Design Considerations for View Interpolation in a 3D Video Coding Framework

[1]Yannick Morvan , [1]Dirk Farin
[1]Eindhoven University of Technology
P.O. Box 513
5600MB Eindhoven
The Netherlands
y.morvan@tue.nl

[1,2]Peter H.N. de With
[2]LogicaCMG Netherlands
RTSE1, P.O. Box 7089
5605JB Eindhoven
The Netherlands

Abstract

A 3D video stream typically consists of a set of views capturing simultaneously the same scene. For an efficient transmission of the 3D video, a compression technique is required. In this paper, we describe a coding architecture and appropriate algorithms that enable the compression and reconstruction of views at the decoder. We explore the concept of predicting views in order to reduce transmission bandwidth, and add a correction to maintain sufficient accuracy. The correction contains occlusion data to cover complicated situations in 3D scenes. Additionally, we present experimental results for occlusion handling, performing a more accurate synthesis of views for the rendering on a multiview display.

1 Introduction

A 3D video stream is typically obtained from a set of synchronised cameras, which are capturing the same scene (multiview video). This technique enables interesting applications such as free-viewpoint video, which enable the viewer to select his preferred viewpoint, or 3D TV, where the depth of the scene can be perceived using a special display. Multiview display technology which allows the presentation of images with the illusion of depth is strongly emerging. This illusion of depth is provided by showing simultaneously several views of the same scene. An independent transmission of these views is inefficient, since there is a strong correlation between the views of the same scene. Moreover, the aforementioned 3D displays support a varying number of views at the input, which makes it impractical to prepare these views prior to transmission. Instead, the views should be *rendered* or *interpolated* at the receiver where the display characteristics are known. It would be efficient to consider views that are coded and transmitted and directly reused for display. To render novel views at the decoder, several approaches have been investigated. A first class of approaches considered the use of *stereo* video from which intermediate views are interpolated. However, interpolation of intermediate views without a 3D geometric description of the scene is a difficult task. Moreover, it is crucial that the interpolation of views is as accurate as possible because no residual image is sent. A second class involves an alternative approach using 3D geometry data from which it is possible to derive the warping transformation for the interpolated views at the decoder. For example, a possible coding setup could first transmit a left image combined with a corresponding 3D geometry description, from which the right image can be predicted. Missing data in occluded regions can be filled in using a residual image. The intermediate views can then be interpolated from the left texture image, 3D geometry and the occlusion-compensated views.

In this paper, we propose to use a depth signal for 3D geometry information that enables the view-prediction, i.e. compression, of a multiple-view video stream. Additionally, we discuss different occlusion-handling techniques for the intermediate views.

This paper is organised as follows. Section 2 describes the 3D compression system using view prediction. Section 3 presents techniques for view interpolation. Section 4 introduces occlusion-handling techniques and presents various approaches for padding missing data areas. The experimental results are included within Section 4. Section 5 draws conclusions.

2 Overview of the 3D video coding framework

Our work is motivated by the problem of capturing and transmitting a set of views to render it onto a remote 3D display. For such an application, a typical capturing setup consists of a set of cameras pointing in the same direction such that the views are highly correlated. To exploit the inter-view correlation, the adopted coding approach is to predict the views from the leftmost view, using a corresponding 3D geometry representation. In our case, the 3D geometry data is represented by a depth image (see Figure 1) that specifies the distance between a point in the 3D world and the camera. This distance information is mapped onto a gray-level image. Therefore, the view predictor is based on an image *interpolation* technique known as Depth Image Based Rendering (DIBR). In practice, a depth image can be estimated from multiple images by calculating the parallax motion of pixels between views. Based on this

(a) (b)

Figure 1: The "Cones" [1] texture image and the corresponding depth image.

depth image and a texture image, arbitrary views can be interpolated at the decoder. In the case a region in the rightmost view is not visible in the leftmost view, it is not possible to predict the corresponding pixel values. This results in undefined pixels, i.e. occluded pixels, in the new view, which shows perceptual artefacts on the display. In this particular case, a perfect reconstruction of views can be obtained by transmitting residual information. An overview of the 3D video coding framework is depicted in Figure 2.

