A NOVEL INTERPOLATION METHOD FOR 3D VIEW SYNTHESIS

Vincenzo Paradiso, Maurizio Lucenteforte, Marco Grangetto

Dipartimento di Informatica, Università di Torino
Corso Svizzera 185, 10149, Torino, ITALY
first.last@unito.it

ABSTRACT

In this paper a novel intermediate view synthesis algorithm for 3D video rendering is presented. The proposal is based on the representation of the pixels to be warped as intervals of finite length. The warping, hole-filling and merging stages of the view synthesis procedure can be recast according to this notation with significant advantages from the point of view of image synthesis quality, algorithm complexity and memory requirements. A novel interpolation rule based on the definition of a foreground and a background weight associated to each warped pixel is proposed and compared with linear interpolation based on pixels distance. The experimental results show that the designed technique yields a significant gain in terms of rendered image quality, while at the same time halving the execution time with respect to the MPEG view synthesis reference software.

Index Terms — 3DTV, low complexity rendering, Video plus Depth representation

1. INTRODUCTION

Nowadays many research and standardization efforts are being spent to shape the future of 3D TV and free-viewpoint TV. In 2011 MPEG issued the call for proposal on 3D video coding technology [1] and in 2012 a new ISO/ITU collaborative team devoted to 3D extensions [2] has been started. Promising results are being achieved using the multiple views plus depths format. Such format has two paramount advantages: by knowing the depth of the scene one can generate pictures at intermediate viewpoints using depth image based rendering (DIBR) techniques and the depth signal turns out to be highly compressible [3, 4, 5, 6].

In this context DIBR techniques play a key role. Indeed on the decoder side they allow the user to adjust the viewpoint of a stereoscopic pair or to feed forthcoming auto-stereoscopic displays with a large number of contiguous views. At the same time DIBR permits the video coder to create supplementary reference pictures improving its inter-view prediction ability, thus achieving the required compression efficiency. On the other hand, DIBR algorithms can be included in the prediction loop of the video encoder, provided that the predicted pictures are achieved at a reasonable computational cost.

The major shortcoming of the first DIBR attempts was that the image quality can be rather poor because of disocclusion and resampling issues, besides the errors induced by the depth estimation algorithms. As a consequence, most of the research activities have been pursuing a higher rendering quality at the expense of added computational complexity [7, 8, 9, 10, 11]. The major improvements have been obtained adopting sub-pixel precision for

warping, boundary aware processing of the disparity, hole filling, inpainting and noise cleaning. Most of these strategies have been included in the MPEG View Synthesis Reference Software [12] that will be used as a benchmark for performance in the present work.

In this paper we propose a novel approach to view synthesis stemming from the interpretation of each pixel as an interval of finite length. Warping an image amounts to shifting pixels intervals that can overlap one another with a geometrical arrangement that depends on their distance from the virtual camera. Such geometrical interpretation allowed us to design a novel resampling and interpolation strategy based on the definition of foreground and background weights. The warping, hole-filling and merging stages of the view synthesis procedure can be recast according to this notation with significant advantages from the point of view of the synthesis quality, the computational complexity and the memory requirements.

2. DEPTH IMAGE BASE RENDERING

In this paper we will consider the most typical multi-view scenario where the cameras recording the video are arranged with parallel optical axes. The parallel arrangement simplifies the correspondence among the pixels of any two views: in particular, a pixel in a reference view can be mapped to a new position in a virtual view, corresponding to a camera with horizontal shift b , by applying only a horizontal disparity δ which depends on the pixel depth z . The depth z is usually stored as an 8-bit unsigned integer d with the following non linear mapping:

$$\frac{1}{z} = \frac{d}{255} \left(\frac{1}{z_{near}} - \frac{1}{z_{far}} \right) + \frac{1}{z_{far}}$$

where z_{far} and z_{near} are the minimal and maximal depth of the reference picture. The horizontal disparity δ is then evaluated as

$$\delta = \frac{f \cdot b}{z} \quad (1)$$

where f is the focal length of the camera. As a consequence, any pixel with coordinates (v, u_r) in the reference picture will be shifted to the coordinate (v, u) in the virtual view with $u = u_r \pm \delta$, where the shift is positive (negative) when warping an image from right to left (left to right). Since δ is not constrained to take on integer values proper resampling and interpolation are required to fill the integer grid corresponding to the pixels of the virtual view. Moreover, along the edges disocclusions may show up uncovering portion of an object (typically the background) that is occluded in the reference view.

