[1]Multimedia Technology and Telecommunications Laboratory. Tel: (+39) 049 827 7774.
[2]Digital Signal and Image Processing Laboratory. Tel: (+39) 049 827 7641.

1. Introduction

In recent years, there has been a growing interest in three-dimensional representation, especially in 3D video systems. While commercially-available stereoscopic video systems allow the viewer to perceive the depth of the scene from a given viewpoint, an unconstrained browsing of the scene from an arbitrary viewpoint (Free-Viewpoint Video or FVV) is a much more complex issue on which research is still very active. Indeed if a depth map associated to a single view of the scene permits warping the view to a novel viewpoint, one needs several views and depth maps in order to achieve satisfactory viewing freedom and to avoid occlusions [1].

The so-called multi-view plus depth representation permits to freely browse a three-dimensional scene represented by multiple views and depth maps as if a 3D model was available. In this way, all the burden connected to the construction of the 3D representation is avoided. This approach nowadays appears to be one of the most effective solutions to the challenges of real-time 3DTV applications. However, in order to permit a satisfactory free browsing of the scene, a large number of views has to be transmitted, and their compression is a fundamental step in the construction of a practical multi-view system.

Although conceivable, compressing multi-view representations independently by standard single-view image and video coding techniques (eventually with *ad-hoc* optimizations [2]) is not very efficient since there is a quite relevant redundancy between different views of the same scene that a similar approach would not use. An effective coding scheme should be able to exploit such a redundancy in order to obtain an optimal compression of the image or video data. To this purpose several solutions have been proposed in the literature. Some of these schemes aim at reducing the inter-view redundancy by some low-pass transformation across the pixels of the different views, like in [3]. This work is based on a 3D-DCT, but the transform is directly applied to the source images without any disparity compensation method. As a result the encoding method of [3] can not fully exploit the energy compaction property of the DCT transform.

Other solutions apply motion compensation techniques, typically adopted to reduce temporal redundancy, in order to predict one view from another relative to a different viewpoint. Following this trend, the video coding standard H.264 MVC [4], developed from the scalable video coder H.264 SVC [5], obtains high compression gains by predicting each frame of a single view both from previous data and from the other views. A disparity based method for motion vectors' prediction is introduced in [6] for stereoscopic video compression. It represents an interesting and up-to-date possibility. 3D warping of the temporal-motion vectors in one view is used to predict the motion vectors in the other one. Warping is performed by an affine transformation based on a disparity model updated frame by frame.

Among the class of Multi-View Coding (MVC) schemes employing motion compensation, one must also mention multi-view video coders based on Distributed Video Coding (DVC) techniques [7]. In this case, each frame is coded according to the correlation existing among the different views, and motion compensation techniques are employed at the decoder. The proposed solutions allow reducing the computational complexity at the encoder and mitigating the effects of data losses on the reconstructed 3D sequences.

Despite the fact that these approaches make possible to reuse a huge amount of previous work on video compression, they do not fully exploit the three dimensional nature of multi-view data. For example, the location of corresponding samples in different views is given by the three dimensional structure of the scene (given by the depth maps), and it is possible to employ this knowledge instead of the standard block-based motion prediction [8].

Moving further, it is possible to exploit different schemes than the classical ones based on motion prediction and residual coding. Even if in the literature it is possible to find some examples of video coders based on three dimensional transforms involving the spatial and temporal dimensions [9], the use of this kind of representations is not very common because of the problems with scene changes and the unreliability of motion prediction. Another interesting possibility is given by the method of [10] which provides scalability and flexibility as well as good coding performances exploiting a multi-dimensional wavelet transform combined with an *ad-hoc* disparity-compensated view filter (DCVF).

This work introduces a novel lossy compression scheme for multi-view data which tries to fully exploit the inter-view redundancy in order to achieve better compression performance. A video coder is essentially made of a redundancy reduction mechanism followed by a coding part. The focus of this work is on the first element, i.e., on the exploitation of inter-view redundancy. The novel elements of the proposed scheme with respect to H.264 MVC or similar schemes are several. Firstly the inter-view motion prediction stage is replaced by a 3D warping procedure based on depth information. Then, the traditional 2D-DCT is replaced by a multi-dimensional transform, namely a 3D-DCT. However, in the proposed method, differently from [3], the 3D-DCT is applied on disparity compensated data, in order to better exploit view-redundancy in the transform stage. The transform coefficients are then coded by standard entropy coding algorithms. Finally, a novel interpolation and compression scheme is introduced in order to handle the regions that are occluded or out of the viewing field in some views but not in others.

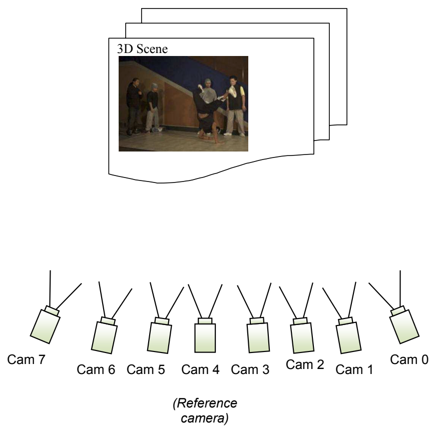
The rest of the paper is organized as follows. Section 2 presents the basic ideas that lie behind the adopted compression strategy. Section 3 provides an overview of the adopted coder describing its main parts. Experimental results in Section 4 show that the proposed approach can improve the average PSNR value over the different views up to 2 dB with respect to H.264 MVC encoding performed with comparable coding features. Finally, conclusions are drawn in Section 5.

2. Overview of the proposed framework

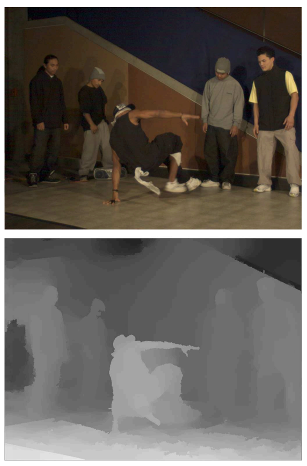
As previously stated in this paper, we aim to efficiently encode multi-view images, i.e. multiple views of a three dimensional scene taken from different viewpoints. We assume that all the images are taken by similar cameras and have the same resolution, but the cameras can be placed in arbitrary positions. As it is easy to expect, the cameras distribution plays an important role in the achievement of a good coding efficiency, since the correlation among views strongly depends on the reciprocal position of the viewpoints.

In addiction to texture images, the proposed scheme makes also use of geometry information. Geometry description could be provided by a complete three dimensional representation of the scene or by depth maps associated to each view. In the following description we will focus on depth maps because they are commonly used in 3D video applications. Anyway, a three dimensional model of the scene can be used in the proposed framework in place of them. In conclusion, the input data of the algorithm we propose is a multi-view image with depth maps relative to each view and all the camera intrinsic and extrinsic parameters.

An example of this kind of representation is provided by the "breakdancers" sequence from Microsoft Research [11], commonly used to validate the performance of 3D video compression algorithms. This sequence is composed of 8 video streams acquired by a set of cameras placed at regular steps along a nearly linear path (Figure 1a). The dataset also includes depth information for each frame of each camera computed by stereo vision algorithms.



a)



b)

Figure 1: Sequence "breakdancers": a) camera arrangement and b) first image (from "Cam 0") and relative depth map.