3 View interpolation using depth images

In this section, we describe the framework that enables the interpolation of views based on a texture image and a depth image.

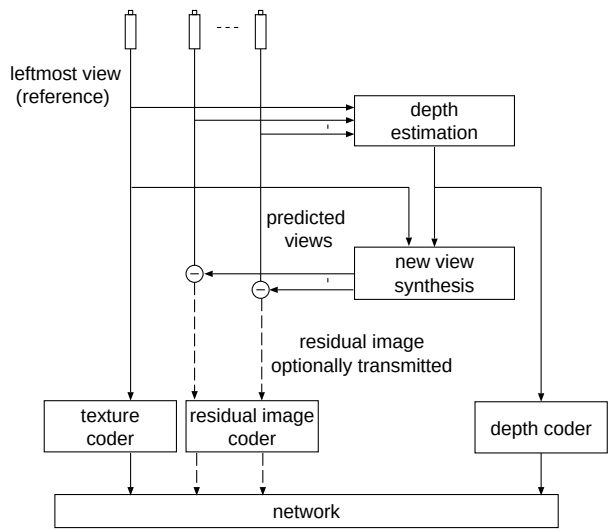


Figure 2: Overview of the 3D video coding system.

A single texture image and a corresponding depth image are sufficient to interpolate synthetic views from arbitrary positions. Let us consider a 3D point at *homogeneous* coordinates $\boldsymbol{p} = (X_p, Y_p, Z_p, 1)^T$ captured by two cameras and projected onto the left and right image planes at pixel positions $\boldsymbol{p_l} = (x_l, y_l, 1)^T$ and $\boldsymbol{p_r} = (x_r, y_r, 1)^T$, respectively (see Figure 3). We assume that the leftmost camera is located at the coordinate-system origin and looking along the Z-direction. The rightmost camera location and orienta-

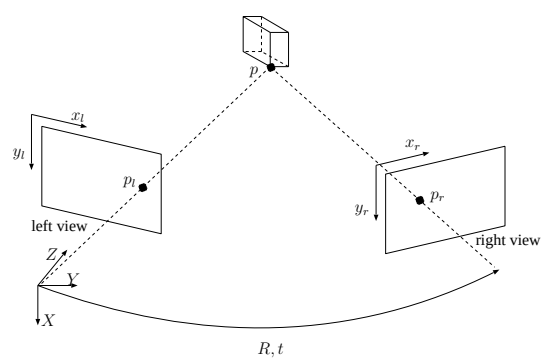


Figure 3: Two projection points $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ of a point $\boldsymbol{p}$.

tion are described by the rotation and translation matrices $\boldsymbol{R}$ and $\boldsymbol{t}$. This allows us to define the pixel positions $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ in both image planes by

$$\lambda_l \boldsymbol{p_l} = [\boldsymbol{K_l}|\boldsymbol{0_3}]\,\boldsymbol{p}\ , \tag{1}$$

$$\lambda_r \boldsymbol{p_r} = [\boldsymbol{K_r}|\boldsymbol{0_3}] \begin{bmatrix} \boldsymbol{R} & -\boldsymbol{R}\cdot\boldsymbol{t} \\ \boldsymbol{0_3^T} & 1 \end{bmatrix} \boldsymbol{p} = \boldsymbol{K_r}\boldsymbol{R} \begin{pmatrix} X_p \\ Y_p \\ Z_p \end{pmatrix} - \boldsymbol{K_r}\boldsymbol{R}\cdot\boldsymbol{t}\ , \tag{2}$$

where $\boldsymbol{K_l}$, $\boldsymbol{K_r}$ represent the 3×3 intrinsic parameter matrix of the corresponding cameras and λ_l, λ_r some positive scaling factors [2] which can be specified in this particular case by $\lambda_l = \lambda_r = Z_p$. From Equation (1), the 3D position of the original

point $\boldsymbol{p}$ in the Euclidean domain can be written as

$$(X_p, Y_p, Z_p)^T = \boldsymbol{K_l}^{-1}\lambda_l \boldsymbol{p_l} = \boldsymbol{K_l}^{-1} Z_p \boldsymbol{p_l}. \tag{3}$$