Many strategies have been proposed to tackle such issues. Since 2008 MPEG 3DV AdHoc group has been developing and

maintaining the VSRS tool [12] that now integrates many different contributions providing state of the art performance. The VSRS uses two reference views and two depth maps to generate virtual view at an intermediate position. The reference views are warped to the target viewpoint using pixel-by-pixel mapping based on 3D warping (General mode option), or horizontal pixel shifting (1D mode) if parallel camera arrangement is used. In 1D mode the reference views are up-sampled horizontally, e.g. by a factor 4, to achieve fractional shift accuracy. Boundary-aware splatting is adopted [11]. Two virtual views are projected from the left and the right references, respectively. Such views are down-sampled at their original resolution and blended together to form a unique virtual view. Finally, the remaining holes, i.e. pixels not covered by previous warping process, are filled using inpainting. For detailed description of VSRS we refer to [11]. It has been already recognized that the upsampling strategy is very demanding in terms of computational and memory requirements [13]. In [13] the upsampling approach is compared with a technique based on linear interpolation without upsampling showing that is possible to obtain competitive results with much lower complexity. In the following we will show that, by adopting a novel interpolation mechanism, it is possible to improve the synthesis quality and reduce the computational cost at the same time.

3. INTERVAL BASED VIEW SYNTHESIS

The proposed view synthesis algorithm stems from the observation that each pixel (in both reference and virtual views) represents a geometrical footprint on the image plane that cannot be simply described as a dimensionless location (v, u) . In the general case each pixel can be represented as a square area; in this paper, where we assume a parallel camera arrangement, it is convenient to simplify the notation assuming that each pixel with horizontal coordinate u represents the geometrical interval $I(u) = [u, u + 1]$ on the image plane. Therefore in the following we will drop the row index v .

As an example in Fig.1 we show 3 pixels from two reference views (Ref0, Ref1) with their corresponding geometrical interpretation and a target pixel at location u in the virtual view. According to the proposed approach it is possible to interpret the warping process as displacements of the reference pixel intervals. The goal of the synthesis process is to identify a set of reference pixels (potentially from all available views) that exhibit non null intersection with the target interval $I(u)$ and then to properly merge their color and depth information. According to the proposed algorithm one has to identify all the candidate pixels intervals $I(u_r)$ such that $I(u) \cap I(u_r + \delta) \neq \emptyset$, where δ is the disparity vector to be applied to reference position u_r . For the purpose of the following algorithm any desired level of accuracy can be used to represent the disparity vector. The proposed view synthesis algorithm first identifies all candidate intervals that intersect with $I(u)$ and then uses all the available geometrical data to merge their color information.

The proposed Interval Based Image Synthesis (IBIS) algorithm is presented using pseudo-code in Algorithm 1. The IBIS algorithm keeps searching for candidate intervals at target location u by increasing the tolerance parameter t so as to embed the hole filling process in the warping procedure. In fact, those target locations, where no reference pixel is warped to, can be covered by enlarging the reference pixel intervals $I(u_r, t) = [u_r - t, u_r + 1 + t]$ with $t > 0$ so as to obtain an intersection with the target interval.

Figure 1. Geometrical interpretation of the warping and blending steps.

Figure 2. Interpolation procedure.

