System Architecture for Free-Viewpoint Video and 3D-TV

Yannick Morvan, Dirk Farin and Peter H.N. de With, *Fellow*, IEEE

Abstract — *This paper presents a system architecture of an acquisition, compression and rendering system for 3D-TV and free-viewpoint video applications. We show that the proposed system yields two distinct advantages. First, it achieves an efficient compression of 3D/multi-view video by extending a standard H.264 encoder such that near backward compatibility is retained. Second, the proposed system can efficiently compress both 3D-TV and free-viewpoint multi-view video datasets using the single proposed system architecture.*

Index Terms — 3D-TV, free viewpoint video, multi-view video coding, 3D video coding.

I. INTRODUCTION

The successful introduction of Blu-ray disk as a medium for distribution HDTV movies and games to the consumer establishes, together with the increasing acceptance of large high-definition LCD displays, a firm basis for various HDTV applications. In this paper, we concentrate on the next generation of television systems, namely 3D-TV. This involves the inclusion of one or more extra video signals besides a regular monocular video signal, such as a depth signal and/or additional views of the same scene. While various formats for 3D-TV video signals are discussed later, we concentrate in this paper on multi-view video only.

A 3D video is typically obtained from a set of synchronized cameras, which are capturing the same scene from different viewpoints (multi-view video). This technique enables applications such as free-viewpoint video and 3D-TV. First, a *free-viewpoint video player* provides the ability for users to interactively navigate and select a viewpoint in the video scene. A free-viewpoint video player is therefore similar to a 3D computer graphics rendering engine, albeit based on natural images. Second, a 3D or multi-view display allows the viewer to perceive the depth of the scene. A multi-view display is a LCD panel onto which a lenticular sheet is accurately superimposed.

[1] Yannick Morvan was with the Eindhoven University of Technology, The Netherlands. He is now with Philips Healthcare, Best, The Netherlands (e-mail: ymorvan@free.fr).

Dirk S. Farin, was with Eindhoven University of Technology, The Netherlands. He is now with the Bosch Security Systems Gmbh, Hilderseim, Germany (e-mail: dirk.farin@gmail.com).

Peter H.N. de With is with the Eindhoven University of Technology, Eindhoven, The Netherlands (e-mail: p.h.n.de.with@tue.nl) and CycloMedia Technology, Waardenburg, The Netherlands.

The lenticular sheet projects several views of the same scene in different directions. By observing slightly different views of the scene, the human brain integrates these views into a 3D representation of the scene. Because both applications rely on multiple views of the scene, both technologies do not exclude each other and can be integrated into a single 3D video system. To enable 3D-TV or free-viewpoint video applications, several 3D video systems have been introduced. They can be classified into three classes with respect to the amount of employed 3D geometry.

A. Three dimensional video systems layout

A first class of 3D video systems is based on multiple texture views of the video scene, called *N-texture* representation format [1]. One advantage of the N-texture representation format is that no 3D geometric description of the scene is required. Therefore, because 3D geometry is not used, this 3D video format allows a simple video processing chain at the encoder. However, such a 3D video representation format involves a high complexity decoder for the following reason. A multi-view display supports a varying number of views at the input, which makes it impractical to prepare these views prior to transmission. Instead, intermediate views should be interpolated from the transmitted reference views at the decoder where the display characteristics are known. To obtain high-quality interpolated views, a 3D geometric description of the scene is necessary, thereby involving computationally expensive calculations at the receiver side.

A second class of 3D video systems relies on a partial 3D geometric description of the scene [2]. The scene geometry is typically described by a *depth map*, or *depth image*, that specifies the distance between a point in the 3D world and the camera. Typically, a depth image is estimated from two images by calculating the parallax motion of pixels between the views. Using depth images, new views can be subsequently rendered or synthesized using a Depth Image Based Rendering (DIBR) algorithm. Considering a 3D-TV application, it is assumed that the scene is observed from a narrow field of view (short baseline distance between cameras). As a result, a combination of only one texture and one depth video sequence is sufficient to provide appropriate rendering quality (*1-texture/1-depth*). The *1-texture/1-depth* approach was recently standardized by the *Part 3* of the *MPEG-C* video specifications [3]. However, considering a video scene with rich 3D geometry, rendered

Contributed Paper
Manuscript received April 15, 2008

virtual views typically show occluded regions that were not covered by the reference camera. A third class of 3D video systems addresses this problem by combining the two aforementioned classes (*N-texture* and *1 texture/1-depth*) by using one depth image for each texture image, i.e. *N-texture/N-depth* [4]-[6]. The advantages of this approach are many-fold. First, as previously highlighted, the problem of occluded regions can be addressed by combining multiple reference images that cover all regions seen by the virtual camera. Second, the N-texture/N-depth representation format is compatible with different types of multi-view displays that support a varying number of views. More specifically, because 3D geometry data is transmitted to the decoder, an arbitrary number of synthetic views that corresponds to the display characteristics can be interpolated. A final advantage is that the N-texture/N-depth representation format provides a natural extension to the 1-texture/1-depth representation format. Therefore, this approach allows a gradual transition from an already standardized technique (MPEG-C part 3) to the next generation of 3D video systems. Because of these advantages, we have adopted in this paper the N-depth/N-texture video representation format.

B. Review of N-depth/N-texture 3D video architecture

Let us now review the previously discussed N-depth/N-texture systems architecture in more detail. It is evident that for the N-signal case, the construction of consistent depth signal forms a crucial part in the system. Furthermore, in order to avoid expensive transmission of N signals in parallel, an efficient compression technique is indispensable.

