Pixel-Level View Synthesis Distortion Estimation for 3D Video Coding

Jian Jin, Jie Liang, *Senior Member, IEEE*, Yao Zhao, *Senior Member, IEEE*, Chunyu Lin, Chao Yao, and Lili Meng

***Abstract*— Recently, region-based 3D video coding has been proposed. However, existing view synthesis distortion estimation (VSDE) methods are performed at the frame level. To guide the rate-distortion optimization process of region-based 3D video coding schemes, this paper proposes the first pixel-level VSDE (PL-VSDE) method. We first give the definition of the pixel-level view synthesis distortion. To estimate it, a backward prediction method is then developed, which starts from the pixels of interest (POIs) in the virtual view and finds their corresponding pixels in the reference view via a coarse-to-fine approach, denoted as coarse-to-fine backward prediction (CFBP) method. Additionally, the CFBP fully considers the details of 3D warping, the rounding operation and the warping competition in view synthesis, leading to improve accuracy of the prediction. Besides, a table-lookup method and a warping property are introduced to speed up the CFBP. After integrating the CFBP into the PL-VSDE, we can estimate the view synthesis distortion at the pixel level. Our method is carried out pixel-by-pixel independently, which is friendly for parallel processing. The experimental results demonstrate that our proposed method has significant advantages in both accuracy and efficiency compared with the state-of-the-art frame-level VSDE methods.**

***Index Terms*— 3D video coding, depth-image-based rendering (DIBR), distortion estimation, 3D warping.**

I. Introduction

THREE dimensional (3D) video technology has been widely studied, since it can provide a fresh immersive experience of the real physical scene [1]. In 3D video systems, a physical scene is firstly captured by several cameras from different viewing angles. Then, the captured data are encoded by multi-view video plus depth (MVD) format [2]. After transmission, the decoded MVD data can be used to synthesize various virtual views that are not captured by the cameras via the depth-image-based rendering (DIBR) technology [3].

Recently, some region-based 3D video coding schemes have been proposed in [4]–[8], which can have better compression performance and more flexible applications. For example, if an interactive application only needs to synthesize some pixels of interest (POIs) or some regions of interest (ROIs) in the virtual view [9], the region-based 3D video coding can be used to only encode the corresponding regions in the reference views instead of the entire frames, which can greatly reduce the encoding time, the amount of transmitted data, and view synthesis complexity.

In 3D video applications, the distortion of the synthesized virtual view is crucial to its quality. Many factors can result in rendering distortions in 3D video systems. Generally, the lossy compression (source coding) is considered as the main factor, which introduces errors into the reference views (both texture and depth images) at the encoder and further causes rendering distortions in the synthesized virtual views at the decoder. Unlike errors in the texture image, which only leads to color distortion of the synthesized view, errors in the depth image can change the spatial geometry information of the physical scene, leading to unpredictable geometry distortion in the synthesized view [10]. Therefore, accurate view synthesis distortion estimation (VSDE) is desired [11]. For instance, it can be used in the rate-distortion optimization to control the view synthesis distortion [12].

However, most VSDE algorithms operate at the frame level, such as algorithms in [13]–[15], where the changes of the entire reference depth images are firstly analyzed with some statistical methods. Then, 3D warping is used to propagate the depth changes of the entire frame from reference views to the virtual view [9], in order to estimate the geometric distortion caused by depth changes. After being combined with the statistics of the color information changes, the VSDE is finally achieved. Generally, depth changes are regarded as noises and the expectation of their associated disparity changes are estimated by averaging over the entire frame in most VSDE methods. This frame-based design limits their accuracies to be at the frame level.

To implement row-by-row processing independently, the method in [15] estimates the first moment by averaging one row of the image, but it is essentially still a frame-based

Manuscript received December 19, 2018; revised March 8, 2019; accepted May 10, 2019. Date of publication May 22, 2019; date of current version July 2, 2020. This work was supported in part by the National Key Research and Development of China under Grant 2016YFB0800404, in part by the National Natural Science Foundation of China under Grant 61532005 and Grant 61772066, in part by the China Scholarship Council, and in part by the Engineering Research Council of Canada under Grant RGPAS478109. This paper was recommended by Associate Editor Z. Chen. *(Corresponding author: Yao Zhao.)*

J. Jin, Y. Zhao, and C. Lin are with the Institute of Information Science, Beijing Jiao Tong University, Beijing 100044, China, and also with the Beijing Key Laboratory of Advanced Information Science and Network Technology, Beijing 100044, China (e-mail: jianjin@bjtu.edu.cn; yzhao@bjtu.edu.cn; cylin@bjtu.edu.cn).

J. Liang is with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: jiel@sfu.ca).

C. Yao is with the Institute of Sensing Technology and Business, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: yaochao1986@gmail.com).

L. Meng is with the Department of Information Science and Engineering, Shandong Normal University, Jinan 250358, China (e-mail: mengll_83@hotmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2019.2918248

1051-8215 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

VSDE method. Therefore, the existing methods are not suitable for the emerging region-based 3D video coding schemes, which need to selectively estimate the distortion of some POIs in the virtual view instead of the entire frame. Additionally, only 3D warping is considered during modeling depth changes propagation in these methods, while some key components in DIBR are ignored, namely the warping rounding operation and warping competition [16]. This causes some loss of their accuracy. A detailed analysis will be given later. Therefore, an accurate and pixel-level VSDE method is desired for region-based 3D video coding applications.