We refer to the set of candidates for target location u as $\mathcal{C}(u)$. The blending procedure included in IBIS is designed to exploit all the geometrical information of the pixels inserted in $\mathcal{C}(u)$. In particular, each candidate interval participates to the final blending according to two weights corresponding to the foreground $w_{i,f}$ and background $w_{i,b}$ contributions as graphically shown in Fig. 2. The color information of the target location u is then computed as:

$$\hat{Y}(u) = \frac{\sum_{j=1}^{|\mathcal{C}(u)|} (w_j^f + \alpha_j \cdot w_j^b) Y(j)}{\sum_{j=1}^{|\mathcal{C}(u)|} (w_j^f + \alpha_j \cdot w_j^b)}, \quad (2)$$

where $Y(j)$ is the intensity corresponding to the j -th reference pixel in $\mathcal{C}(u)$, corresponding to a displaced interval $I_j(u_r, t)$. The weights w_j^f and w_j^b are computed as the portion of the interval of the j -th candidate in the foreground and background with respect to the target interval $I(u)$. To this end the candidate pixels in $\mathcal{C}(u)$ are processed for increasing values of their depth coordinate z , i.e. from the camera towards the background. The foreground weight is evaluated as

$$w_j^f = |I_j(u_r, t) \cap I^j(u)| \quad (3)$$

where $I^j(u)$ is the portion of the target interval that is not yet occluded by previous pixel intervals, i.e. with lower z , when processing the j -th candidate and operator $|\cdot|$ evaluates the interval length. $I^j(u)$ is computed recursively according to the following

Table 1. Test sequences.

Name	Resolution	Frames	ID
UndoDancer	1920 × 1088	250 , 25 Hz	S1
GTFLy	1920 × 1088	250 , 25 Hz	S2
Kendo	1024 × 768	300 , 30 Hz	S3
Balloons	1024 × 768	300 , 30 Hz	S4
News	1024 × 768	300 , 30 Hz	S5

equation:

$$I^j(u) = \begin{cases} I(u) & \text{if } j = 0 \\ I^{j-1}(u) \setminus (I^{j-1}(u) \cap I_{j-1}(u_r, t)) & \text{if } j > 0 \end{cases} \quad (4)$$

The j -th background weight is computed as

$$w_j^b = |I(u) \cap I_j(u_r, t)| - w_j^f. \quad (5)$$

Algorithm 1 Interval Based Image Synthesis (IBIS)

```

for  $u < \text{ImageWidth}$  do
  Initialize the candidate set  $\mathcal{C}(u) \leftarrow \emptyset$ 
  Set tolerance  $t \leftarrow 0$ 
  while  $\mathcal{C}(u) = \emptyset$  do
    Define pixel interval intervals  $I(u_r, t) = [u_r - t, u_r + 1 + t)$ .
    Add to  $\mathcal{C}(u)$  all pixel with position  $u_r : I(u_r + \delta, t) \cap I(u) \neq \emptyset$ 
    Increase tolerance  $t \leftarrow t + 1$ 
  end while
  Compute  $\hat{Y}(u)$  using (2)
end for

```

The parameter α_j is used to control the blending among foreground and background contributions. The experiments have shown that is reasonable to set this parameter as function of the difference between the depth of the j -th and first interval, i.e. the difference with respect to the foreground at location u . The following experiments have been worked out setting $\alpha_j = 1 - (z_j - z_1)/z_1$, i.e. using $\alpha_j \approx 1$ for pixels with a depth similar to the foreground at location u , and $\alpha_j \rightarrow 0$ for pixels that are far apart in terms of z .

The proposed IBIS algorithm exhibits a number of advantages. With respect to VSRS IBIS does not require upsampling of the reference pictures and downsampling of the synthesized view; moreover it allows for fractional displacements with any desired level of precision. All algorithm operations are local to a few pixels of a single row and therefore amenable to implementations with limited memory requirements. IBIS jointly performs warping, hole-filling and interpolation from any number of reference views using a single pass. Finally, the proposed geometric interpolation has the potential to improve the image quality as shown in Sect. 4.