1) Multi-view depth estimation

The problem of multi-view depth estimation has been intensively investigated in the computer-vision community [7]. The principal problem is to localize corresponding pixels (point-correspondences) in the multiple views that represent the same 3D scene point. By finding point-correspondences, the depth information can be derived by triangulation for each pixel, resulting in a so-called *dense depth image*. Since the N-texture/N-depth representation was adopted, a depth-image sequence should be estimated for each view. In the context of 3D-TV, it was recently proposed to estimate a depth image for each camera sequentially [8]. Let us first discuss this proposal in more detail.

In a first step, the set of N views is divided into N-1 pairs of (left and right) images and the image pairs are aligned with respect to each other, i.e. *rectified.* This first step, called image rectification, is the process of transforming two images such that their epipolar lines are horizontal and parallel. As a result, point-correspondences are located on the same row-index of both images and the search of point-correspondences can be performed along horizontal raster image scan-lines. In a second step, a disparity value is estimated for each pixel of the rectified image pairs. The estimation of the disparity value is performed by searching a block of pixels in the right image most similar to the considered block in the left reference image. Therefore, the problem of point-correspondence search is similar to the motion-estimation problem, albeit performed across two views sampled simultaneously and not for two consecutive frames in time. The second step of the algorithm results in a disparity map that indicates the horizontal motion (disparity) between the two corresponding pixels, where the disparity corresponds to the inverse of the depth (caused by parallax motion). In a third step, the disparity map is converted into a depth image using camera calibration parameters. Finally, disparity maps are inversely rectified, so that the depth images are represented in the original (non-rectified) coordinate system of the cameras. Additionally, to avoid inaccurate and noisy depth estimates, a post-processing step based on image segmentation ensures smooth depth variations for the same object.

This above technique has multiple disadvantages. First, while multiple views of the scene are available, only two views are employed simultaneously. This results in a less-constrained point-correspondence search and thus noisy depth estimates. Second, the approach involves multiple unnecessary pre- and post-processing operations: (1) image-pair rectification, (2) disparity-to-depth conversion and (3), inverse image rectification.

To circumvent these issues, we propose in this paper a method that starts with a concept that is based on omitting the undesired image rectification pre- and post-processing steps. As a bonus of our proposal, the technique allows the estimation of depth images using all available views simultaneously, so that a more accurate view interpolation is facilitated.

2) Multi-view video compression

A major problem when dealing with multiple depth and texture video signals is the large amount of data to be encoded, decoded and rendered. For example, an independent transmission of 8 views of the ``Breakdancers'' sequence requires about 10 Mbit/s and 1.7 Mbit/s with a PSNR of 40 dB for the texture and depth data, respectively. Therefore, efficient multi-view depth and texture compression algorithms are highly beneficial. In a typical multi-view acquisition system, the acquired views are highly correlated. As a result, a coding gain can be obtained by exploiting the interview redundancy between neighboring cameras. So, in our system, besides strict adjacent camera view redundancy, we also exploit neighboring camera view redundancy in a broader sense. Bearing the above in mind, two different approaches for predictive coding of views have been investigated.

A first inter-view prediction technique [9] uses *block-based disparity-compensated prediction* scheme. Besides compatibility with H.264 coding, a primary advantage

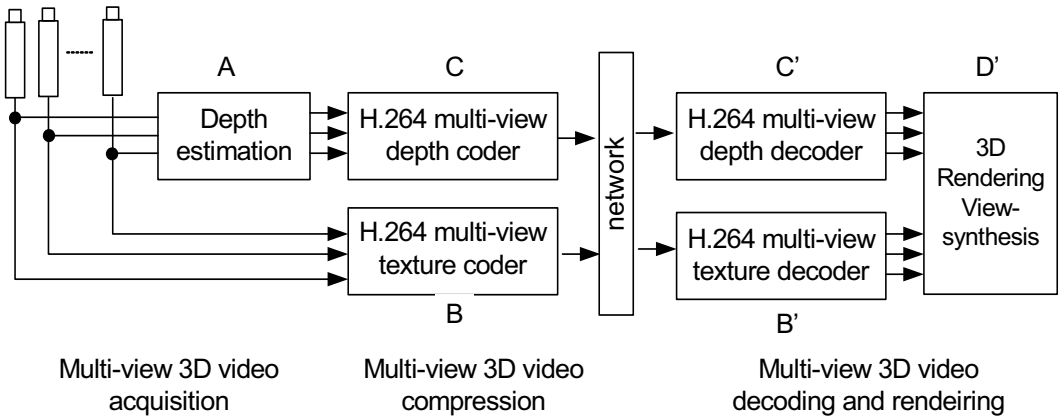


Fig. 1. Overview of the proposed 3D video processing system that includes (A) the acquisition of 3D video (depth estimation), (B/B')-(C/C') the coding and decoding of the multiple texture and depth images sequences and (D) the rendering sub-system.

of this approach is that disparity-compensated prediction does not rely on the geometry of multiple views, so that camera calibration parameters are not required. However, in the case that the baseline distance between cameras is large, it has been reported [9] that a disparity-compensated prediction scheme yields a limited coding gain over independent coding of the individual views. An explanation for this limited performance is that the translational motion model employed by the disparity-compensated prediction scheme is not sufficiently accurate to predict the motion of objects with different depth.

A second view-prediction scheme [4]-[22] is based on an image rendering algorithm, i.e. *DIBR-based prediction*. The synthesis algorithm employs a reference texture and depth image as input data. The advantage of the DIBR-based prediction is that the views can be better predicted even when the baseline distance between the reference and predicted cameras is large, thereby yielding a high compression ratio. However, as opposed to the previous approach, the multi-camera acquisition system needs to be fully calibrated prior to the capture session. Additionally, a depth image should be estimated for the central reference view. Because estimated depth images may be inaccurate, the view-prediction quality may also be reduced.