In this paper, we develop the first VSDE method that can estimate the distortion of the POIs in the synthesized view at the pixel level so that it can be used by region-based 3D video coding schemes. The main contributions of the paper are listed as follows:

- We carefully analyze the depth-change-caused view synthesis distortion, which shows that the accuracies of the traditional depth-change-caused VSDE algorithms are not satisfactory, since they only use 3D warping to roughly propagate the depth-change-caused distortion from reference view to the warped view. To improve their accuracies, both the rounding operation and warping competition should be considered as well.
- We propose a pixel-level view synthesis distortion estimation method (PL-VSDE) to better serve the emerging region-based 3D video coding schemes. Our method is based on a backward prediction, which starts from a pixel in the virtual view, and find its corresponding pixel in the reference view via a coarse-to-fine approach. Besides, it also considers the details in 3D warping, rounding operation, and warping competition.
- We use a table-lookup method and the warping property to further speed up the CFBP. Besides, our method can be implemented pixel-by-pixel independently, which is friendly for parallel processing.

The outline of the rest of our paper is as follows. Section II reviews the techniques of view synthesis at first, and then analyzes the depth-change-caused view synthesis distortion. Section III and IV present the PL-VSDE and CFBP methods, respectively. Section V presents experimental results and Section VI concludes this paper.

II. Review and Analysis

A. Review of View Synthesis Algorithm

In this paper, only the DIBR-based view synthesis algorithm with 1D parallel mode is considered, which is briefly reviewed in this section. Generally, the algorithm contains two main steps: i) forward warping step, and ii) blending step. The details can be found in [16] and [17].

The forward warping step aims to warp all the texture pixels in the reference view to the warped view, which includes three sub-steps: i) 3D warping, ii) rounding operation, and iii) warping competition. To better understand these sub-steps, an example is shown in Fig. 1, where the left reference view

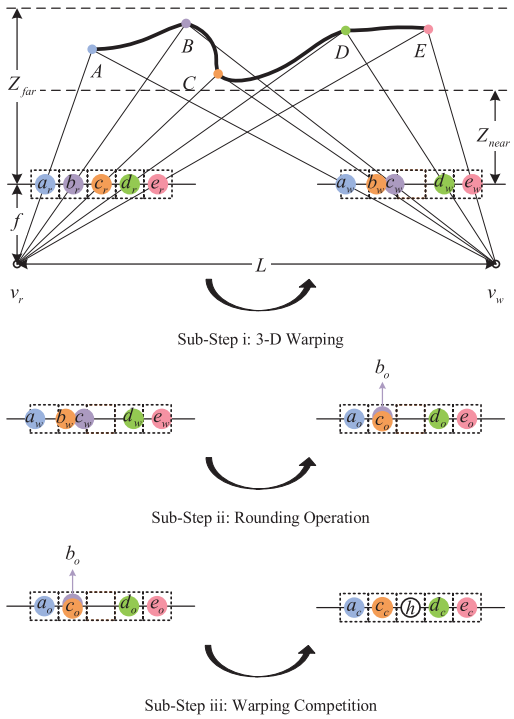

Fig. 1. Illustration of the sub-steps of the forward warping step. The size of the dashed squares represent the precision of the rounding operation.

v_r is warped to the virtual view v_w. The warping method from the right reference view to the virtual view is similar.

During 3D warping, all the texture pixels a_r, b_r, …in the left reference view are processed sequentially and projected to a_w, b_w, …in the left warped view, which correspond to points A, B, …in the physical scene. The initial floating-point disparity value between a_r and a_w, i.e., the difference between their horizontal coordinates, can be formulated as follows

$$\delta(d) = \widetilde{l}_r(a_r) - l_w(a_w) = \frac{fLd}{255}\left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}}\right) + \frac{fL}{Z_{far}}, \quad (1)$$

where $\widetilde{l}_r(a_r)$ is the horizontal coordinate of a_r in the left reference view, which is an integer, while the horizontal coordinate of a_w is usually a floating-point number due to the operations in the right-hand side of this equation, denoted as $l_w(a_w)$. f is the camera focal length. d is the depth value of point A in the scene. L is the baseline distance between reference view and virtual view. Z_{near} and Z_{far} define the depth range of the physical scene.

After getting the locations of a_w, b_w, … from Eq. (1), a rounding operation is applied so that the floating-point locations of a_w, b_w, … will be rounded to the desired precision, such as full-pixel, half-pixel, or quarter-pixel, denoted by points a_o, b_o, …

After forward warping and rounding the entire reference frame to the virtual view, if several pixels appear in the same location in the warped view, such as b_o and c_o in Fig. 1, warping competition will happen, as studied in [16]–[18]. Generally, the pixel c_o with the largest depth value (closest to the camera) will be selected as the winner in the warped view, denoted as c_c in Fig. 1. If there is no warping competition,