Finally, we obtain the predicted pixel position $\boldsymbol{p_l}$ by substituting Equation (3) into Equation (2):

$$Z_p \boldsymbol{p_r} = \boldsymbol{K_r}\boldsymbol{R_r}\boldsymbol{K_l}^{-1} Z_p \boldsymbol{p_l} - \boldsymbol{K_r}\boldsymbol{R_r} \cdot \boldsymbol{t}. \tag{4}$$

Equation (4) constitutes the image-warping equation (see [3]) that enables the interpolation of the rightmost view from the original textured leftmost-view pixels and their corresponding depth value.

The image-warping equation can be simplified with the following assumptions. First, using identical cameras, the intrinsic parameter matrices are assumed to be equal and can be written as

$$\boldsymbol{K_r} = \boldsymbol{K_l} = \begin{pmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{pmatrix},$$

where f and (o_x, o_y) corresponds to the focal-lenght of the recording camera and the principal point of the image, respectively. Second, the leftmost camera is not rotated, i.e. $\boldsymbol{R} = \boldsymbol{I}_{3\times 3}$ and third, the synthetic camera is translated horizontally by a distance t_x, i.e. $\boldsymbol{t} = (t_x, 0, 0)^T$. Rewriting Equation (4) and using the previously mentioned assumptions, the analysis simplifies to a 2D problem except for a scaling factor in the third (Z) dimension. This scaling factor Z_p can be removed by converting the 2D homogeneous coordinates of both points $\boldsymbol{p_l}$ and $\boldsymbol{p_r}$ into the Euclidean domain, so that

$$(x_r, y_r)^T = (x_l - \frac{f \cdot t_x}{Z_p}, y_l)^T. \tag{5}$$

Equation (5) provides the relation between the depth Z_p and the so-called *disparity* $\frac{f t_x}{Z_p}$, when the inter-view motion is limited to the case of horizontal motion. As a result, in this restricted case, interpolating the rightmost-view corresponds to shifting the leftmost-view pixels horizontally. For simplicity, we only consider the restricted case of horizontal motion between cameras in the sequel of this paper.

Note that a warping transformation uniquely computes the destination position for each input pixel in the other view, but that it is not possible to *invert* this function in order to find a corresponding pixel in the input image for each pixel in the new view. This problem becomes mostly apparent in occlusion regions, where no texture information can be found in the input image.

4 Occlusion handling

When no texture is available, i.e. in an occluded region, texture and depth data in the new view must be interpolated [3]. We first address the problem of determining the position of occluded pixels in the interpolated view and secondly describe several techniques to properly interpolate values of occluded pixels.

4.1 Determining the position of occluded pixels

During the warping procedure, multiple original pixels can be projected onto the same pixel position in the interpolated view. For example, a foreground pixel can occlude a background pixel in the interpolated view, which is resulting in overlapped pixels. Additionally, some regions in the interpolated view are not visible from the original viewpoint which results in holes in the interpolated image (see Figure 4). This problem is known as the *visibility problem* in the computer-graphics community.