Important requirements of the view prediction algorithm are that (a) it should be robust against inaccurately estimated depth images, and (b) an efficient compression should be obtained for various baseline distances between cameras. As discussed above, both presented view-prediction algorithms have their limitations and cannot be used under variable capturing conditions. Therefore, our novel strategy is to use both algorithms selectively on an image-block basis, depending on their coding performance. Having the outcomes of both approaches available, an *a-posteriori* decision is made for selection of the preferred prediction algorithm.

We propose a H.264-based multi-view encoder that employs both prediction techniques. The most appropriate prediction technique is selected for each image block independently, using a rate-distortion criterion. The first view-prediction algorithm is based on block-based disparity-compensated prediction. The second view-prediction technique works by rendering an image as seen by the predicted camera. The algorithm is designed such that it is uniform for the signal type and can handle both multi-view depth and texture data in a multiplexed fashion.

The remainder of this paper is organized as follows. Section II details each sub-system of the proposed 3D video processing system, namely depth estimation, image rendering and multi-view video coding. Experimental results, which are provided in Section III, address a comparison between the prediction techniques and show rate-distortion curves of the final performance. The paper concludes with Section IV.

II. 3D VIDEO SYSTEM ARCHITECTURE

Figure 1 shows the system architecture, which is composed of a depth-estimation sub-system (A), a multi-view video coder/decoder (B/B'), a multi-view depth video coder/decoder (C/C') and a 3D-video rendering engine (D). In the following, we address each of these sub-systems.

A. Depth Estimation

We now present a method that enables the estimation of depth images using all views simultaneously without performing pre-preprocessing image rectification steps. The proposed depth-estimation algorithm can be summarized as follows. In a first step, given a selected pixel, we compute for each depth candidate the corresponding pixel position in the neighboring views. To compute the pixel positions in the neighboring views, the 3D position of the selected pixel is calculated using a back-projection operation [10]. Note that this back-projection operation is based on internal and external camera parameters that indicate the location and orientation of cameras. Next, the (candidate) pixel positions in neighboring views are calculated by projecting the 3D point onto the image planes of the neighboring cameras. Therefore, by exploiting camera parameters, multiple views can be employed simultaneously. Because all views are employed simultaneously, a reliable similarity value can be

obtained. The similarity measure, i.e. matching cost, between the selected pixel and the pixels in neighboring views is measured using the Sum of Absolute Differences (SAD) and the matching cost is stored in a so-called Depth Space Image (DSI) structure [11]. The DSI structure is a 2D table of size $d_{max} \times w$ that contains the matching cost of all pixels at all possible candidate depth values for all pixels along a scanline. Here, d_{max} and w correspond to the number of possible depth values and the image width, respectively. For example, the entry *DSI (x, d)* contains the sum of SADs between a reference pixel $p_1=(x_1, y_1, 1)$ and the candidate pixels of all neighboring views at depth d. Additionally, to enforce a smooth variation of depth pixels, a so-called *smoothness term* is integrated into the DSI as a transition cost between each entry of the DSI. Therefore, considering the DSI structure as a graph, matching costs and transition costs correspond to nodes and edges, respectively (see Figure 2). The objective of the algorithm is then to calculate the depth sequence that minimizes the total cost associated with the path through the DSI. This minimum-cost path can be calculated using a dynamic programming algorithm. Because the optimization is performed independently for each scanline, horizontal line-based artifacts may be visible in the depth image. To avoid these artifacts, we introduce an inter-scanline penalty cost that penalizes fast variations of depth estimates across scanlines. In practice, the inter-line penalty cost is similar to the smoothness term introduced in the previous paragraph, albeit in the vertical direction of the image. The described depth-estimation procedure is re-iterated for each view so that one depth-image sequence is obtained for each camera. The advantage of the algorithm is that it provides an efficient computation because it is essentially line-based. Moreover, the inter-scanline penalty reduces the typical artifacts of scanline algorithms.

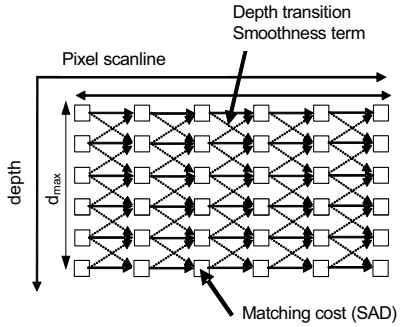


Fig 2. The DSI structure contains the matching cost of all pixels for each depth value. The sequence of disparities that yields the minimum cost-path can be calculated using a dynamic programming algorithm. In this example, the smoothness term allows only one unity of depth change between two consecutive pixels is allowed for consistency. Note that another admitted path can also be used.