The basic idea behind most video encoders is to take advantage of the strong temporal correlation between consecutive frames in order to reduce as much as possible the size of the final bistream. In the multi-view case, a similar principle is applied. The correlation lying among all the views can be exploited in order to achieve good compression performances. To this purpose, a multi-view image could be seen as a video sequence where adjacent views of the same scene can be

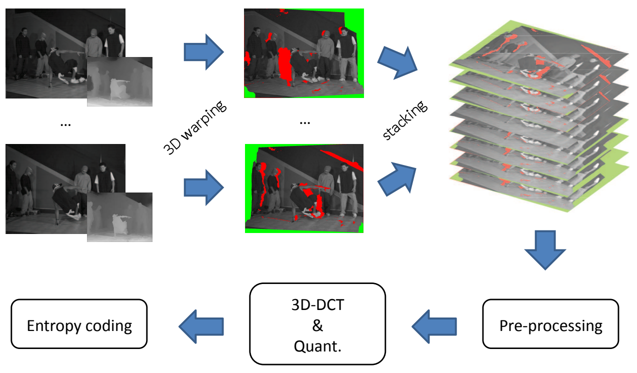


Figure 2: Algorithm's main steps

referred to temporally-adjacent frames. As a matter of fact, many multi-view compression schemes reuse the temporal motion estimation and compensation algorithm along the inter-view dimension.

However, differently from a classical video sequence, in a typical multi-view image the difference between adjacent images (views) is not due to the motion of objects or people, but rather to changes of viewpoint. In this situation, experimental results show that the *motion estimation* approach of many up-to-date video coding standard, like H.264/AVC, does not provide optimal results both because the different views are less correlated with respect to temporally-adjacent frames of a standard video and because they do not take advantage of the information about the scene geometry. This suggests trying different *ad-hoc* coding techniques in order to better exploit this new kind of correlation among images.

The coding scheme we propose is based on two main operations: 3D-warping of the available views and 3D-DCT based encoding of the warped images (Figure 2). The key idea is to pile up all the different views into a unique "stack" of views and to efficiently encode such a structure.

Let us denote with V_i , $i = 0, \ldots, k-1$ the set of views to be encoded. The coding algorithm has three main steps:

1. In the first step, the algorithm warps all the available views with respect to a reference one V_{ref} chosen among the available views V_i , $i = 0, \ldots, k-1$. In this way, for each of the k views to be encoded we obtain a new image denoted with $V_{i \to ref}$ that represents a prediction of the reference view from the i-th view. Note that due to occlusions and out-of-bound regions (i.e. regions in the field of view of V_{ref} but not of V_i), there will be regions in the warped image that can not be determined. The problem of occluded regions will be discussed later. With a reverse-warping analog to the previous one, it is possible to reconstruct all the available views except

for the regions not visible from V_{ref}. As result, the warping process creates a final set of prediction views $V_{i\to ref}$, $i = 0, \dots, k-1$ (which includes the reference view itself). All the prediction images are then arranged in a "image-stack", which represents most of the data we aim to efficiently encode.
2. In the second step, after a hole-filling step, a 3D-DCT transform is applied to the image-stack, in order to exploit the well-known "energy compaction" property of the Discrete Cosine Transform. Transformed coefficients are then conveniently quantized and entropy coded.
3. In the final step we deal with the occlusions encoding the regions of the available views that are not visible in the reference view V_{ref}.

Algorithm implementation and occlusion regions handling are discussed in the next section.

3. Description of the coding architecture

The proposed coding algorithm encompasses four main steps:

- 3D-warping of all the views with respect to the reference view and creation of the image stack.
- Pre-processing operations on the stack.
- Stack encoding.
- Encoding of occlusions.

Figure 3 shows a block-diagram of the proposed procedure. In the following subsections each step will be described in detail.

3.1. Warping of the different views into the image stack

The first step consists in 3D-warping each view V_i to the viewpoint of the reference view V_{ref} using depth information (let us recall that for each view V_i we assure to have available the associated depth map D_i). From calibration information and depth values it is possible to compute the 3D point $\mathbf{P}$ corresponding to each pixel $\mathbf{p_i}$ in each view. By the standard pinhole camera model the 3D point $\mathbf{P}$ can then be reprojected to the various views. Given the availability of depth information for each view, 3D-warping can be done either by forward mapping the pixels from each target view V_i to the reference view V_{ref} with the aid of D_i, or by backward mapping the pixels from the reference view V_{ref} with the aid of D_{ref} to the various views V_i. It is worth noting that both detection and processing of occluded regions are quite different in the two approaches. In this work, we chose backward mapping (that uses only the depth information D_{ref} of the reference view) unless differently specified. However for a better performance in the steps concerning detections and management of occlusions we also considered using the depth map of the source views. Thus,

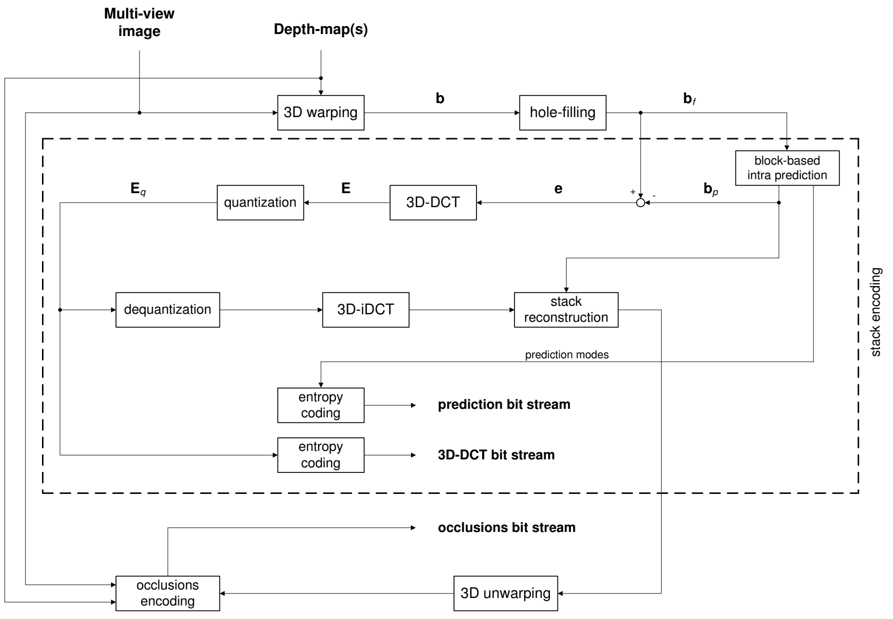


Figure 3: Encoder block-diagram.

our approach is a hybrid approach, essentially based on backward mapping for the image stack formation. The following 3D-warping procedure, illustrated in Figure 4, is used to build the image stack:

1. Let us denote with Π_{ref} the projection matrix corresponding to view V_{ref}. In the pinhole camera model the projection matrix in homogeneous coordinates Π_{ref} can be represented in terms of its intrinsic and extrinsic parameters by the standard equation [12]:

$$\Pi_{ref} = \mathcal{K}_{ref}[\mathcal{R}_{ref}|\mathbf{t}_{ref}], \tag{1}$$

where $\mathcal{K}_{ref}$ is the intrinsic parameters' matrix and the extrinsic parameters are given by the rotation matrix $\mathcal{R}_{ref}$ and the translation vector $\mathbf{t_{ref}}$. For notation simplicity let us call $d_{ref} = D_{ref}(\mathbf{p_{ref}})$ the depth value associated to pixel $\mathbf{p_{ref}}$. The coordinates $\mathbf{P}$ (see Figure 4) of the 3D point corresponding to sample $\mathbf{p_{ref}}$ of V_{ref} can be computed by the following equation (step ①):

$$\mathbf{P} = \mathcal{R}_{ref}^{-1}(\mathcal{K}_{ref}^{-1}\mathbf{p_{ref}}d_{p_{ref}} - \mathbf{t}). \tag{2}$$

2. In the subsequent step 3D point $\mathbf{P}$ is mapped to the corresponding location $\mathbf{p_i}$ on the target view V_i using the pinhole camera model (step ②). If Π_i represents the projection matrix corresponding to view V_i, the location of $\mathbf{p_i}$ is simply $\mathbf{p_i} = \Pi_i\mathbf{P}$. Then the value of the sample in position $\mathbf{p_{ref}}$

on the warped image can be computed using bilinear interpolation of the color of the four samples of the target view closer to $\mathbf{p_i}$ (remember that location $\mathbf{p_i}$ is represented by real valued coordinates while the image is sampled at integer values).