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

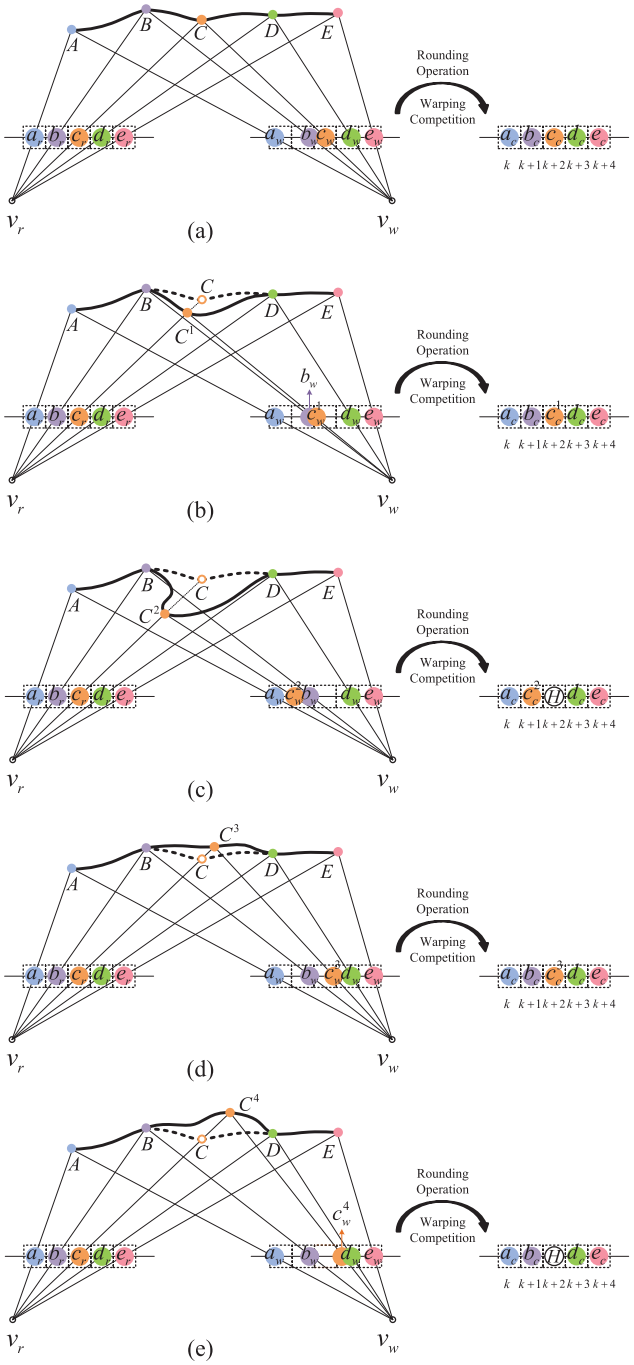

Fig. 2. Different cases of the 3D warping.

the rounded result is used directly as the final result, e.g., $a_c = a_o$. Besides, rounding operations may cause some disocclussion holes in the virtual view, such as point h in Fig. 1.

The process above is repeated to warp the right reference view to the virtual view. After that, the left and right warped views will be blended into the final virtual view so that most of the disocclusion regions will be filled by each other. In general, three cases could happen during the blending stage: i) if the blended pixels are available in both warped views, the two values will be linearly combined [19]; ii) if the blended pixel is only available in one of the warped views, it is used directly. iii) if the blended pixels are not available in both warped views, this location in the virtual view will be regarded as a hole, which will be filled by some inpainting methods [20]–[22]. As mentioned in [9], the third case is quite infrequent (hole areas are around 1 percent of the whole virtual view).

B. Analysis on Depth-Change-Caused View Synthesis Distortion

As reviewed above, depth information plays an important role in the DIBR. In this section, we will conduct a thorough analysis to investigate the impact of the change of the reference depth value to the quality of the synthesized virtual view. We will show that frame-based 3D warping can only yield an approximate prediction of the virtual view with limited accuracy. More examples are shown in Fig. 2 (a), where the rounded horizontal locations in the virtual views are denoted by k, $k+1$, etc. Without loss of generality, we use point c_r as an example to analyze different scenarios.

Case 1: After source coding, the reconstructed depth value of c_r is only slightly larger than its original value, the corresponding point in the scene will becomes closer, as shown by C^1 in Fig. 2 (b). After 3D warping, it will be warped to c_w^1 in v_r. After rounding operation and warping competition, it is very likely that its location is still at $k+2$. In this case, the depth error for c_r will not cause any view synthesis distortion.

Case 2: If the reconstructed depth value of c_r is larger than a threshold, the corresponding point in the scene, i.e., C^2 in Fig. 2(c), will be warped to c_w^2, which will be closer to b_w. After rounding, it could lead to warping competition with b_w. In this example, since C^2 is closer to the camera than B, c_w^2 will be the winner at location k+1. Meanwhile, a hole will appear at location $k+2$. Therefore, view synthesis distortion will occur at locations $k+1$ and $k+2$.

Case 3: If the reconstructed depth value of c_r is slightly smaller than its original value, as shown in Fig. 2 (d), the result is similar to Case 1, i.e., no view synthesis distortion will appear.

Case 4: If the reconstructed depth value of c_r is smaller than a threshold, as shown in Fig. 2 (e), then after 3D warping and rounding, it could lead to competition with d_w at location $k+3$. In this example, d_w will be the winner since D is closer to the camera than C^4. Eventually, d_c is still the correct result in the warped view. Therefore, there is no distortion at location $k+3$. Meanwhile, a hole will appear at location $k+2$, leading to view synthesis distortion at location $k+2$.

From Cases 1 and 3 above, it can be seen that small changes in the reference depth values will not cause any warping error. Only large depth changes could lead to warping errors in the virtual view, i.e., the warped positions are changed, as shown in Cases 2 and 4. However, not all warping errors will win the warping competition and eventually manifest themselves as synthesis distortions, as shown in Case 4.

Most VSDE algorithms firstly use the depth changes to predict the floating-point disparity changes to model the distortion propagation from the reference view to the warped view. To achieve this, Eq. (8) in [15] is derived according

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

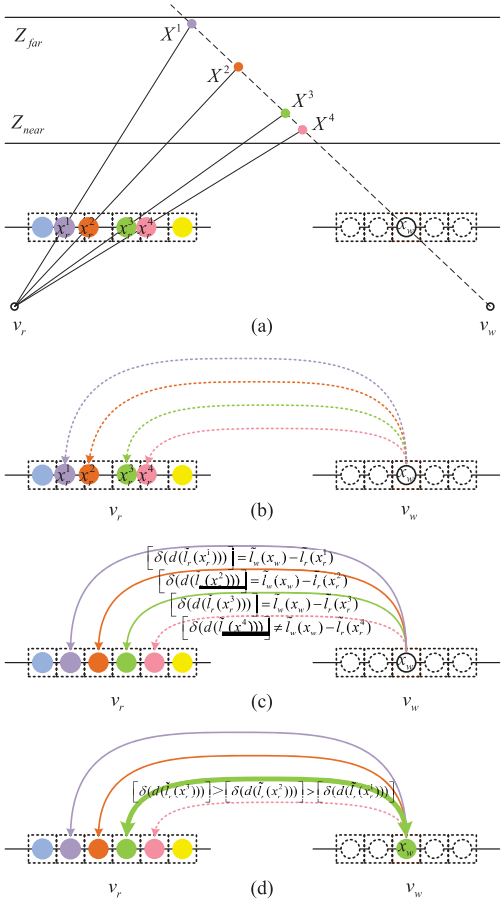

Fig. 3. Illustration of the coarse-to-fine backward prediction method.