4. EXPERIMENTAL RESULTS

The proposed IBIS has been included as an option of the VSRS tool, version 3.5.1 available at [1]. For research purpose the source code of the modified VSRS is made available through an svn repository upon request. The VSRS is used as a benchmark for performance. VSRS is configured to operate in 1D mode with 1/4 pixel disparity accuracy; default edge detection thresholds and splatting

options have been activated. The proposed approach, on the other hand, does not make use of splatting and avoids upsampling the reference pictures; in IBIS disparity (1) and interpolation (2) are computed in floating point precision. The synthesis experiments have been worked out using 5 MPEG 3DV test sequences [1], whose characteristics are summarized in Tab. 1. The first two sequences are 1920 × 1088 computer graphics animations, whereas the remaining ones are 1024 × 768 real video shots; in this latter case the original sequences underwent rectification and color correction.

Tab. 2 shows the experimental results obtained synthesizing a virtual view from two references. Each row of the table reports the ID of the sequence, the indexes of the reference views, e.g. r1-5, and the index of the synthesized target, e.g. s2. The virtual views have been generated in positions where original data are available so as to be able to compute the PSNR of the rendered pictures versus the original ones. In Tab. 2 the average Luma PSNR obtained for each synthesis by the VSRS and IBIS are shown along with the execution times reported on an Intel i7-2640M CPU at 2.80 GHz, 8GB RAM. The two rightmost columns report the IBIS gain in terms of PSNR increment and execution time reduction (the last column shows the execution time of IBIS in percentage versus VSRS). It can be noted that IBIS achieves an average gain of 0.61 dB in terms of PSNR with peak gains larger than 1 dB. This improvement is achieved along with a neat reduction of the running times that turn to be about 50% of the VSRS ones. This is mainly due to the absence of upsampling in IBIS.

In order to better investigate the effectiveness of the interpolation (2) an alternative blending procedure proposed in [10] has been implemented for comparison. In [10] the authors propose i) to keep as candidates only those pixels that are closest to the camera, i.e. with $z_j - z_1 \leq Th$ (using the notation of Sect. 3), ii) to blend the color information using a weight proportional to the distance between the candidate and the target coordinate u . We have tested IBIS with this alternative interpolation method, setting $Th = 0.1 \cdot z_1$. The column labeled as IBIS-[10] in Tab. 2 shows that, on average, the obtained PSNR is slightly lower than VSRS and therefore significantly lower than the proposed approach.

Finally, Fig. 3 and Fig. 4 show the PSNR obtained with IBIS and VSRS on all the frames of a computer graphic (GTFLy) and natural (Kendo) sequence, respectively. Also this results confirms the good performance of the proposed method. It can be noted that IBIS gain is more significant on the computer graphic sequence; this is likely to be due to the higher quality of the available depth maps, that in case of natural video need to be estimated with unavoidable inaccuracy. Depth errors are likely to limit the advantages of the IBIS accurate interpolation method. Due to lack of space some rendered pictures cannot be included in the manuscript; a sample video comparing VSRS versus IBIS is made available as supplementary material.

5. CONCLUSIONS

A novel approach to pixel warping for intermediate view synthesis has been proposed. The method promises to improve the image quality while keeping the computational complexity limited. The proposed view synthesis tool has been compared with VSRS and an alternative linear interpolation rule showing that it is able to improve the image quality. Moreover, the proposed algorithm is amenable to low complexity implementation. Future works include the investigation of IBIS performance when used for inter-

Table 2. IBIS performance in terms of average Luma PSNR and execution time.