B. Depth Image Based Rendering (DIBR)

A single texture image and a corresponding depth image are sufficient to synthesize novel views at an arbitrary position. Let us consider a point in the 3D world $\boldsymbol{P}_w=(X_w, Y_w, Z_w)$ which is captured by two cameras and is projected onto the reference and the synthetic image at homogeneous pixel positions $\boldsymbol{p}_1=(x_1, y_1, 1)$ and $\boldsymbol{p}_2=(x_2, y_2, 1)$, respectively. We assume that the first reference camera is located at the coordinate-system origin and looks along the Z-dimension. The location and orientation of the predicted camera are described by its camera center, indicated with the column vector $\boldsymbol{C}_2$, and the rotation matrix $\boldsymbol{R}_2$. The pixel positions $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ in both image planes are defined by

$$\lambda_1 \boldsymbol{p}_1 = \boldsymbol{K}_1.(X_w, Y_w, Z_w)^T, \quad (1)$$

$$\lambda_2 \boldsymbol{p}_2 = \boldsymbol{K}_2 \boldsymbol{R}_2.(X_w, Y_w, Z_w)^T - \boldsymbol{K}_2 \boldsymbol{R}_2 \boldsymbol{C}_2, \quad (2)$$

where $\boldsymbol{K}_1$, $\boldsymbol{K}_2$ represent the 3×3 intrinsic parameter matrix of the corresponding cameras and λ_1, λ_2 some positive scaling factors [10]. Because the matrix $\boldsymbol{K}_1$ is upper-triangular and $\boldsymbol{K}_1(3, 3)=1$, the scaling factor λ_1 can be specified in this particular case by $\lambda_1 = Z_w$. From (1), the 3D position of the original point $\boldsymbol{P}_w$ in Euclidean coordinates can be written as

$$(X_w, Y_w, Z_w)^T = \boldsymbol{K}_1^{-1} \lambda_1 \boldsymbol{p}_1 = \boldsymbol{K}_1^{-1} \boldsymbol{Z}_w \boldsymbol{p}_1. \quad (3)$$

Finally, we obtain the pixel position $\boldsymbol{p}_2$ in the synthetic image by substituting (3) into (2) so that

$$\lambda_2 \boldsymbol{p}_2 = \boldsymbol{K}_2 \boldsymbol{R}_2 \boldsymbol{K}_1^{-1} Z_w \boldsymbol{p}_1 - \boldsymbol{K}_2 \boldsymbol{R}_2 \boldsymbol{C}_2. \quad (4)$$

Equation (4) constitutes the image-warping equation [12] that enables the synthesis of a virtual view from the original reference view and its corresponding depth image.

One issue of the previously described method is that input pixels $\boldsymbol{p}_1$ of the reference view may not always be mapped to a pixel $\boldsymbol{p}_2$ at an integer pixel position. A second difficulty is that multiple original pixels can be projected onto the same pixel position in the synthetic view. For example, a foreground pixel can occlude a background pixel in the synthetic view, which is resulting in overlapping pixels. Additionally, some regions in the synthetic view are not visible from the original viewpoint, which results in holes in the synthetic image. To address the aforementioned issues, we introduce a variant of the relief texture mapping technique [13], which we have modified to the geometry of multiple views.

The guiding principle of the relief texture algorithm is to factorize the 3D image-warping equation into a combination of 2D texture-mapping operations. One well-known 2D texture-mapping operation corresponds to a perspective projection of planar texture onto a plane defined in a 3D world. Mathematically, this projection can be defined using homogeneous coordinates by a 3×3 matrix multiplication, and corresponds to a homography transform between two images. The advantage of such a transformation is that a hardware implementation of this function is available in most of the Graphic Processor Units (GPU). Therefore, when using a GPU, processing time is dramatically reduced.

Let us now factorize the warping function, so as to obtain a homography transform in the factorization. From

(4), it follows that

$$\frac{\lambda_2}{Z_w}\boldsymbol{p}_2 = \boldsymbol{K}_2\boldsymbol{R}_2\boldsymbol{K}_1^{-1}\cdot(\boldsymbol{p}_1 - \frac{\boldsymbol{K}_1\boldsymbol{C}_2}{Z_w}). \quad (5)$$

Analyzing this equation, it can be seen that the first factor $\boldsymbol{K}_2\boldsymbol{R}_2\boldsymbol{K}_1^{-1}$ is equivalent to a 3×3 matrix and represents the desired homography transform. Let us now analyze the second factor of the factorized equation, i.e. $(\boldsymbol{p}_1 - \boldsymbol{K}_1\boldsymbol{C}_2 / Z_w)$. This second factor projects the input pixel $\boldsymbol{p}_1$ onto an intermediate point $\boldsymbol{p}_i=(x_i,\ y_i,\ 1)$ that is defined by

$$\lambda_i\boldsymbol{p}_i = \boldsymbol{p}_1 - \frac{\boldsymbol{K}_1\boldsymbol{C}_2}{Z_w}, \quad (6)$$

where λ_i defines a homogeneous scaling factor. It can be seen that this last operation performs the translation of the reference pixel $\boldsymbol{p}_1$ to the intermediate (synthetic) pixel $\boldsymbol{p}_i$. The translation vector can be expressed in homogeneous coordinates by

$$\lambda_i\begin{pmatrix}x_i\\y_i\\1\end{pmatrix} = \begin{pmatrix}x_1-t_1\\y_1-t_2\\1-t_3\end{pmatrix} \text{ with } (t_1,t_2,t_3)^T = \frac{\boldsymbol{K}_1\boldsymbol{C}_2}{Z_w}.$$

Written in Euclidean coordinates, the intermediate pixel position is defined by

$$x_i = \frac{x_1-t_1}{1-t_3}, \qquad y_i = \frac{y_1-t_2}{1-t_3}.$$

It can be seen that this result basically involves a 2D texture-mapping operation, which can be further decomposed into a sequence of two 1D transformations. In practice, these two 1D transformations are performed first along rows, and then along columns. This class of warping methods is known as scanline algorithms [14]. An advantage of this additional decomposition is that a simpler 1D texture-mapping algorithm can be employed (as opposed to 2D texture-mapping algorithms). For the padding of occluded pixels, we have employed simple heuristic techniques, where occluded pixels are padded by adjacent background pixels [15]. As an algorithmic summary, the synthesis of the view using relief texture mapping is performed as follows.