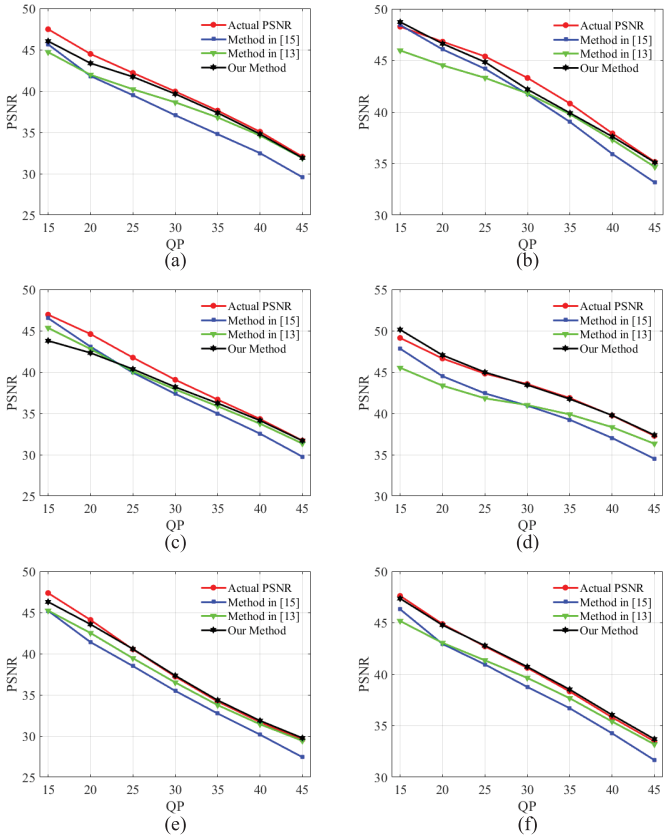

Fig. 4. The comparation of actual PSNR and estimated PSNR with different methods for the first frame with different texture/depth QP pairs. (a) - (f) are the results of sequences *BookArrival*, *Kendo*, *PoznanStreet*, *PoznanHall2*, *UndoDancer*, *GT-Fly*, respectively.

to 3D warping. Then, all floating-point disparity changes of the entire frame are indiscriminately accumulated to estimate the expectation of the disparity changes, which includes the small changes discussed in Case 1 and Case 3, even though these small changes will not cause any distortion. Finally, the expectations of the disparity changes are roughly associated with the texture values of the synthesized view. For example, once the current pixel is warped to a new position due to depth changes, its texture value will be given to the new position, regardless of if warping competition happen or not. However, this is not always true, as discussed in Case 4. Therefore, only considering 3D warping in modeling depth-change-caused view synthesis distortion cannot capture all these cases above, which makes the accuracy of these VSDE algorithms dissatisfactory. To improve their accuracy, both rounding operation and warping competition should be considered as well.

III. Pixel-Level View Synthesis Distortion Estimation Method

In this section, we develop a novel pixel-level VSDE method (PL-VSDE) to better serve the emerging region-based 3D video coding schemes. To this end, we first review the definition of frame-level view synthesis distortion. After that, the pixel-level view synthesis distortion is defined accordingly. Then, we estimate the pixel-level view synthesis distortion by using a backward prediction method.

As studied in [13] and [15], the frame-level view synthesis distortion is defined as the following Mean Squared Error (MSE) over an entire virtual view frame:

$$\mathrm{MSE} = \frac{1}{H \cdot W} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \left(T(i,j) - \bar{T}(i,j)\right)^2, \tag{2}$$

where H and W are the height and width of the image. $T(x,y)$ and $\bar{T}(x,y)$ denote the texture values of the two pixels with the same location (x,y) in two virtual views, which will be synthesized by the original reference views and the reconstructed reference views, respectively.

To obtain the result of Eq. (2), the DIBR algorithm described in Sec. II is actually carried out, which is not efficient for region-based 3D video coding. The reasons are as follows: i) During forward warping step, it indexes through all pixels in the reference views to find their corresponding locations in the virtual view. Before finishing the forward warping step, we do not know which reference pixel will be warped to the POI (or if a reference pixel will be warped to the POI or not). ii) During blending step, the texture value of each pixel in the virtual view is obtained by making a weighted calculation, even for pixels that do not belong to the POIs.

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

Therefore, in region-based video coding, to reduce the complexity, instead of warping the reference pixels to the virtual view, we use a backward prediction method, i.e., we start from the desired location in the POIs of the virtual view, and locate their corresponding pixels' locations in the reference views. After that, only the located pixels, which will be warped to the POIs, are used to make the weighted calculation accordingly. Hence we define pixel-level view synthesis distortion to be the Squared Error between $T(i, j)$ and $\bar{T}(i, j)$.

$$\mathrm{SE}((i, j)) = (T(i, j) - \bar{T}(i, j))^2, \tag{3}$$

where (i, j) is the location of the POI.

Next, we will solve Eq. (3) by expanding $T(i, j)$ and $\bar{T}(i, j)$ according to the principle of blending step.