3. Using the Z-buffer algorithm to check for occlusions, it is so possible to warp all the views with respect to the viewpoint of V_{ref} using only D_{ref}.
4. If depth data D_i is also available for view V_i, it is possible to perform an additional check in order to better detect occlusions (e.g. due to objects not visible in V_{ref}). The 3D world point $\mathbf{P'}$ corresponding to the pixel $\mathbf{p_i}$ can be found by depth map D_i relative to V_i (step ③), and the distance between $\mathbf{P}$ and $\mathbf{P'}$ can then be computed. If there is no occlusion, $\mathbf{P}$ and $\mathbf{P'}$ are the same point (Figure 4a), and their mutual distance is zero. Otherwise, the two points will differ since the two cameras are pointing to different parts of the scene (Figure 4b). Thus, occlusion detection can be performed by a threshold-based classification on the distance between couples of projected points.

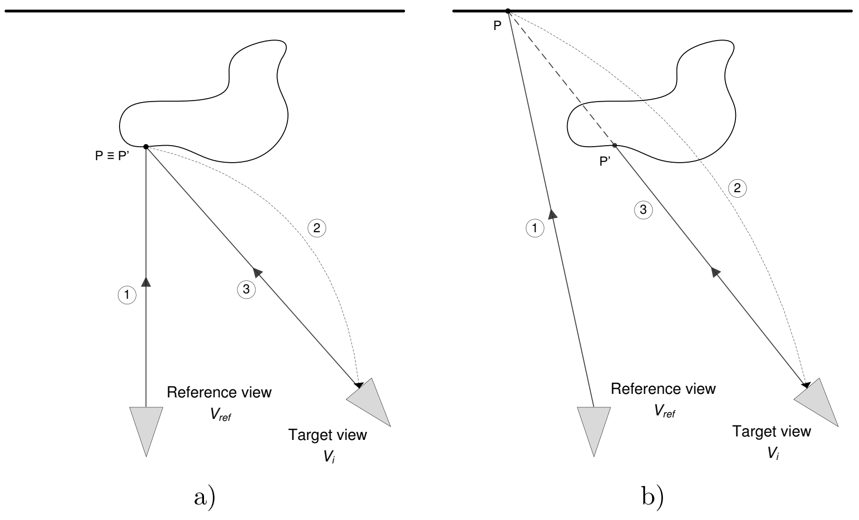


Figure 4: Examples of occlusion detection when depth data of both reference view and target view are available. In the a) case, $\mathbf{P}$ coincides with $\mathbf{P'}$ so the scene's point is correctly viewed by both cameras. In the b) case, $\mathbf{P}$ and $\mathbf{P'}$ differs because the target camera is not able to see $\mathbf{P}$. In this case, an occlusion is detected and the target pixel is not used in the warped image.

Figure 5 refers to the "breakdancers" sequence and shows the image obtained by warping the first frame of view V_0 with respect to the reference view V_4.

Once all the warped images have been computed, the image-stack is created. The image-stack simply consists in a 3D matrix obtained by putting all the warped views one over the other, in the same order of the camera arrangement. An example of all the views composing the stack for the "breakdancers" sequence, before and after the filling process, is shown in Figure 6.

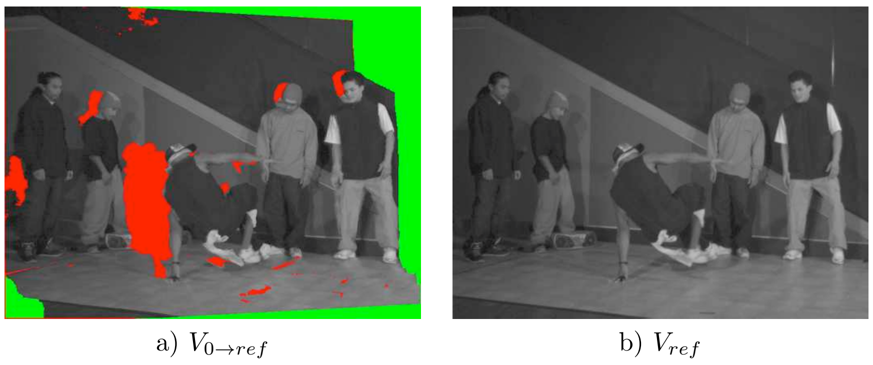

a) $V_{0\to ref}$ b) V_{ref}

Figure 5: a) Warped first image and b) original central view. Red refers to occluded regions while green refers to out-of-bound regions. The red pixels on the floor, background and other artifacts are due to depth maps low accuracy.

In order to reconstruct each original image, the corresponding image-stack view is used. It is important to underline that the previously revealed occluded regions are not used to fill the reconstructed images. For this reason occluded regions in the image-stack do not need to be encoded. However, in order to obtain better compression performances, it is really important to fill in the best way all the pixels in the occluded and out-of-bounds regions as it will be shown in Subsection 3.2.

3.2. Pre-encoding operations

After the warping operation, the image-stack is partitioned into $8 \times 8 \times k$ three-dimensional pixel matrices $\mathbf{b}$ containing the warped pixels $b(x, y, i)$, $x, y = 0, \dots, 7$ and $i = 0, \dots, k-1$. In the following step, each matrix $\mathbf{b}$ has to be transformed into the matrix $\mathbf{B}$ via a separable $8 \times 8 \times k$ 3D-DCT transform, which is obtained applying a 1D-DCT along the rows, the columns, and the views of block $\mathbf{b}$. Applying the transform before the filling process would lead to inefficient performances, since missing pixels are replaced by zero-valued pixels which add high-frequency components to the original signal and reduce the compression gain. In order to mitigate this effect, an interpolation algorithm has been developed. The solution we propose is based on linear pixel interpolation along view dimension. For an 8-views input image, an array $\mathbf{b}(x, y, .)$ of 8 pixels is defined for each position (x, y) along the dimension relative to the different views. Whenever a zero-valued pixel is revealed, it is filled by linear interpolation of the two closest nonzero coefficients around it. If only one nonzero neighbor is available, its value is copied in the position of the missing pixels. Figure 7 depicts some filling examples. In these examples, circles refer to available pixels while stars refer to the filled pixels. Note that the pixel related to the reference view (V_4 in the examples) is always present, since the reference view does not have missing pixels.

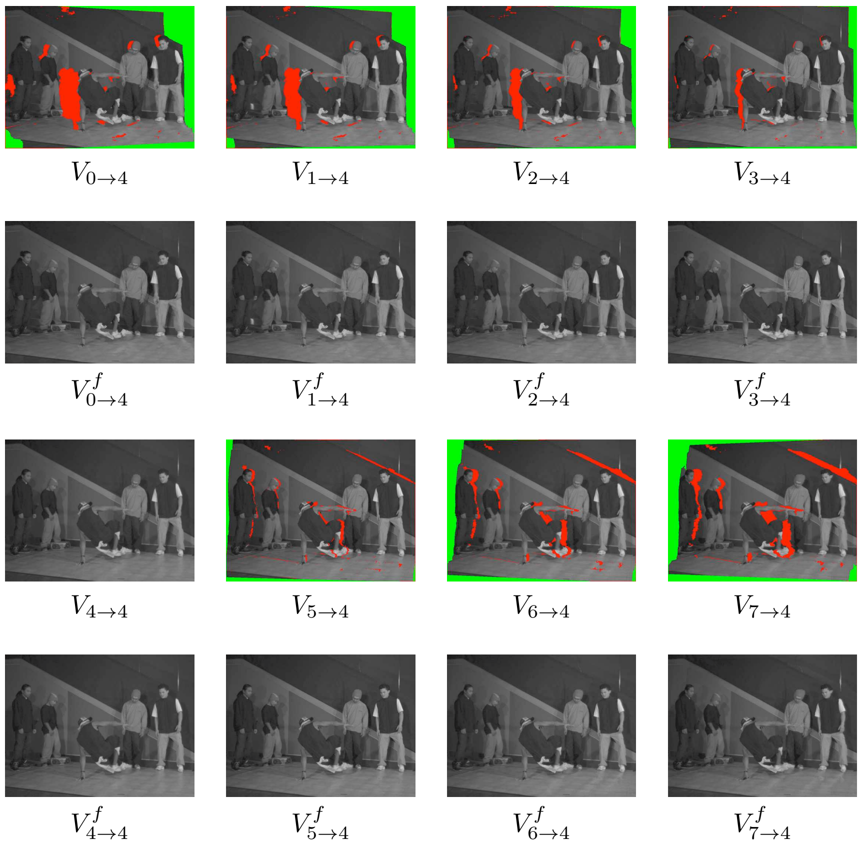


Figure 6: All the eight warped views in "breakdancers" dataset before ($V_{i\to4}, i = 0, \dots, 7$) and after ($V^f_{i\to4}, i = 0, \dots, 7$) the filling process. The filled images are the ones used in the image-stack.