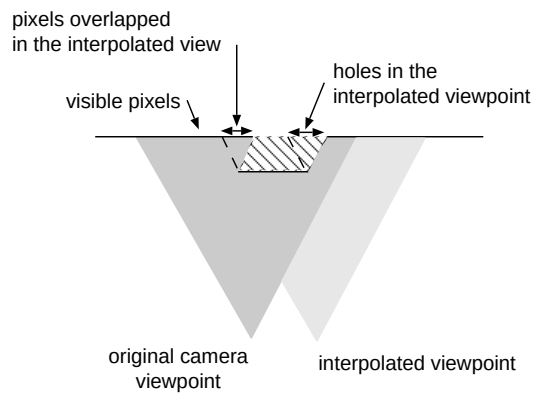


Figure 4: An interpolated view consists of visible, overlapped and undefined pixels.

First, to determine which pixels are visible in the interpolated view, one approach is to scan the original view such that eventually occluded background pixels are obtained prior to the occluding foreground pixels [3]. Since we are considering only a horizontal shift of the camera, view interpolation can be processed row by row. The scan of the depth image is performed with increasing x-coordinate such that foreground pixels overwrite background pixels, while foreground pixels are always preserved. Second, to determine the undefined-pixel positions, a per-pixel flag is set when the interpolated pixel position is written. These per-pixel flags finally provide a map of occluded pixels. The procedure is illustrated by Figure 5.

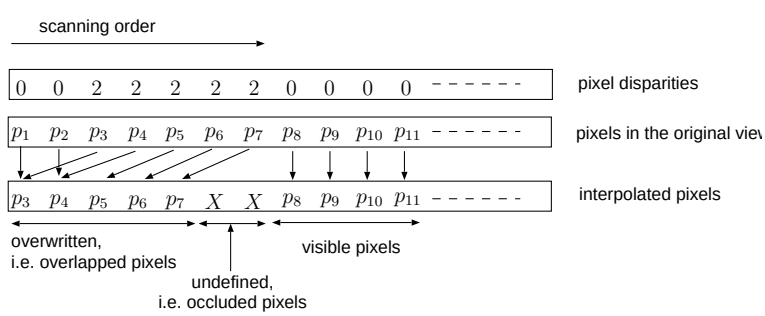
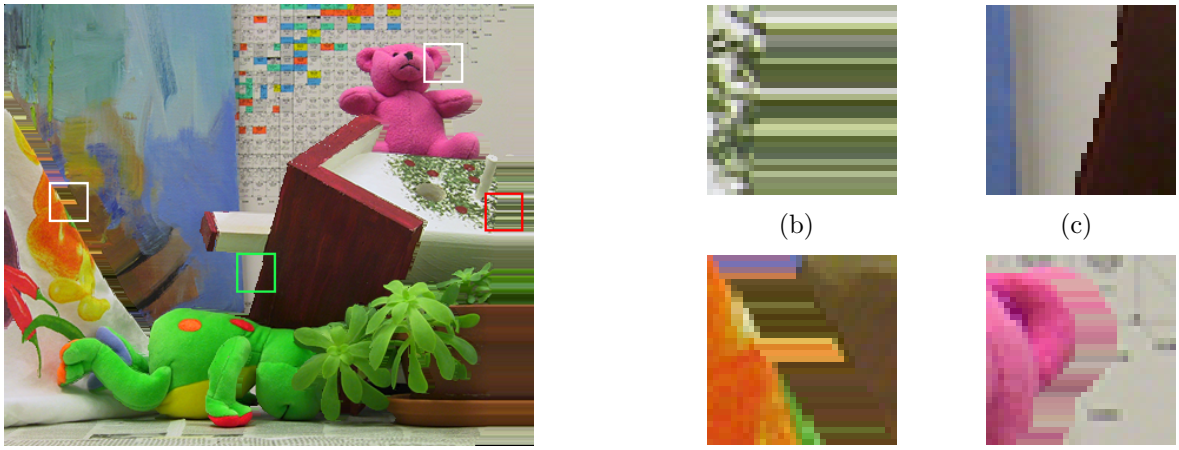


Figure 5: A proper scanning order of the original view enables that foreground pixels, p_3, p_4, p_5, p_6 overwrite the background pixels, i.e. p_1, p_2. Visible background pixels p_8, p_9, p_{10}, p_{11} are preserved. The arrows indicate a simple copy operation. The symbol X represents undefined data.