- Step 1: Perform warping of reference texture along horizontal scanlines.
- Step 2: Perform warping of the (already horizontally-warped) texture along vertical scanlines.
- Step 3: Compute the planar texture projection of the intermediate image using the homography transform defined by $\boldsymbol{K}_2\boldsymbol{R}_2\boldsymbol{K}_1^{-1}$ (exploit the GPU for fast computing).

C. Multi-view texture video coding

In this section, we describe our novel H.264 coding architecture for multi-view coding that employs a block-based motion-prediction scheme and the previously introduced relief texture-mapping DIBR technique.

In this sub-section, we present the coding system as covered by the concept introduced in Section I-*B.2*, which was based on combining two image prediction techniques.

One possible approach to integrate both prediction techniques could be to select the best prediction for each block. A disadvantage of this approach is that the DIBR-based prediction error is not ensured to be minimal, leading to lower compression efficiency. An alternative technique is to employ a combination of two predictors: (a) a DIBR-based predictor subsequently followed by (b) a disparity-compensated predictor. The resulting system becomes as follows. First, we provide an approximation of the predicted view using relief texture mapping and, second, we refine the DIBR-based prediction using block-based motion prediction. In the refinement stage, the search for matching blocks is performed in a region of limited size, e.g. 32×32 pixels. In contrast to this, the disparity between two views in the "Ballet" sequence can be as high as 50 pixels. Figure 3 shows an overview of the described coding architecture.

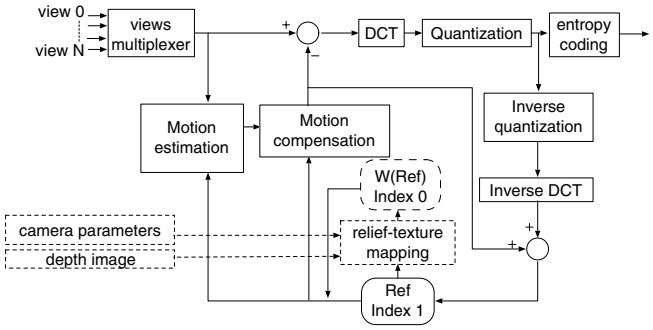


Fig. 3. Architecture of an H.264 encoder, that adaptively employs a block-based disparity-compensated prediction or a DIBR-based prediction followed by a prediction-refinement. The reference frame and the corresponding synthetized reference frame are denoted Ref and W(Ref), respectively.

Let us now discuss the prediction structures employed in the proposed multi-view encoder. Considering the free-viewpoint video application, random access to neighboring views after coding is necessary, so that an appropriate coding structure should be adopted. To exploit both spatial (i.e. inter-view) and temporal redundancy (inter-frame), we propose to use predefined views as a spatial reference from which neighboring views are predicted. Observing the coding structure of Figure 4, it can be seen that temporal correlation is exploited only with respect to the two central reference views (in the vertical direction of Figure 4). Similarly, only non-central views exploit the spatial inter-view redundancy (in the horizontal direction of Figure 4). For this reason, by exploiting an appropriate mixture of temporal and spatial prediction, views along the chain of cameras can be randomly accessed. Therefore, we have adopted the coding structure from Figure 4 to perform multi-view coding. In contrast with the proposed coding structure, an independent compression of views (simulcast coding) does not allow a random access to an arbitrary selected view. In such a case, all views (N depth and N

texture video) should be decoded in parallel, involving a very high decoding computational complexity.

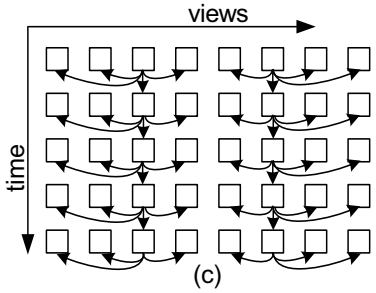


Fig. 4 Coding structure where only two central views exploit the temporal redundancy. These two central views are then used as a reference for inter-view prediction.

D. Multi-view depth video coding

Similar to the compression of multi-view texture video, we propose to encode the multi-view depth video using a DIBR view-synthesis prediction algorithm. The proposed depth-prediction technique works by synthesizing/ computing the depth of 3D points based on the reference depth image [5]. As performed for multi-view texture video, the depth-image prediction is also based on the relief texture algorithm and inserted into the H.264 encoder as described by Figure 3.

At first glance, the multi-view texture coding and multi-view depth coding architectures look identical. However, there is one key difference between them. In the multi-view texture coding case, the depth signal should be already available to the DIBR prediction scheme. Conversely, for multi-view depth coding, the texture need not to be available.

III. EXPERIMENTAL RESULTS

For evaluating the performance of the coding algorithm, experiments were carried out using the texture and depth "Ballet" and "Breakdancers" multi-view sequences. The presented experiments investigate the impact of the DIBR-based prediction accuracy on the rate-distortion performance, using the prediction structure depicted by Figure 4. For each presented rate-distortion curve, we perform the multi-view compression of texture and depth under two different conditions.

1. The prediction of views is carried out using only the block-based disparity-compensated H.264 prediction.
2. The prediction of views is carried out *adaptively* employing the two-stage prediction technique (DIBR-prediction followed by disparity-compensated prediction) from Section II-*C*.