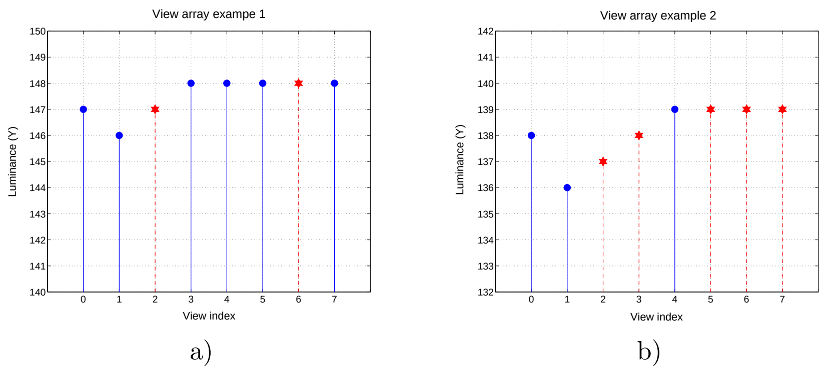


Figure 7: Examples of view arrays filling process.

3.3. Encoding operations

Once the interpolation step is completed, the block $\mathbf{b}$ is converted into a low-pass block $\mathbf{b}_f$. Then, an H.264-like Intra prediction process based on $8 \times 8 \times k$ pixel matrices is applied to blocks $\mathbf{b}_f$. This coding process can be divided into two phases. During the first phase, the original block $\mathbf{b}_f$ is processed in such a way that it can be compressed with a limited number of bits at the expense of a tunable amount of distortion. In the second phase, an entropy coder converts each syntax element into a binary bit stream. Details of both procedures are given next.

3.3.1. Processing of the original signal

After interpolating the block $\mathbf{b}$ into $\mathbf{b}_f$, the three-dimensional matrix $\mathbf{b}_f$ is spatially predicted from the already-coded neighboring blocks. Since blocks $\mathbf{b}_f$ are coded in raster scan order, the pixels of the left, upper, and upper-right blocks are considered as references for spatial prediction (depending on the position of the current block within the image). For each view i, a 2-dimensional predictor $\mathbf{b}_p(.,.,i)$ is created from the reconstructed pixels of the same view belonging to the neighboring blocks. The prediction process inherits the interpolating equations of `Intra8x8` coding mode defined within the H.264/AVC standard [13]. This coding mode defines 9 possible prediction equations associated to 9 different spatial orientations. In the proposed coder, the coding process employs the same prediction mode m for all the views and chooses the orientation that minimizes a Lagrangian cost function analogous to the one proposed in [14]. The residual error $\mathbf{e} = \mathbf{b}_f - \mathbf{b}_p$ is then transformed by $8 \times 8 \times k$ separable 3D-DCT into the matrix $\mathbf{E}$.

The coefficients in $\mathbf{E}$ are then quantized by different uniform scalar quantizers, where the quantization steps depend on the positions (x, y, i) and are defined by a 3D quantization matrix called $\mathbf{Q}_{\boldsymbol{\Delta}}$. The basic idea behind such a 3D quantization matrix was to extend the 2D quantization matrix used in the JPEG compression standard [15] to a 3D matrix preserving the fact that the higher the frequency associated to the coefficient is, the bigger the quantization coefficient is. Several tests were performed in order to choose the quantization coefficients. Best results, in terms of reconstruction accuracy, were obtained defining the cube as a 3D extension of the array $\boldsymbol{\Delta} = [\Delta_z]_z = (8, 16, 19, 22, 26, 27, 29, 34)/8$. The coefficient $E(x, y, i)$ is quantized into the coefficient $E_q(x, y, i)$ using a quantization step $Q_\Delta(x, y, i)$ proportional to Δ_z, where $z = \max\{x, y, i\}$. More precisely, the quantization step $Q_\Delta(x, y, i)$ can be expressed as

$$Q_\Delta(x, y, i) = \left[0.69 \cdot 2^{\frac{QP}{6}} \cdot \Delta_z\right], \tag{3}$$

where $[\cdot]$ represents the rounding operator. The scaling factor $0.69 \cdot 2^{\frac{QP}{6}}$ has been derived from the equation that links the Quantization Parameter QP defined within the H.264 MVC standard with the final quantization step (QP assumes integer values between 0 and 51 inclusive). In this way, it was possible to equalize the adopted quantizers with those employed by H.264 MVC. The higher the QP is, the bigger the scaling factor is, the higher the quantization steps are and the lower the quantized data quality is. The quantization steps $Q_{\Delta}(x, y, i)$ can be grouped into a $8 \times 8 \times k$ quantization matrix $\mathbf{Q}_{\Delta}$, whose structure is depicted in Figure 8.

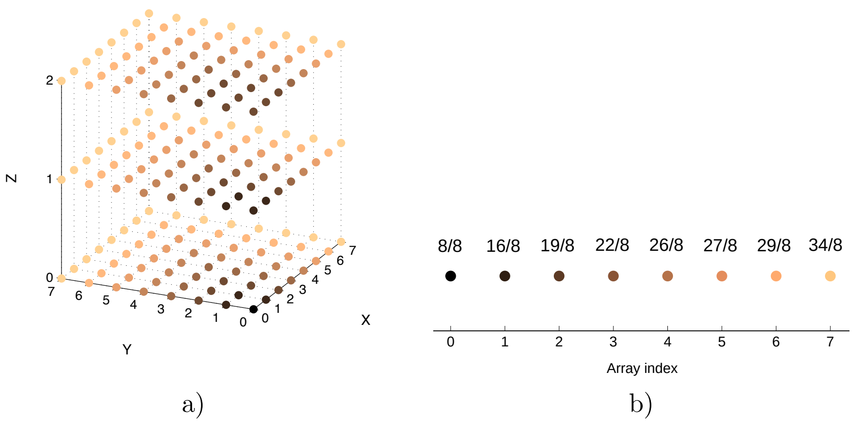


Figure 8: First three planes of the $8 \times 8 \times 8$ quantization matrix $\mathbf{Q}_{\Delta}$. The quantization coefficient for the DC coefficient is in position (0, 0, 0). Colors refer to the coefficient values.