(a) (b) (c) (d) (e)

Figure 6: (a) Interpolated view using an occlusion-padding technique based on background pixels. (b) Magnified view that shows the "rubber band" effect. (c) Magnified view with aliasing artefacts at the object boundaries. (d)+(e) Visual artefacts generated by propagating blended colour in occluded pixels.

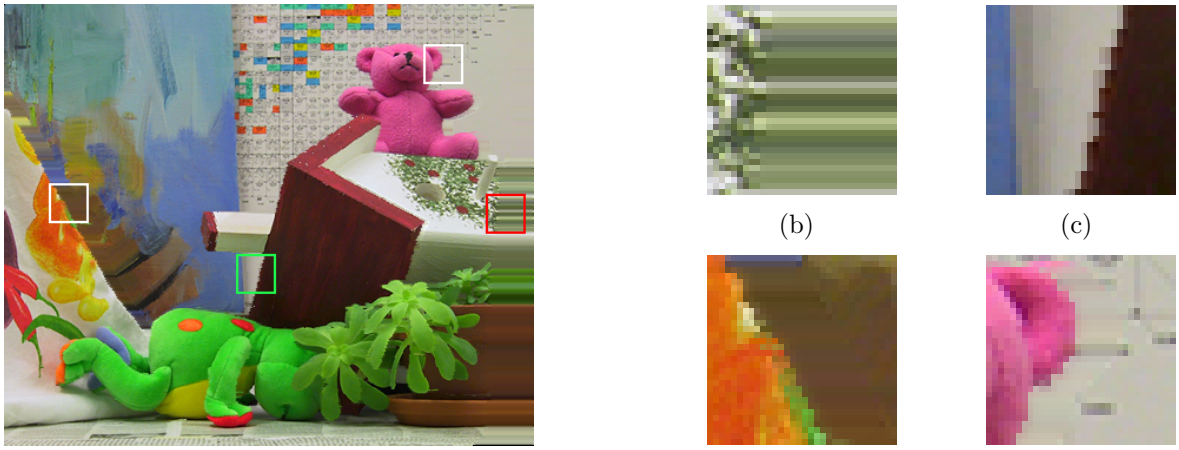
4.2 Occlusion padding

For the padding of occluded pixels, we have basically three possibilities. First, the occluded data could be sent as new independent data (intra mode). Second, undefined pixels could be extrapolated from the background i.e. the surrounding pixels with the largest depth. Third, these areas could be extrapolated from the background, and post-processing operations could be performed to increase subjective perceptual quality of interpolated images.

Intra-mode. To obtain a perfect reconstruction of the rightmost view, a residual data that corrects the prediction error should be transmitted. However, in some cases, heuristic padding techniques which fill occluded regions by synthetic texture, provide a sufficient perceptual quality.

Occlusion padding based on background pixels. The second approach exploits the idea that occluded pixels belong to the background. Thus, to interpolate occluded pixel values, only neighboring *non-occluded* pixels with the highest depth value should be taken into account. For simplicity, we define neighboring pixels as the two nearest left and right non-occluded pixels in the image line. The value of occluded pixels is then derived from the neighboring pixel with the highest depth value, i.e. background. An example of a warped image with the occlusion padded with background pixels is shown in Figure 6(a).