For coding experiments, we employed the open-source H.264 encoder x264 [18]. The arithmetic coding algorithm CABAC was enabled for all experiments and the motion search was 32×32 pixels. All predicted frames are encoded as P-frames while reference texture and depth frames are encoded as I-frames. We set the number of reference frames to two: one reference for the block-based motion-prediction and a second reference for the DIBR-based prediction. Prior to image rendering, the reference depth is encoded with quantizer setting QP=29. Note that depth images should be encoded at a relatively high quality to avoid ringing-artifacts along object borders in the depth map. This prevents that rendering artifacts occur in the DIBR-based predicted view. This remark is similar to the conclusions related to recent depth compression results [8]. A dedicated depth image coder has been recently developed by the authors to prevent these specific rendering artifacts [22]. Because depth data is necessary for 3D rendering in any case, it can be assumed that depth images are transmitted even in the case no DIBR-based prediction is employed. Hence, employing the DIBR-based prediction does not involve any bit-rate overhead. It should therefore be noted that the presented rate-distortion curves of texture video do not include the bit-rate of depth images.

Let us now discuss the obtained rate-distortion curves for *multi-view texture coding* of Figure 5 and Figure 6. First, it can be observed that the proposed DIBR-based prediction algorithm consistently outperforms the block-based disparity-compensated prediction scheme. For example, considering Figure 5, the DIBR-based prediction algorithm yields a quality improvement of up to 1 dB at 200 kbit/s and 0.4 dB at 400 kbit/s over the block-based disparity-compensated prediction algorithm. Considering Figure 6,

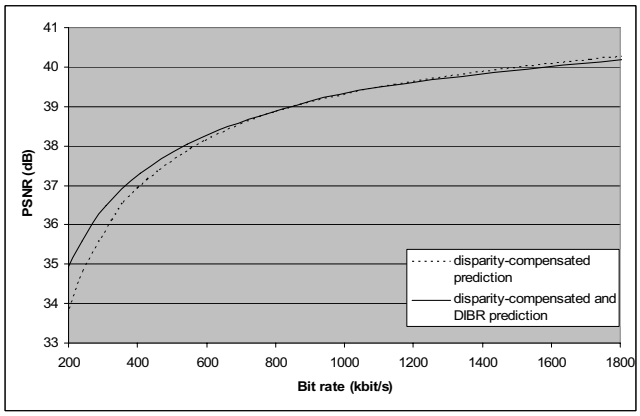


Fig. 5 Rate-Distortion curves of the multi-view texture sequence "Breakdancers".

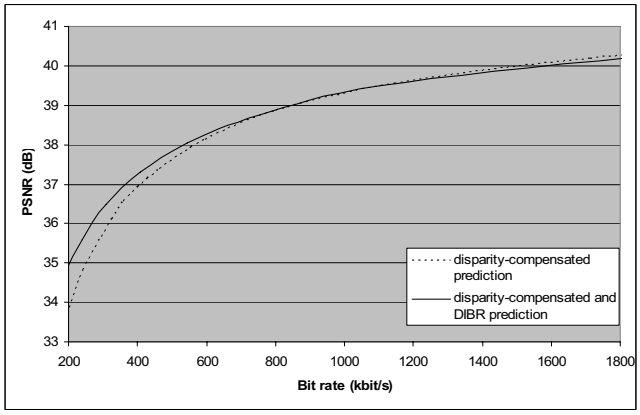


Fig. 6 Rate-Distortion curves of the multi-view texture sequence "Ballet".

although predicted views show large regions of occluded pixels, 0.2 dB quality improvement was obtained at a bit-rate of 500 kbit/s. Therefore, a DIBR-based prediction yields a coding improvement especially at low bit rates. It should be noted that because the two prediction algorithms are employed in parallel, the disparity-compensated prediction curve represents the lower bound of the coding performance of the proposed multi-view coder, as it again employs the disparity-compensated prediction.

We now discuss the obtained rate-distortion curves for *multi-view depth coding* of Figure 7 and Figure 8. First, it can be observed that significant coding improvements can be obtained by using the proposed DIBR-based prediction for coding multi-view depth images. For example, observing Figure 7, it can be seen that a constant gain of 1 dB can be obtained at any bit-rate. Similarly, considering the "Ballet" sequence, an improvement of up to 3 dB is obtained at a bit-rate of 500 kbit/s. These very attractive results can be explained by two aspects. First, the smooth properties of depth images simplify the DIBR prediction, so that an accurate depth prediction is possible. Second, the integration of camera parameters into the prediction scheme allows the algorithm to obtain accurate prediction of depth pixels for various baseline distances between cameras. In contrast, a disparity-compensated prediction scheme yields a low view-prediction quality for a large baseline multi-view acquisition setup.

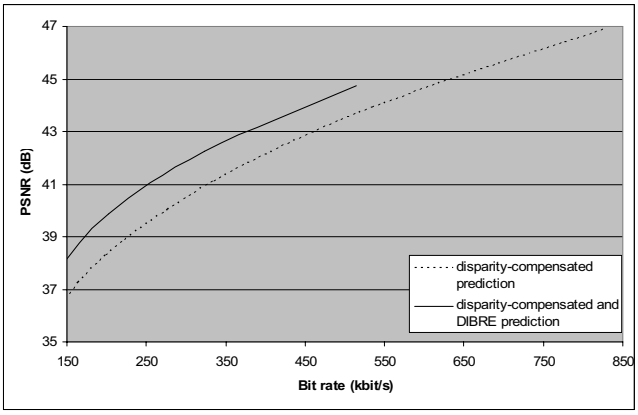


Fig. 7 Rate-Distortion curves of the multi-view depth sequence "Breakdancers".