Once the quantization process is completed, all of the quantized coefficients $\mathbf{E}_q$ have to be reordered into a 1D sequence. The reordering is supposed to facilitate entropy coding by placing low-frequency coefficients, which are more likely to be nonzero, before high-frequency coefficients. In this way it is possible to obtain long zero sequences at the end of the scanning, which is suitable for an efficient run-length coding. Like in most video coders, DC coefficients and AC coefficients were treated separately. The first ones were coded using a DPCM scheme, like in the classical JPEG approach [15]. For ACs, a scanning order has to be defined. A widely-adopted 2D solution to this issue is the "zig-zag" order of traditional DCT-based video coders [5]. In case of 3D data, different methods have been proposed in order to appropriately handle the data along the third dimension. In our case, the data we aim to compress do not refer to a video but to a multi-view image, and therefore, an *ad-hoc* solution has to be developed. Sgouros *et al.* [16] propose three different scanning orders for a quantized coefficient cube, choosing between them according to the standard deviation of the coefficients in the cube. Based on experimental results, we decided to adopt a solution similar to the one described in [17] and [18], where coefficients in the matrix $\mathbf{E}_q$ are scanned along planes perpendicular to the main diagonal. In each plane, the coefficients are then zig-zag scanned. Starting from this approach, Chan *et al.* [19] introduce the "*shifted complement hyperboloid*", which seems to provide better performances. In addition, DC coefficients are predicted via a classical DPCM scheme that considers the DC coefficients of neighboring blocks in order to reduce residual redundancy still present. The AC coefficients are coded by a classical run-length method which transforms the sequence of levels $E_q(x, y, i)$ into a sequence of couples $(run, level)$ as customary in most of the currently-adopted video coders. The sequences of couples $(run, level)$ and DPCM-coded DCs are then converted into a binary bit stream by a Huffman entropy coder.

3.3.2. Entropy coding of the syntax elements

The entropy coder converts spatial prediction modes and run-length couples into a binary bit stream. In order to reduce the size of the coded bit stream, we used an adaptive approach that chooses the variable length coding strategy according to the characteristics of the coded image among a set of multiple coding tables.

With respect to the prediction mode, the proposed scheme selects for the block $\mathbf{b}_f$ a different Huffman code according to the prediction modes of the upper and left neighboring blocks. In fact, the prediction modes of adjacent blocks are strongly correlated, and the prediction modes probabilities are significantly affected by their values. Therefore, different coding tables are available in order to code the most probable modes with the lowest number of bits, as it is done within the H.264/AVC standard [13]. More precisely, named A and B the upper and left neighboring blocks respectively, their prediction modes m_A and m_B are used to identify a probability distribution for the best prediction mode, similarly to the approach of [20]. The identified probability distribution infers a Huffman variable-length code that converts the prediction mode m into a binary string.

As for the couples $(run, level)$, *runs* are coded using a dedicated Huffman table, while *levels* are coded using 5 different coding tables. The first table is dedicated to DC prediction residuals, while the following 3 tables are used to code the following three coefficients in the scanning order. The fifth table is used for all the other coefficients.

In order to improve the compression performance of the proposed scheme, we have also adapted the structure of the CABAC arithmetic coder [21] to the $8 \times 8 \times 8$ blocks of quantized coefficients. Syntax elements are converted into binary strings, and each bit is then coded using the binary arithmetic coding engine defined within the standard H.264/AVC. More precisely, at first the coefficients of each $8 \times 8 \times 8$ block are scanned according to the previously-described zigzag order, and the number N_{nz} of non-zero coefficients is computed. Then, this value is mapped into a variable-length binary string using the Exp-Golomb binarization strategy adopted within the H.264/AVC standard (see [21] for a detailed description). The same binarization strategy is adopted to map runs and levels of run-length couples into binary arrays. These strings are then processed by the binary arithmetic coder which receives a binary digit and its context in input and maps them into an interval (see [21]). The preliminary results reported in Section 4 show that this solution significantly improves the coding performance despite the fact that in our implementation contexts have not been optimized, and other enhanced binarization techniques could be more effective.

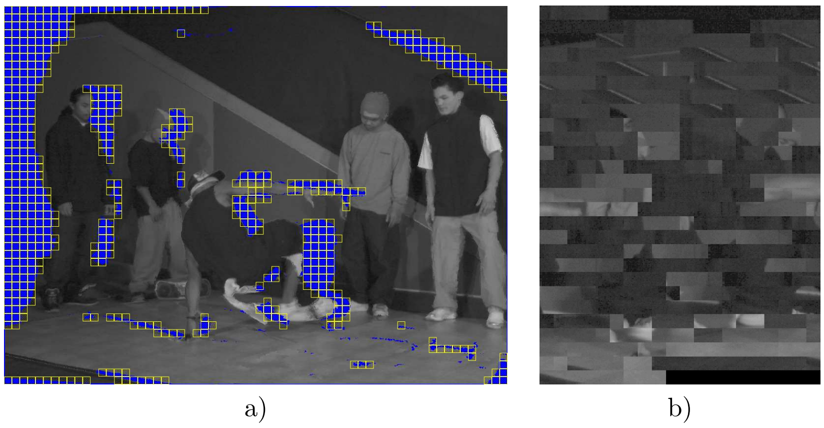


Figure 9: Reconstructed first view: a) Macroblocks corresponding to large holes and b) relative "Macroblock image". Blue pixels refer to occluded regions. Note that small holes are not filled using macroblocks.

At the decoder side, the performed operations are the same ones but in the inverse order: entropy decoding of quantized coefficients, de-quantization, inverse 3D-DCT transformation and 3D-unwarping of the reconstructed views. Before the unwarping process, a simple low-pass filter is applied to each image of the reconstructed image-stack, in order to reduce blocking artifacts due to the quantization process. However, in the unwarped images there are still areas to be filled corresponding to occlusions that can not be obtained from the reference view.

3.4. Hole-filling for occlusion areas

Holes in the unwarped images are filled with different techniques depending on their size. Small holes are simply interpolated on the basis of the surrounding available pixels and of local depth information, while larger ones are filled with 16×16 macroblocks from the original texture images. The filling mode is decided in the following way: at first, the missing samples are grouped into connected regions and the number of included samples is counted. Regions smaller or equal to a threshold t_h are interpolated while regions bigger than t_h pixels are filled by suitable macroblocks from the original images. Experiments have shown that good results are obtained with a threshold $t_h = 36$ pixels. Figure 9a shows the selected macroblocks of the first reconstructed image for the "breakdancers" dataset. In the next step, for each view we build a still image (called "Macroblock image") with all the macroblocks needed to fill it aligned on a regular grid (see Figure 9b). Finally, each"Macroblock image" is coded using H.264 Intra coding mode at the same quality of the rest of the image.

In order to reduce as much as possible the number of macroblocks needed, an inter-view filling process is performed while coding macroblocks. Specifically,

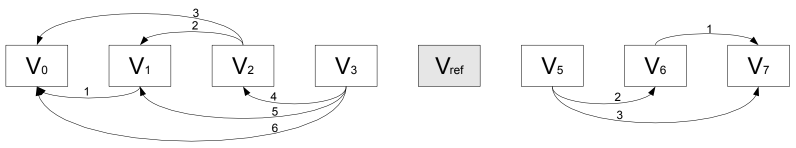


Figure 10: Architecture of the inter-view filling algorithm in the case of an 8-views dataset. The numbers on the arrows indicate the order in which the filling process is performed.

all the macroblocks of the first reconstructed image (V_0 in the examples) corresponding to large holes are filled from the source view (the resulting image will be denoted as V_{0f}). Then the missing macroblocks (large holes) of the second view (V_1) are warped to the first one in order to check if the new macroblocks in V_{0f} coming from its "Macroblock image" contribute to fill the holes of V_1. Moving to V_2, the same operation is performed orderly using the data from V_{1f} and V_{0f}. The process is then iterated for V_3 and eventually for other views (if $k > 8$) until V_{ref} is reached. This procedure is symmetrically performed on the right side of the reference view (in the examples views V_6, V_7 and V_8). Figure 10 shows the architecture of the proposed inter-view filling scheme. An example of the effective reduction of holes using the previously filled images is provided by progression a), b) and c) of Figure 11. Figure 12 instead shows how the proposed algorithm considerably reduces the number of macroblocks to be coded (for the "breakdancers" dataset the reduction is more than 50%).