For $T(i, j)$, we have

$$T(i,j) = \begin{cases} \alpha T_l^w(i,j)+(1-\alpha)T_r^w(i,j), & T_l^w(i,j)\notin\emptyset,\ T_r^w(i,j)\notin\emptyset \\ T_l^w(i,j), & T_l^w(i,j)\notin\emptyset,\ T_r^w(i,j)\in\emptyset \\ T_r^w(i,j), & T_l^w(i,j)\in\emptyset,\ T_r^w(i,j)\notin\emptyset \\ \text{Inpainting}, & T_l^w(i,j)\in\emptyset,\ T_r^w(i,j)\in\emptyset \end{cases}, \tag{4}$$

where $T_l^w(i, j)$ and $T_r^w(i, j)$ are the texture values of pixels from the left and right warped views, which are generated with the original data. α ($\alpha \in [0, 1]$) is a scaling factor that is obtained from the location of the virtual view. $\emptyset$ denotes the hole in the warped view.

Similarly, for $\bar{T}(i, j)$, we have

$$\bar{T}(i,j) = \begin{cases} \alpha \bar{T}_l^w(i,j)+(1-\alpha)\bar{T}_r^w(i,j), & \bar{T}_l^w(i,j)\notin\emptyset,\ \bar{T}_r^w(i,j)\notin\emptyset \\ \bar{T}_l^w(i,j), & \bar{T}_l^w(i,j)\notin\emptyset,\ \bar{T}_r^w(i,j)\in\emptyset \\ \bar{T}_r^w(i,j), & \bar{T}_l^w(i,j)\in\emptyset,\ \bar{T}_r^w(i,j)\notin\emptyset \\ \text{Inpainting}, & \bar{T}_l^w(i,j)\in\emptyset,\ \bar{T}_r^w(i,j)\in\emptyset \end{cases}, \tag{5}$$

where $\bar{T}_l^w(i, j)$ and $\bar{T}_r^w(i, j)$ are the predicted texture values of pixels from the left and right warped views, which are generated with the reconstructed data. Since the last case is quite infrequent, it is usually ignored in most of VSDE methods. Eq. (4) and (5) are also used in frame-level VSDE methods such as [13], [15], [16].

According to Eq. (4) and (5), given the locations of POIs in the virtual views, their associated pixels in the warped views will be located since they share the same locations. The remaining problem is how to predict their associated pixels in the reference views. To handle this, a coarse-to-fine backward prediction method (CFBP) is developed next. With the CFBP, the relationship between each pixel in the *left (or right)* warped view and its associated pixel in the *left (or right)* reference view can be built, denoted by R^l (or R^r). Specifically, the R^l and R^r of the *original data* (R^l_O and R^r_O) can be generated first. Similarly, the R^l and R^r of the *reconstructed data* (R^l_R and R^r_R) can be obtained as well. Then, the required predictions can be achieved accordingly.

It should be noted that in the DIBR with general mode setting, a technique called *backward warping* was proposed [17], [23], [24], where instead of warping the reference texture images, the reference depth images are first projected to the warped view, followed by blending and hole filling. After the warped depth image is obtained, it will be utilized to find each texture pixel in the warped view from the corresponding texture pixel in the reference view. However, the backward warping is only applicable when the depth information of the warped depth map is known. In our case, since only the locations of POIs in the warped view can be obtained according to Eq. (4) and (5), but their depth information is unavailable, the backward warping cannot be used to solve our backward prediction issue.

IV. Coarse-to-Fine Backward Prediction Method

Our coarse-to-fine backward prediction method is firstly presented when the 3D warping, rounding operation and warping competition in view synthesis are all taken into account. Then, an optimization is designed to speed up our method. The framework of coarse-to-fine backward prediction method contains two steps: i) Finding the preliminary candidates in the reference view according to the epipolar line theory, ii) selecting the intermediate candidates from the preliminary candidates, and deciding the final candidate from the intermediate candidates. The located final candidate pixel in the reference view is the pixel which is warped to the POI. For simplification, only R^l_O is elaborated in this part, and other relationships can be obtained similarly.

A. Finding Preliminary Candidates

As discussed in Section II, if the POI in the warped view is not a hole, it could correspond to multiple pixels in the reference view, as further is illustrated in Fig. 3 (a). Let x_w be the POI in the left warped view v_w with location $\widetilde{l_w}(x_w)$ (x_w has an integer horizontal coordinate since it is a POI, specified in the warped view). Since its depth value is unconfirmed, it may correspond to any point on the line between Z_{near} and Z_{far} such as points X_1, X_2, …Their associated pixels in the left reference view v_r are x_r^1, x_r^2, …, with location $l_r(x_r^1)$, $l_r(x_r^2)$, …, respectively. This is a special case of the epipolar line in multiview geometry [25], [26]. In this paper, x_r^1, x_r^2, …are defined as the preliminary candidates of x_w. Therefore, the locations of preliminary candidates can be confined within a range, denoted as candidate search range, which can be determined as follows

$$\begin{cases} s_{min} = \widetilde{l_w}(x_w) - \lambda_1 \\ s_{max} = \widetilde{l_w}(x_w) - \lambda_2 \\ \lambda_1 = \left[fL\left(\frac{d_{max}}{255}\left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}}\right) + \frac{1}{Z_{far}}\right)\right] \\ \lambda_2 = \left[fL\left(\frac{d_{min}}{255}\left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}}\right) + \frac{1}{Z_{far}}\right)\right] \end{cases}, \tag{6}$$

where d_{min} and d_{max} are the minimum and maximum depth values, namely 0 and 255. $[\cdot]$ denotes the rounding operation. s_{min} and s_{max} are the lower and upper bounds of the candidate search range. λ_1 and λ_2 could be regarded as a pair of constants, since they are defined by the camera parameters (f, L, Z_{near}, and Z_{far}). Therefore, the candidate search range of

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

each POI can be easily obtained by a fixed shifting process. In the case of Fig. 3 (a), the relationship between the POI in the left warped view and each preliminary candidate in the left reference view is represented with a dot line in Fig. 3 (b).