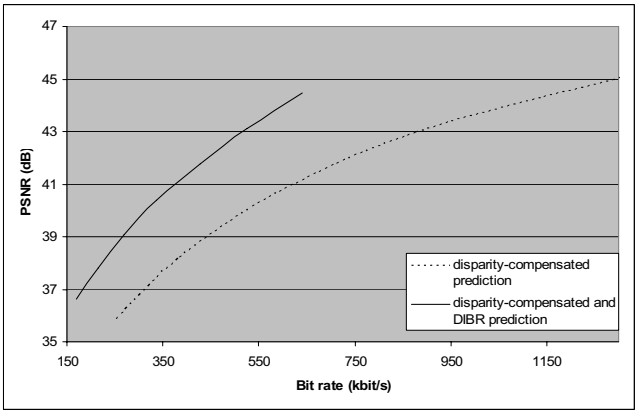


Fig. 8 Rate-Distortion curves of the multi-view depth sequence "Ballet".

IV. CONCLUSIONS

In this paper, we have presented an architecture for a complete 3D video coding system that yields high-quality image rendering while retaining high compression efficiency. To enable high-quality rendering and avoid occluded regions, the proposed 3D video system is based on a *N-depth/N-texture* video representation format, where each camera view covers different regions of the video scene. We have addressed the problem of multi-view depth estimation. Then we have proposed a depth-estimation technique that allows employing several views simultaneously and circumvents the need of image-pair rectification. Therefore, the technique simultaneously features a lower complexity and an increased accuracy when compared to a recent proposal [8]. Additionally, because the depth estimation is a line-based algorithm, the depth estimation require low memory, and is therefore attractive for consumer electronic products. In a second part, to render synthetic images, we have introduced a variant of the relief texture mapping. The described rendering technique efficiently handles holes/occluded pixels in the rendered images and can also be executed favorably by a Graphic Processor Unit. Finally, in a third part, the problem of multi-view depth and texture video coding was discussed. We have presented an algorithm for the predictive coding of multiple camera views that employs two different view-prediction algorithms: (1) a block-based motion prediction and (1) a DIBR-based prediction. The advantages of the algorithm are that the compression is robust against inaccurately estimated depth images and that the chosen prediction structure features random access to different views. Furthermore, we have integrated the prediction scheme into an H.264 encoder, such that disparity-compensation prediction is combined with the DIBR-based prediction. Experimental results have shown that the DIBR-based predictive-coding algorithm can improve the resulting *texture image* quality by up to 1 dB at low bit-rate and the quality of *depth image* by up to 3 dB when compared to solely performing H.264 disparity-compensated prediction. Finally, the proposed system is able to cope with different multi-view datasets with varying baseline distances between cameras. As a result, both 3D-TV and free-viewpoint multi-view video datasets can be efficiently compressed using the single proposed system architecture.

REFERENCES

[1] W. Matusik, H. Pfister "3D TV: A Scalable System for Real-Time Acquisition, Transmission and Auto stereoscopic Display of Dynamic Scenes"", *ACM Transactions on Graphics (TOG) SIGGRAPH,* Vol. 23, Issue 3, pp. 814-824, August 2004.

[2] C. Fehn "Depth-Image-Based Rendering (DIBR), Compression and Transmission for a New Approach on 3D-TV" In *Proceedings of SPIE Stereoscopic Displays and Virtual Reality Systems XI,* pp 93–104, San Jose, CA, USA, January 2004.

[3] "Applications and Requirements for StereoScopic Video (SSV)" ISO/IEC JTC1/SC29/WG11 Bangkok, Thailand, January 2006.

[4] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. A. J. Winder and R. Szeliski "High-quality Video View Interpolation Using a Layered Representation", *ACM Transactions on Graphics (TOG) SIGGRAPH,* Vol. 23, Issue 3, pp 600–608, August 2004.

[5] Y. Morvan,, D. Farin, P. H.N. de With "Predictive coding of depth images across multiple views" in *Proceedings of SPIE, Stereoscopic Displays and Applications*, Vol. 6490, January 2007, San Jose, USA.

[6] Y. Morvan, D. Farin, P. H.N. de With "Incorporating depth-image based view-prediction into H.264 for multiview-image coding" in IEEE International Conference on Image Processing (ICIP), Vol. I, pp 205–208, September 2007, San Antonio, USA.

[7] D. Scharstein and R. Szeliski "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms" *International Journal of Computer Vision*, Vol. 47(1/2/3), pages 7–42, April-June 2002.

[8] P. Kauff and N. Atzpadin and C. Fehn and M. Mueller and O. Schreer and A. Smolic and R. Tanger, "Depth map creation and image-based rendering for advanced 3DTV services providing interoperability and scalability", in *Signal Processing: Image Communication,* Vol. 22, no.2, pp. 217–234, 2007.

[9] P. Merkle, K. Mueller, A. Smolic, and T. Wiegand, "Efficient compression of multi-view video exploiting inter-view dependencies based on H.264/MPEG4-AVC," in *IEEE International Conference on Multimedia and Expo (ICME 2006),* Vol. 1, pp. 1717–1720, Toronto, Canada, 2006.

[10] R. Hartley and A. Zisserman "Multiple View Geometry in Computer Vision*" Cambridge University Press,* 2004.

[11] Aaron F. Bobick and Stephen S. Intille, "Large occlusion stereo", *International Journal of Computer Vision,* Vol. 33, no. 3, pp. 181–200, 1999,

[12] L. McMillan "An Image-Based Approach to Three-Dimensional Computer Graphics" *Ph.D. thesis* University of North Carolina, April 1997.