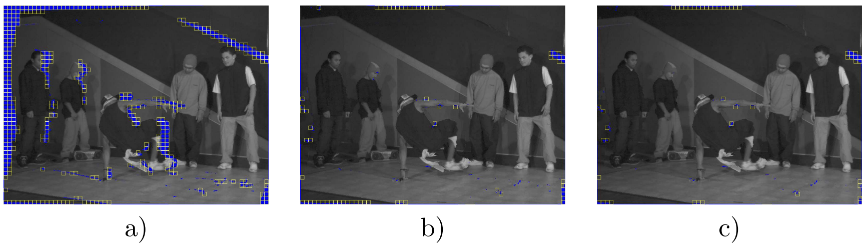


Figure 11: a) V_2 before the filling process, b) after hole-filling from second image and c) after hole-filling from both the second and the first image c). It is possible to note the strong reduction of the number of macroblocks that need to be encoded using the inter-view filling process.

4. Experimental Results

The proposed approach was tested on 8-views video sequences taken from a set of cameras pointing to the scene from linear and arc-shaped arrays. However, the proposed approach can be easily applied to systems with different number

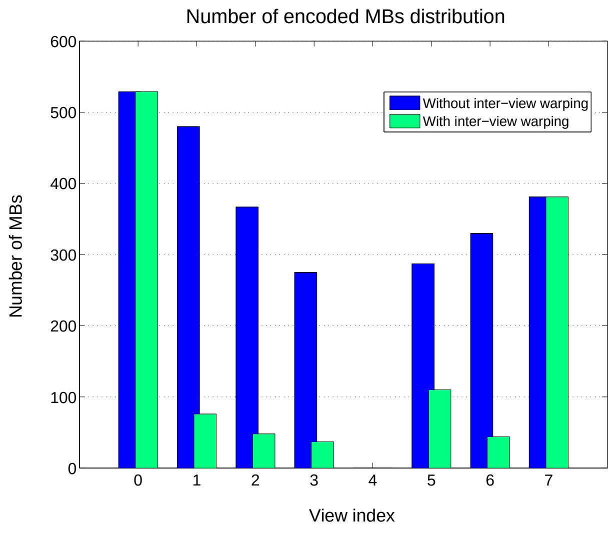


Figure 12: Number of macroblocks distribution with and without the inter-view warping filling process for the “breakdancers” dataset. In this case, the filling process provides a reduction of 53% on the total number of macroblocks to be encoded.

of views and different spatial configurations of the cameras. The comparisons were made on the luma component of each image.

The first dataset we used has been created from a synthetic 3D-model of the interior of a a modern kitchen (referenced as the “kitchen” dataset[3]). The use of a synthetic model allows to test the coding scheme without taking care of noise, camera distortions and other non-ideal features typical of real multi-view sequences. Moreover, ideal depth maps (i.e., nearly perfect geometry information) are available, therefore, all the warping issues related to unreliable depth information can be avoided. Note that, differently from H.264 MVC, our approach makes also use of depth information. However, in many free viewpoint schemes the depth maps should be transmitted anyway in order to be used in novel view reconstruction. Depth data can also be compressed very efficiently using H.264 MVC or *ad-hoc* compression schemes [22] and the literature reports that 10% to 20% of the total bitrate is usually enough for good quality depth data [23].

In order to check the performance of the proposed coding scheme with real world data, we used the “breakdancers” sequence [11], considering only its first frames. Note that in this case, depth information was computed from the video camera streams through computer vision algorithms and the accuracy is inevitably worse than that directly obtainable from synthetic models.

Rate-Distortion performances of the H.264 MVC coder and of the proposed

[3]Dataset and camera parameters are available at `http://lttm.dei.unipd.it/downloads/kitchen`

method are compared in Figures 13 and 14 for both the "kitchen" and "breakdancers" datasets, respectively. Two different setups were considered for the H.264 MVC coder. The first one, called "low complexity" setup (l.c.), includes coding parameters and features similar to the one used by the proposed approach. The second one, called "full complexity" setup (f.c.), enables all the features and coding techniques defined within the standard H.264 MVC. The initial l.c. configuration has been introduced in order to evaluate the efficiency of the scheme based on 3D warping and 3D-DCT with respect to standard motion compensation in exploiting inter-view redundancy. This configuration permits a "fair" comparison of the two methods without being biased by other additional features (e.g., the rate-distortion optimization algorithm and the arithmetic coding engine available in H.264 MVC). The adoption of such additional coding techniques allows the H.264 MVC coder, as well as our approach, to improve the R-D performances. This fact is highlighted by the rate-distortion performance of our solution with the adoption of the CABAC coding engine (reported in Figure 13 and 14 as well).

Table 1 summarizes the coding parameters used to generate the experimental results for both the H.264 MVC l.c. and f.c.

Feature, tool or setting	H.264 MVC l.c.	H.264 MVC f.c.
R-D Optimization	No	Yes
Deblocking Filter	No	Yes
Entropy Coder	CAVLC	CABAC
Block Size	8×8 fixed	Adaptive
Search Range	± 32	± 32
ME Accuracy	Quarter-pel	Quarter-pel
# Reference Pictures	4	4
Sequence Format String	A0*n{P0}	A0*n{P0}

Table 1: H.264 MVC coding parameters used to generate the comparison results.

The reported plots display the average PSNR value for all the reconstructed views as a function of the coded bitrate. It is possible to notice that for the "kitchen" sequence the proposed approach proves to be extremely competitive with respect to the H.264 MVC low-complexity coder, obtaining a quality increment of 1.2 dB in terms of PSNR at 0.027 bpp. An example of the resulting images is shown in Figure 15. For the "breakdancers" sequence instead, the performance of the scheme varies according to the coded bitrate. Figure 16 compares our approach with H.264 MVC low-complexity at 0.020 bpp: the proposed scheme improves the average PSNR value for the different views of approximately 2 dB. The improvement can also be appreciated from a subjective quality evaluation. The images related to our approach show a lower amount of artifacts. This fact is mainly due to the effectiveness of the warping operation in mapping corresponding pixels from different views with respect to the block-based compensation of H.264 MVC, which is limited by the block dimensions. At higher bitrates, the plots of the proposed approach and of the H.264 MVC l.c. progressively move closer and intersect at 0.036 bpp. This fact is mainly due to the inaccuracy of the depth maps, which introduces a significant amount of error in matching the pixels from different views. While at lower bitrates this error is compensated by a strong quantization, at high data rates it significantly affects the performance of the scheme since corresponding pixels could result displaced. The reported plots also display the performance of H.264 MVC full-complexity, where the CABAC coding engine and the rate-distortion optimization algorithm significantly improve the coding performance. As a drawback, the computational complexity sensibly increases. However, the adoption of the CABAC coder has proved to be effective for the proposed scheme as well. The plot in Figure 14 labelled as "`Proposed with CABAC`" shows that for the sequence "breakdancers" the quality of the reconstructed views at low bitrates improves up to 2 dB with respect to the approach labeled as "`Proposed`". Moreover, it is possible to obtain an improvement of 1 dB with respect to the H.264 MVC f.c. approach. This gain is mainly due to the strategy of specifying the number of non-zero coefficients in each block, which permits saving a significant amount of bits for high QP values. At high bitrates, this solution is less effective since many coefficients are different from zero, and extra bitrate is needed to specify N_{nz} in addition to the bits that code their positions. As for the sequence "kitchen", the gain is more evident at all the bitrates despite the fact that for small QP values the approach H.264 MVC f.c. provides the best compression performance (see Figure 13). The proposed scheme does not have a rate-distortion optimization algorithm like H.264 MVC, and therefore, its coding choices may result sub-optimal. Future investigations will be devoted to adapt the Rate-Distortion optimization routine of H.264 MVC to the proposed coder in order to further compress the coded signal at the expense of slightly lower perceptual quality. Moreover, the arithmetic coder can be utterly optimized by changing the binarization strategies and redesigning the structure of the contexts. These changes prove to be crucial for low QP values, which produce a significant amount of non-zero high-frequency coefficients. In these cases, the coefficients statistics require an accurate probability estimation to avoid an excessive increment of the coded bit stream.