B. Selecting the Intermediate and Final Candidates

In this step, the preliminary candidates with suitable depths that can be projected to the location of the POI will be selected, which are defined as the intermediate candidates. In the worst situation, the candidate search range will be divided into 256 floating-point horizontal coordinates, corresponding to 256 depth values, which will be considered during intermediate candidate selection. For simplicity purpose, only the integer horizontal coordinates in the candidate search range are considered. We use a mask function $M(\cdot)$ of x_r^m to select all the intermediate candidate from the candidate search range, which can be represented as follows

$$M(\widetilde{l}_r(x_r^m)) = \begin{cases} 1, & \text{if } [\delta(d(\widetilde{l}_r(x_r^m)))] = \widetilde{l}_w(x_w) - \widetilde{l}_r(x_r^m) \\ 0, & \text{otherwise} \end{cases}, \quad (7)$$

where x_r^m is the m^{th} preliminary candidate with the integer horizontal coordinate $\widetilde{l}_r(x_r^m)$ ($\widetilde{l}_r(x_r^m) = s_{min} + m - 1$), where $\widetilde{l}_r(x_r^m) \in [s_{min}, s_{max}]$. According to the $\widetilde{l}_r(x_r^m)$, $d(\widetilde{l}_r(x_r^m))$ can be easily obtained from the reference depth image.

After this, another mask function $H(\cdot)$ will be used to check if the current POI in the virtual view is a hole or not, which can be formulated as follows

$$H(x_w) = \begin{cases} 1, & \text{if } \sum_{\widetilde{l}_r(x_r^m) \in [s_{min}, s_{max}]} M(\widetilde{l}_r(x_r^m)) = 0 \\ 0, & \text{otherwise.} \end{cases}. \quad (8)$$

If the sum of $M(\widetilde{l}_r(x_r^m))$ ($\widetilde{l}_r(x_r^m) \in [s_{min}, s_{max}]$) is 0, x_w will be treated as a hole. Otherwise, the final candidate selection will happen. It should be noticed that if the sum of $M(\widetilde{l}_r(x_r^m))$ is larger than 1, several intermediate candidates will be projected to the POI, thus warping competition will happen. This case can be shown in Fig. 3 (c), where the points locating at $\widetilde{l}_r(x_r^1)$, $\widetilde{l}_r(x_r^2)$ and $\widetilde{l}_r(x_r^3)$ are the intermediate candidates that satisfy the condition, and the relationship between the POI and each intermediate candidate is represented by a solid line.

By the warping competition rule, the intermediate candidate with the largest depth value (closest to the camera) will be selected as the final candidate. Therefore, we have

$$\widetilde{l}_r(x_r) = \underset{\widetilde{l}_r(x_r^m) \in [s_{min}, s_{max}]}{\arg\max} d(\widetilde{l}_r(x_r^m)), \quad (9)$$

where x_r is the final candidate, and its location is represented with $\widetilde{l}_r(x_r)$. In Fig. 3 (d), we assume x_r^3 has the largest depth value, which is selected as our final candidate x_r. The relationship between our final candidate and the POI is represented by a thick green line, namely R_O^l. Then, the backward prediction from x_w to x_r is achieved, which means the texture value of x_w is projected from x_r.

However, if x_w is a hole, 128 will be given as its texture value [16]. Therefore, for the texture value of the pixel of interest in the original left warped view, namely $T_l^w(\widetilde{l}_w(x_w), j)$, we have

$$T_l^w(\widetilde{l}_w(x_w), j) = \begin{cases} 128, & \text{if } H(x_w) = 1 \\ T_l^r(\widetilde{l}_r(x_r), j), & \text{otherwise.} \end{cases} \quad (10)$$

Similarly, we can obtain the $T_r^w(\widetilde{l}_w(x_w), j)$, $\bar{T}_l^w(\widetilde{l}_w(x_w), j)$, $\bar{T}_r^w(\widetilde{l}_w(x_w), j)$ according to R_O^r, R_R^l, and R_R^r, which are the texture values of the POI in the original right warped view, the reconstructed left warped view, and the reconstructed right warped view, respectively.

After integrating the CFBP method into the PL-VSDE, we can directly estimate the distortion of the POIs in the synthesized view at the pixel level.

The biggest difference between the traditional backward warping and the proposed CFBP method is that we cannot carry out the 3D warping for each POI from virtual view to the reference view due to the lack of the depth information. Instead, we first generate the candidate search range by a fixed shifting in Eq. (6). After taking the warping rounding operation and warping competition into account, the final candidate can be accurately found.

C. Speedup of Coarse-to-Fine Backward Prediction Method

In this subsection, table-lookup method and the warping property are used to speed up our CFBP method.

Firstly, a table-lookup method is used to speed up the rounding calculation of disparity, namely $[\delta(\cdot)]$ in Eq. (7). $[\delta(\cdot)]$ could be regarded as a rounding operation of Eq. (1), which can be rewritten as

$$\begin{cases} [\delta(d(\widetilde{l}_r(x_r^m)))] = [c_1 \cdot d(\widetilde{l}_r(x_r^m)) + c_2] \\ c_1 = \dfrac{f \cdot L}{255} \cdot (\dfrac{1}{Z_{near}} - \dfrac{1}{Z_{far}}) \\ c_2 = \dfrac{f \cdot L}{Z_{far}} \end{cases}, \quad (11)$$

where $d(\widetilde{l}_r(x_r^m)) \in [0, 255]$. Therefore, the relationship between $d(\widetilde{l}_r(x_r^m))$ and $[\delta(\cdot)]$ can be pre-calculated in a lookup table. It can then be called when Eq. (7) is calculated.