Sequence	VSRS		IBIS		IBIS with [10]		IBIS vs. VSRS	
	PSNR [dB]	Time [s]	PSNR [dB]	Time [s]	PSNR [dB]	Time [s]	PSNR [dB]	Time (%)
S1,r1-5,s2	37.64	547.05	37.97	283.67	36.90	277.30	0.33	51.85
S1,r1-5,s3	37.50	546.75	38.66	281.80	36.71	278.76	1.16	51.54
S2,r1-5,s2	40.30	545.80	41.44	279.05	40.03	277.80	1.14	51.13
S2,r1-5,s3	39.30	547.00	41.13	278.80	39.68	276.67	1.83	50.97
S3,r1-3,s2	35.45	250.00	35.82	121.57	35.47	127.82	0.37	48.63
S3,r3-5,s4	37.64	250.08	38.42	122.37	37.72	127.01	0.78	48.93
S4,r1-3,s2	36.27	261.51	36.35	134.55	35.95	119.03	0.08	51.45
S4,r3-5,s4	36.64	254.62	37.00	134.75	36.70	120.06	0.36	52.92
S5,r2-4,s3	32.14	251.19	32.14	134.08	31.91	126.14	0.00	53.38
S5,r4-6,s5	34.88	253.32	34.95	134.4	34.50	125.28	0.07	53.07
Average	36.78	370.73	37.39	190.51	36.56	185.59	0.61	51.39

Figure 3. GTFly view 3 synthesis from reference 1 and 5.

Figure 4. Kendo view 2 synthesis from reference 1 and 3.

view prediction within a 3D video codec.

6. REFERENCES

- [1] "Call for proposals on 3D video coding technology," Tech. Rep. N12036, ISO/IEC JTC1/SC29/WG11, mar 2011, <ftp://ftp.merl.com/pub/avetro/3dv-cfp/>.
- [2] "ITU-T/ISO/IEC joint collaborative team on 3D video coding extension development," <http://phenix.it-sudparis.eu/jct2/index.php>.
- [3] Sehoon Yea and Anthony Vetro, "View synthesis prediction for multiview video coding," *Image Commun.*, vol. 24, no. 1-2, pp. 89–100, Jan. 2009.
- [4] K. Muller, A. Smolic, K. Dix, P. Merkle, and T. Wiegand, "Coding and intermediate view synthesis of multiview video plus depth," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, Nov. 2009, pp. 741–744.
- [5] Woo-Shik Kim, A. Ortega, Jaejoon Lee, and HoCheon Wey, "3D video coding using depth transition data," in *Picture Coding Symposium (PCS), 2010*, Dec. 2010, pp. 178–181.
- [6] D. Strohmeier and G. Tech, "Sharp, bright, three-dimensional: open profiling of quality for mobile 3DTV coding methods," in *SPIE Conference Series*, Feb. 2010, vol. 7542.
- [7] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *ACM SIGGRAPH and ACM Transactions on Graphics*, 2004, pp. 600–608.
- [8] Liang Z. and W.J. Tam, "Stereoscopic image generation based on depth images for 3D TV," *Broadcasting, IEEE Transactions on*, vol. 51, no. 2, pp. 191–199, june 2005.
- [9] Ya mei Feng, Dong xiao Li, Kai Luo, and Ming Zhang, "Asymmetric bidirectional view synthesis for free viewpoint and three-dimensional video," *Consumer Electronics, IEEE Transactions on*, vol. 55, no. 4, pp. 2349–2355, Nov. 2009.
- [10] Zefeng Ni, Dong Tian, S. Bhagavathy, J. Llach, and B.S. Manjunath, "Improving the quality of depth image based rendering for 3D video systems," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, nov 2009, pp. 513–516.
- [11] D Tian, Lai P., P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," in *SPIE proceedings, Applications of Digital Image Processing XXXII*, Aug. 2009, vol. 7443.
- [12] "View synthesis reference software (VSRS) 3.5," Tech. Rep., ISO/IEC JTC1/SC29/WG11, Mar. 2010.
- [13] G. Tech, K. Muller, and T. Wiegand, "Evaluation of view synthesis algorithms for mobile 3DTV," in *3DTV Conference 2011*, may 2011, pp. 1–4.