Finally, Figures 17 and 18 report the bitstream partition in function of the reconstructed images' quality for both the datasets. The QP values reported on their x-axes refer to Equation (3). It is possible to notice that most of the bitrate is used to code coefficients obtained from the matrices $\mathbf{E}_q$. Because of a different camera arrangement, it is noticeable that a significant part of the bitrate for the sequence "kitchen" is used to code occlusions via the `Intra8x8` mode of H.264, which results less competitive than the 3D-DCT stack encoding. As a consequence, at low bitrates the PSNR gain over H.264 MVC low-complexity for the sequence "kitchen" is lower with respect to the gain for "breakdancers". As a matter of fact, at high data rates the PSNR gain is lower for "breakdancers", since depth maps are less reliable and the occluded areas are less extended with a consequent lower number of Intra coded macroblocks. Nevertheless, at low bitrates the proposed strategy proves to be advantageous in all the cases with

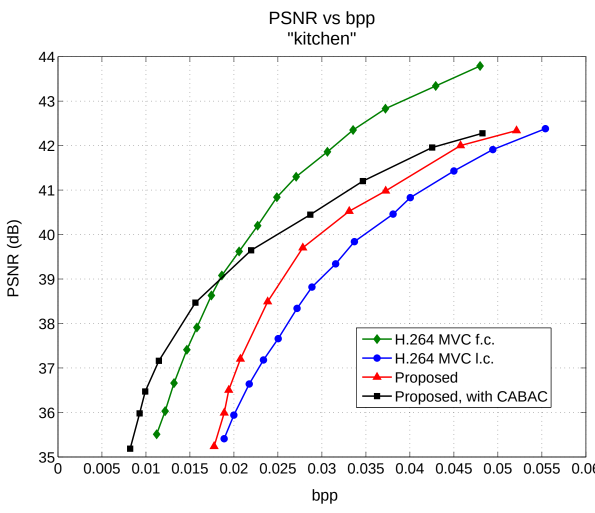


Figure 13: Coding performance comparison on “kitchen” dataset.

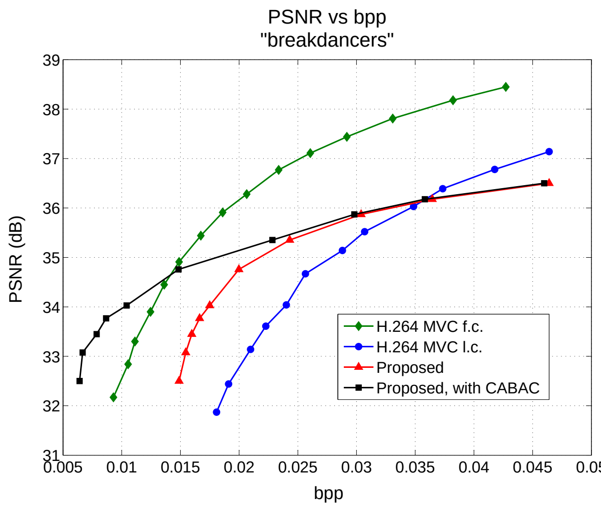


Figure 14: Coding performance comparison on “breakdancers” dataset.

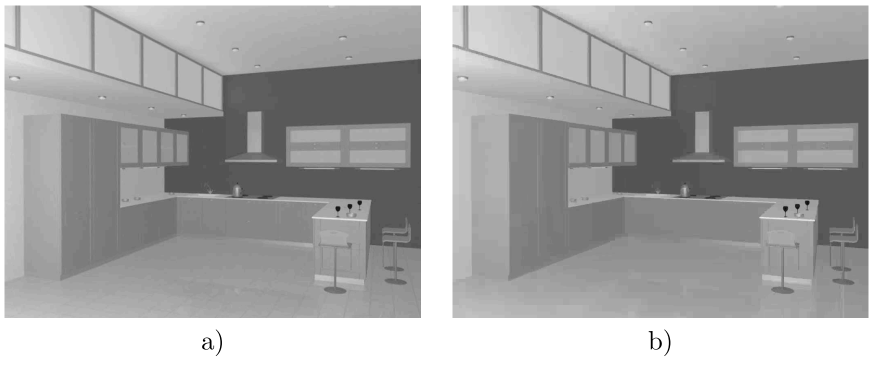


Figure 15: View V_5 of "kitchen" dataset at 0.027 bpp: a) Proposed scheme, b) H.264 MVC l.c. coder.

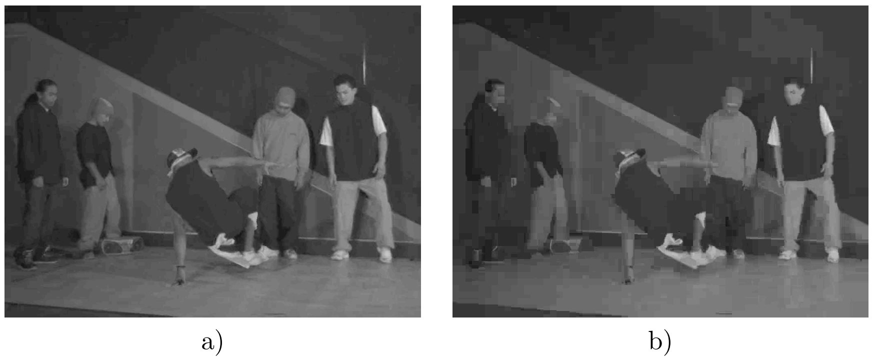


Figure 16: View V_5 of "breakdancers" dataset at 0.020 bpp: a) Proposed scheme, b) H.264 MVC l.c. coder.

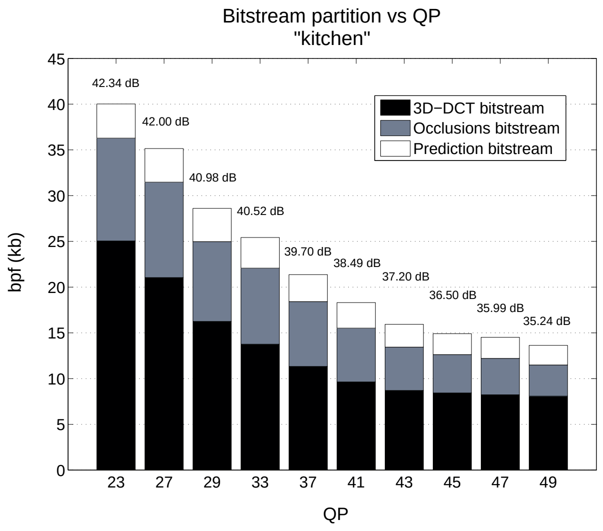


Figure 17: Bitstream partition of "kitchen" encoded dataset. Quality is expressed by PSNR over each bar.

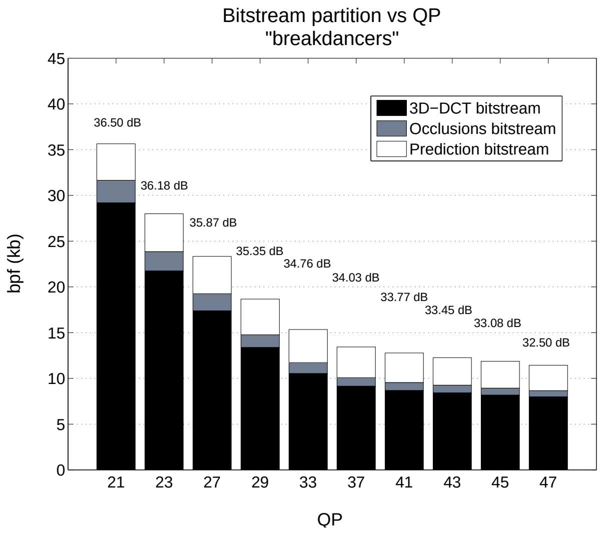


Figure 18: Bitstream partition of "breakdancers" encoded dataset. Quality is expressed by PSNR over each bar.

respect to the H.264 MVC low-complexity, which is less effective in exploiting the inter-view redundancy.