Besides, function $[\delta(\cdot)]$ could be regarded as a non-decreasing piece-wise constant function, since c_1 and c_2 are positive constants for the left reference view. Therefore, we can conclude that bigger $[\delta(\cdot)]$ will lead to bigger depth value $d(\widetilde{l}_r(x_r^m))$. However, for the right reference view, the opposite is true, namely the pixel with larger disparity has smaller depth value, as exhibited in Fig. 3 (a), where if x_w is warped to the leftmost point x_r^1 with larger disparity, it has to be the farther point X_1 with smaller depth value.

According to this warping property, the CFBP can be further simplified as follows: For the left reference view, after the first step of our coarse-to-fine backward prediction algorithm, we can decide the final candidate by searching from the rightmost to the leftmost preliminary candidate, and selecting the first candidate that satisfies the condition in Eq. (7), since it has the largest depth value. If no preliminary candidate satisfies the condition, the POI will be treated as a hole. For the right reference view, the order of picking up the final candidate is the opposite.

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

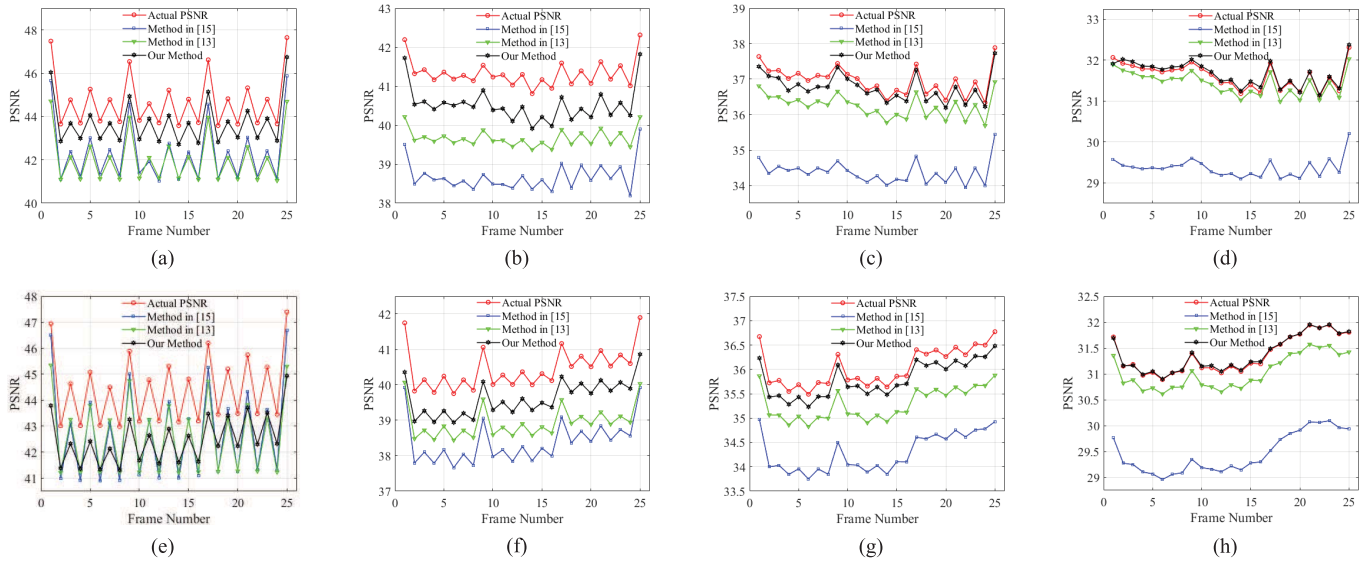

Fig. 5. The comparation between actual PSNRs and three estimated PSNRs provided by different methods, where the first 25 successive frames are used for evaluation. (a) - (d) are the results of sequence *BookArrival*, with QP(15, 24), QP(25, 34), QP(35, 42), QP(45, 48). (e) - (h) are the results of sequence *PoznanStreet*, with QP(15, 24), QP(25, 34), QP(35, 42), QP(45, 48).