[13] Manuel M. Oliveira, "Relief Texture Mapping", *Ph.D. thesis,* University of North Carolina, March 2000.

[14] G. Wolberg "Digital Image Warping" *IEEE Computer Society Press,* July 1990.

[15] Y. Morvan, D. Farin, and P. H. N. de With, "Design considerations for view interpolation in a 3D video coding" *27th Symposium on Information Theory in the Benelux,* 2006.

[16] P. E. Debevec, Y. Yu and G. D. Borshukov "Efficient view-dependent image-based rendering with projective texture mapping" Eurographics Rendering Workshop 1998, pp 105–116, June 1998, Vienna, Austria.

[17] "MPEG document, Survey of algorithms used for Multi-View Video Coding (MVC)." ISO/IEC JTC1/SC29/WG11, MPEG2005/N6909, January 2005.

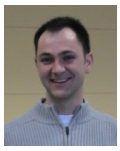
[18] Webpage title: x264 a free H.264/AVC encoder, http://www.videolan.org/developers/x264.html, last visited: April 2008.

[19] C. Fehn, N. Atzpadin, M. Müller, O. Schreer, A. Smolic, R. Tanger, and P. Kauff, "An advanced 3DTV concept providing interoperabilty and scalabilty for a wide range of multi-baseline geometries" in *IEEE International Conference on Image Processing*, Atlanta, USA, October 2006.

[20] Y. Morvan, P. H.N. de With, and D. Farin, "Platelet-based coding of depth maps for the transmission of multiview images", in *Stereoscopic Displays and Applications XVII, Proceedings of the SPIE*, 2006.

[21] Y. Morvan, D. Farin, and P. H. N. de With, "Design considerations for a 3D-TV architecture" in IEEE International Conference on Consumer Electronics, Las Vegas, USA, January 2008.Y. Morvan, D. Farin, and P. H. N. de With, "Depth-Image Compression based on an R-D Optimized Quadtree Decomposition for the Transmission of Multiview Images" in IEEE International Conference on Image Processing (ICIP), pp 205–208, San Antonio, USA, Sept. 2007.

[22] E. Martinian and A. Behrens and J. Xin and A. Vetro, "View Synthesis for Multiview Video Compression" in Picture Coding Symposium, Beijing, China,. April 2006.



Yannick Morvan received his M.S. in Electrical Engineering from the Institut Supérieur d'Electronique et du Numérique (ISEN), France in 2003. During his undergraduate studies, he worked, in 2002, at Philips Research on embedded image processing software and, in 2003, at Philips Medical Systems on X-ray image quality enhancement algorithms. In 2004, he joined, as a Ph.D. candidate, the Video Coding and Architectures research group at the Eindhoven University of Technology, The Netherlands. During his Ph.D. project, he was involved in a joint project of Philips Research and the Eindhoven University of Technology about the development of a multi-camera video acquisition and compression system for 3-D television. In 2006, he co-organized with Philips Research the "IEEE workshop on Content Generation and Coding for 3D-television". His research interests include multi-view coding, 3D reconstruction and image rendering. One of his papers on multi-view coding was a Best Paper Finalist at the 2007 Picture Coding Symposium in Lisbon, Portugal. In 2008, Yannick Morvan became imaging scientist at Philips Healthcare, Best, The Netherlands.

Dirk Farin graduated in computer science and electrical engineering from the University of Stuttgart, Germany. In 1999, he became research assistant at the Department of Circuitry and Simulation at the University of Mannheim. He joined the Department of Computer Science IV at the University of Mannheim in 2001 and the VCA group at the University of Technology Eindhoven in 2004. In 2005, he received his Ph.D. degree from the University of Technology Eindhoven, Netherlands. He received a best student paper award at the SPIE Visual Communications and Image Processing conference in 2004 for his work on multi-sprites, and two best student paper awards at the Symposium on Information Theory in the Benelux in 2001 and 2003. His research interests include video-object segmentation, video compression, content analysis, and 3-D reconstruction. In 2005, he organized a special session about sports-video analysis at the IEEE International Conference on Multimedia and Expo. In 2008, Dirk Farin became scientist at Robert Bosch Security System, Hildesheim, Germany.

Peter H.N. de With, IEEE Fellow, obtained his MSc. in electrical engineering from the Eindhoven University of Technology, and his PhD. From Delft University of Technology, The Netherlands. He joined Philips Research Labs Eindhoven in 1984, where he worked on video coding for digital recording. From 1985 to 1993, he was involved in several European projects on SDTV and HDTV recording. In this period, he contributed as a principal coding expert to the DV standardization for digital camcording. Between 1994 and 1997 he was leading the design of advanced programmable video architectures at the same lab. In 1996, he became senior TV systems architect and in 1997, he was appointed as full professor at the University of Mannheim, Germany, at the faculty Computer Engineering. In Mannheim he was heading the chair on Digital Circuitry and Simulation with the emphasis on video systems. Between 2000 and 2007, he was with LogicaCMG in Eindhoven as a principal consultant and also professor at the Eindhoven University of Technology, at the faculty of Electrical Engineering. He is now with CycloMedia Technology, The Netherlands. He has written and coauthored over 200 papers on video coding, architectures and their realization. Regularly, he is a teacher of the Philips Technical Training and for other post-academic courses. In 1995 and 2000, he coauthored papers that received the IEEE CES Transactions Paper Award, and in 2004, the VCIP Best Paper Award. In 1996, he obtained a company Invention Award. Mr. de With is IEEE Fellow, advisor to Philips, scientific advisor of the Dutch Imaging school ASCII, IEEE ISCE and board member of various working groups.