As expected, H.264 MVC with all the coding features enabled outperforms the proposed approach at high bitrates for both sequences. This fact is due to the use of effective R-D optimization techniques and to the adoption of an optimal context structure, which significantly improves the overall coding performances as the difference with respect to the H.264 MVC l.c. plots indicates. Future work will be devoted to applying such solutions to our approach in order to improve the CABAC performance at high bitrates and to avoid coding unnecessary coefficients.

5. Conclusions and future work

This paper introduces a novel coding scheme for multi-view images. Differently from most currently available solutions, the proposed approach is not an extension of any standard image or video compression tool, but it is instead based on an innovative algorithm explicitly targeted to multi-view data. Firstly 3D warping is used in place of the classical motion prediction stage in order to improve the motion compensation accuracy and to avoid the limitations given by fixed block sizes. Another original contribution is the use of a 3D-DCT transform in order to directly exploit inter-view redundancy in the transform stage instead of relying on the standard prediction and residual approach.

Finally, the critical issue of occluded areas has also been taken into account with an *ad-hoc* interpolation and coding strategy. Experimental results show how the proposed algorithm outperforms H.264 MVC at low bitrates in most practical configurations. However, the performance of the method at high bitrates depends on the accuracy of the available geometry information.

Further research will initially be focused on improving the performance of the transform stage as well as the entropy coding stage with the inclusion of features like adaptive block size, in-loop deblocking filtering, and R-D optimization that are still missing in the current implementation. Moreover, at high bitrates the coding gain can be utterly improved by optimizing the adopted context-adaptive binary arithmetic coder (i.e., modifying the current structure of contexts and the function that maps syntax elements into binary strings). Another important research direction aims at making the system more robust with respect to unreliable depth information. Optimized solutions for the compression of the depth data, possibly exploiting some of the techniques proposed in this paper, will also be included in the proposed framework. Another improvement will be the inclusion of a motion prediction stage in order to exploit this scheme also in the compression of multi-view video. Finally, scalability issues will also be considered with particular focus on view scalability. In order to achieve this target, considering also the very promising results of [10], we will also take into account the possibility of replacing the 3D-DCT with more flexible transforms like the 3D/4D wavelet.

References

[1] Y.-S. Ho, and K.-J. Oh, "*Overview of Multi-view Video Coding*," Proc. of the 14th International Workshop on Systems, Signals and Image Processing (IWSSIP 2007) and 6th EURASIP Conference focused on Speech and Image Processing, Multimedia Communications and Services (EC-SIPMCS 2007), Maribor, Slovenia, pp. $5-12$, Jun. $27-30$, 2007.

[2] P. Zanuttigh, N. Brusco, D. Taubman, and G.M. Cortelazzo, "*A Novel Framework for the Interactive Transmission of 3D Scenes*," Signal Processing: Image Communication, vol. 21, no. 9, pp. $787-811$, Oct. 2006.

[3] L. Li, and Z. Hou, "*Multiview video compression with 3D-DCT*," Proc. of ITI 5th International Conference on Information and Communications Technology (ICICT 2007), Dhaka, Bangladesh, pp. $59-61$, Dec. $16-18$, 2007.

[4] ISO/IEC MPEG & ITU-T VCEG, "*Joint Draft 8.0 on Multiview Video Coding*," JVT-AB204, Jul. 2008.

[5] H. Schwarz, D. Marpe, and T. Wiegand, "*Overview of the Scalable Video Coding Extension of the H.264/AVC Standard*," IEEE Trans. on Circuits and Systems for Video Technology, vol. 17, no. 9, pp. $1103-1120$, Sept. 2007.

[6] X. Guo, Y. Lu, F. Wu, and W. Gao, "*Inter-View Direct Mode for Multiview Video Coding*," IEEE Trans. on Circuits and Systems for Video Technology, vol. 16, no. 12, pp. $1527-1532$, Dec. 2006.

[7] F. Dufaux, M. Ouaret, and T. Ebrahimi, "*Recent Advances in Multiview Distributed Video Coding*," SPIE Defense and Security Symposium (DSS 2007), Orlando, Florida, USA, Apr. $9-13$, 2007.

[8] E. Martinian, A. Behrens, J. Xin, A. Vetro, and H. Sun, "*Extensions of H.264/AVC for Multiview Video Compression*," IEEE International Conference on Image Processing (ICIP), pp. $2981-2984$, Oct. 2006.

[9] J. Li, J. Takala, M. Gabbouj, and H. Chen, "*Variable temporal length 3D DCT-DWT based video coding*," Proc. of the 2007 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2007), Xiamen, China, pp. $506-509$, Nov. 28-Dec. 1, 2007.

[10] W. Yang, Y. Lu, F. Wu, J. Cai, K.N. Ngan, and S. Li, "*4-D Wavelet-Based Multiview Video Coding*," IEEE Trans. on Circuits and Systems for Video Technology, vol. 16, no. 11, pp. $1385-1396$, Nov. 2006.

[11] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "*High-quality video view interpolation using a layered representation*," ACM SIGGRAPH and ACM Trans. on Graphics, Los Angeles, CA, pp. $600-608$, Aug. 2004.

[12] R.I. Hartley, and A. Zisserman, "*Multiple View Geometry in Computer Vision*," 2nd edition, Cambridge University Press, 2004.

[13] T. Wiegand, "*Version 3 of H.264/AVC,*" JVT-L012, Joint Video Team of ISO/IEC MPEG & ITU-T VCEG 12th meeting, Redmond, WA, USA, Jul. 17 – 23, 2004.

[14] T. Wiegand, and B. Girod, "*Lagrange Multiplier Selection in Hybrid Video Coder Control*," IEEE International Conference on Image Processing (ICIP 2001), Thessaloniki, Greece, Sept. 2001.

[15] G.K. Wallace, "*The JPEG still picture compression standard*," Communications of the ACM, vol. 34, no. 4, pp. 30 – 44, 1991.

[16] N.P. Sgouros, S.S. Athineos, P.E. Mardaki, A.P. Sarantidou, M.S. Sangriotis, P.G. Papageorgas, and N.G. Theofanous, "*Use of an adaptive 3D-DCT scheme for coding multiview stereo images*," International Symposium on Signal Processing and Information Technology, pp. 180 – 185, Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005.

[17] B.L. Yeo, and B. Liu, "*Volume Rendering of DCT-Based Compressed 3D Scalar Data*," IEEE Transactions on Visualization and Computer Graphics, vol. 1, no. 1, pp. 29 – 43, Mar. 1995.

[18] T. Frýza, "*Properties of Entropy Coding for 3D DCT Video Compression Method*," Radioelektronika, 17th International Conference, Brno, Czech Republic, Apr. 24 – 25, 2007.

[19] Raymond K.W. Chan, and M.C. Lee, "*3D-DCT Quantization as a Compression Technique for Video Sequences*," VSMM, pp. 188, 1997 International Conference on Virtual Systems and MultiMedia, 1997.

[20] S. Milani, "*A Belief-Propagation Based Fast Intra Coding Algorithm for the H.264/AVC FRExt coder*," Proc. of the 16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, Aug. 25 – 29, 2008.

[21] D. Marpe, H. Schwarz, and T. Wiegand, "*Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard*," IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 620 – 636, Jul. 2003.

[22] P. Zanuttigh, and G.M. Cortelazzo, "*Compression of depth information for 3D rendering*," Proc. of 3DTV Conference, pp. 1 – 4, May 2009.

[23] H.M. Ozaktas, and L. Onural, "*Three-Dimensional Television: Capture, Transmission, Display*," Series: Signals and Communication Technology, Springer, 2008.

List of Tables

List of Figures