Unfortunately, as illustrated in Figure 6(b), because the padding is performed row by row, occluded regions show lines of equal pixels, which is known as the "rubber band" artefact. Secondly, in a typical textured image, object-border pixels show a mixture of foreground and background colour, which results in blended colour pixels. As a consequence, occluded pixels are padded by the blended colour instead of the background colour (see Figure 6(d) and Figure 6(e)). Finally, it can be observed in Figure 6(c) that



(a) (b) (c) (d) (e)

Figure 7: (a) Interpolated view with occlusion padding based on background pixels and smoothened discontinuities. (b) Rubber band artefacts attenuated by smoothing the occluded pixels. (c) Attenuated aliasing artefacts along the objects boundaries. (d) (e) Blended colours are not propagated in occluded pixels.

objects borders can show non-soft edges due to the pixel-based interpolation technique which lead to aliasing artefacts. For these reasons, we have investigated an alternative padding technique that appropriately handles object boundaries.

Occlusion padding with object-boundary smoothing. As described previously, apparent artefacts mostly occur along object borders. Therefore, we now introduce a simple algorithm that intends to minimize the perceptual impacts of these artefacts.

In a first step of the algorithm, we replace the blended edge pixels in the original image. Because pixels along the objects border are a mixture of foreground and background colour, we classify the pixels on the object border as not valid. To obtain the classification map of unreliable pixels, an edge-detection procedure is performed on the depth image. It can be found in literature that recent depth-estimation algorithms provide sufficiently accurate depth images to detect the object borders [4, 5]. The algorithm then replaces each unreliable pixel by the nearest valid pixel in the image line. This results in an image I_{nb} in which object boundaries are not blended (the suffix nb means "not blended").

In a second step, the not-blended image is warped and subsequently, edge pixels are blended. The image I_{nb} and the depth image are warped to obtain the interpolated view and the corresponding warped depth map. In this warping step, texture and depth samples are extrapolated from the background pixels (as described in the previous paragraph). Because the resulting warped view shows non-blended object boundaries, the algorithm finally smoothens the edges to obtain soft object boundaries. To attenuate the "rubber band" artefacts, the occluded regions are smoothened as well.

Though multiple heuristic techniques have been employed for padding the occluded pixels, experiments have revealed that our proposed algorithm can interpolate new views with sufficient quality. For example, Figure 7(d) and Figure 7(e) show clearly

that blended colours are not replicated in the occluded regions. Moreover, the aliasing (jagged edges) and "rubber band" artefacts are attenuated by a simply smoothing of the occluded pixels (Figure 7(b)) and the object borders (Figure 7(c)), respectively.

5 Conclusions

In this paper, we have described a framework for 3D video coding system which employs a prediction of views based on an original view combined with depth information. The depth information is enhanced with residual image data for handling occlusion. The prediction of views is performed using a view interpolation and occluded data can be efficiently predicted or explicitly transmitted. It was shown that the interpolated view can be reconstructed by a transformation of the original view using the warping equation. As an illustration, we have derived a simple example of the transformation based on horizontal camera-motion only. This architecture enables multiview synthesis by predicting any desired in-between view from the geometry data coded by the depth image, where the occlusion data ensures sufficient quality at object borders.

We have provided a new algorithm for padding of occluded pixels. The key to this algorithm is that blended object-border pixels are first removed and replaced by neighbouring data. Afterwards, the image is warped and object-border pixels are blended.

The advantage of the presented padding technique is the low complexity as only pixel-copy operations and spatial filters are employed to interpolate the new view.

Subjective evaluation of the interpolated views based on the new technique shows that the quality of the edges improves and blending colour effects are significantly reduced. Moreover, the low complexity view-interpolation technique makes our proposal an appealing candidate for consumer electronic applications such as 3D TV receivers.

References

[1] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 195–202, June 2003.

[2] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An Invitation to 3-D Vision*. Springer, November 2003.

[3] L. McMillan, *An Image-Based Approach to Three-Dimensional Computer Graphics*. University of North Carolina, April 1997.

[4] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *International Journal of Computer Vision*, vol. 47, Kluwer Academic Publisher, April-June 2002.

[5] M. Bleyer and M. Gelautz, "A layered stereo algorithm using image segmentation and global visibility constraints," in *IEEE International Conference on Image Processing*, pp. 2997–3000, 2004.