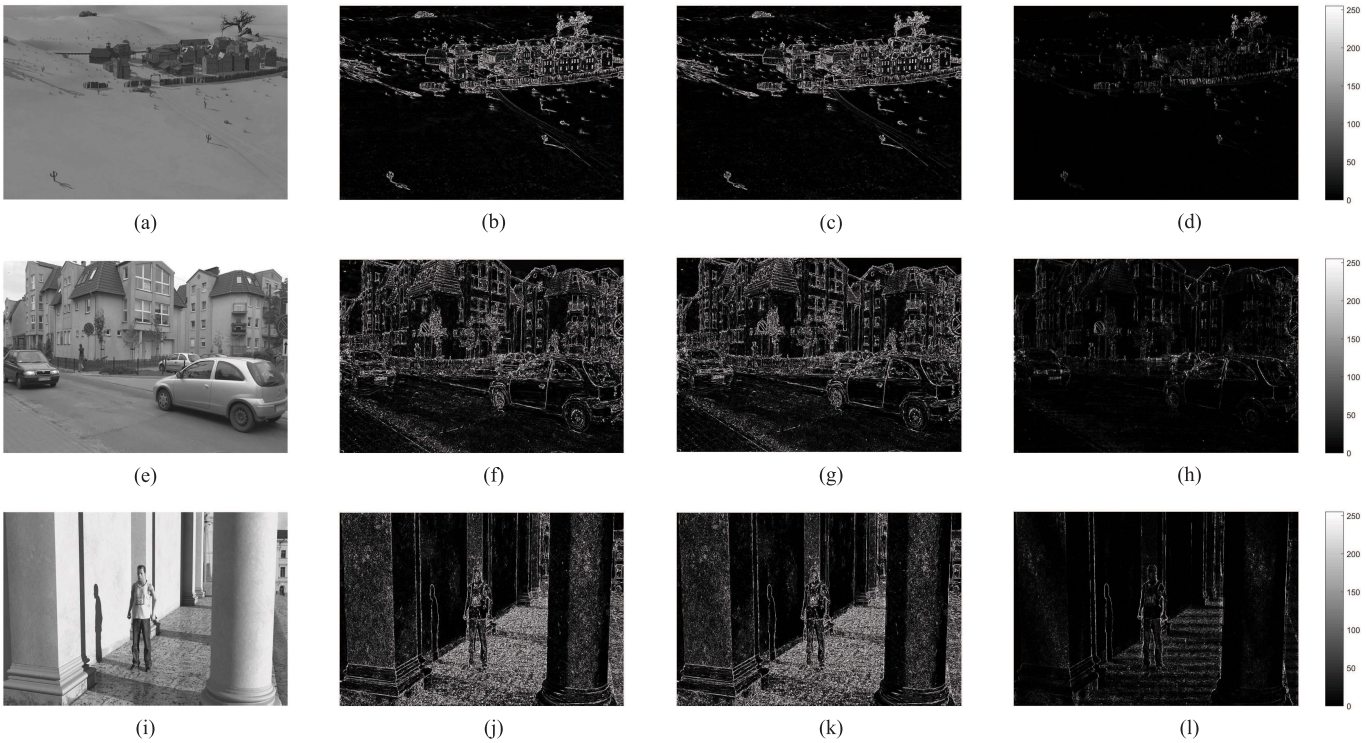

Fig. 6. The comparation between the actual SEs and our estimated SEs for each pixel. (a), (e), and (i) are the first frames of the synthesized vew in *GT-Fly*, *PoznanStreet*, and *UndoDancer*. (b), (f), and (j) are the actual SEs between two synthesized views, which are rendered using the original data and reconstructed data. (c), (g), and (k) are the estimated SEs, which is generated by our method. (d), (h), and (l) are the absolute difference between the actual SEs results and our estimated SEs results. For a fair comparation, results in this figure are reasonably scaled between 0 to 255. For instance, for the results in (b) and (c), we firstly find the largest SEs in these two images, then this value and 0 will be the upper and lower bounds. Finally, these two images are scaled between 0 to 255 according these bounds. (d) is the absolute difference between these two scaled images.

V. Experimental Results

In this section, various simulation results are presented, which show that our method has better performance than methods in [15] and [13], which are regarded as state-of-the-art methods in efficiency and accuracy for frame-based VSDE.

Two main performance metrics are used in our experiments: the accuracy measured by PSNR, and the efficiency measured by the running time of each method.

The test multiview sequences used for our evaluation are recommended in the Common Test Conditions (CTC) of the

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

TABLE I
ESTIMATION ACCURACY AMONG THREE METHODS (THE AVERAGE RESULTS OF THE FIRST 25 FRAMES)

Sequences	QPt		Acutual	Method in [15]		Method in [13]		Proposed Method	
	Texture	Depth	PSNR	$PSNR_1$	$\Delta PSNR_1$	$PSNR_2$	$\Delta PSNR_2$	$PSNR_3$	$\Delta PSNR_3$
BookArrival	15	24	44.682	42.313	2.368	42.051	2.631	43.668	**1.013**
	20	29	43.116	40.359	2.756	40.869	2.246	42.180	**0.936**
	25	34	41.328	38.681	2.647	39.667	1.661	40.505	**0.823**
	30	39	39.258	36.560	2.699	38.133	1.125	38.942	**0.316**
	35	42	36.957	34.377	2.579	36.243	0.714	36.764	**0.193**
	40	45	34.422	32.099	2.323	34.026	0.396	34.374	**0.048**
	45	48	31.625	29.367	2.257	31.432	0.192	31.673	**0.048**
	Average		38.770	36.251	2.519	37.489	1.281	38.301	**0.482**
Kendo	15	24	46.464	45.726	**0.738**	44.606	1.858	44.530	1.934
	20	29	45.223	43.970	**1.253**	43.419	1.804	43.344	1.879
	25	34	43.678	42.042	1.636	42.215	**1.463**	42.136	1.542
	30	39	41.473	39.533	1.940	40.522	**0.951**	40.327	1.146
	35	42	39.032	37.100	1.932	38.370	0.662	38.410	**0.622**
	40	45	36.349	34.434	1.915	35.885	0.464	36.075	**0.274**
	45	48	33.610	31.803	1.806	33.270	0.339	33.564	**0.045**
	Average		40.833	39.230	1.603	39.755	1.077	39.769	**1.063**
PoznanStreet	15	24	44.419	42.756	**1.662**	42.642	1.777	42.487	1.932
	20	29	42.366	40.196	2.170	40.513	1.853	41.038	**1.328**
	25	34	40.461	38.384	2.076	38.938	1.523	39.619	**0.841**
	30	39	38.217	36.343	1.873	37.156	1.061	37.720	**0.497**
	35	42	36.040	34.286	1.754	35.284	0.757	35.801	**0.239**
	40	45	33.821	32.070	1.750	33.295	0.526	33.698	**0.123**
	45	48	31.365	29.458	1.908	31.025	0.340	31.378	**0.013**
	Average		38.098	36.213	1.885	36.979	1.119	37.392	**0.710**
PoznanHall2	15	24	46.026	44.035	1.991	43.257	2.769	45.804	**0.222**
	20	29	44.556	42.123	2.433	41.980	2.576	44.180	**0.376**
	25	34	43.612	41.046	2.566	41.324	2.288	43.131	**0.481**
	30	39	42.497	39.729	2.767	40.578	1.919	41.981	**0.515**
	35	42	41.010	38.121	2.890	39.452	1.558	40.627	**0.383**
	40	45	39.094	36.347	2.747	37.899	1.195	38.944	**0.150**
	45	48	36.769	33.948	2.821	35.919	0.850	36.795	**0.025**
	Average		41.938	39.336	2.602	40.058	1.879	41.637	**0.308**
UndoDancer	15	24	44.597	42.219	2.377	43.209	1.388	43.634	**0.963**
	20	29	41.689	39.111	2.578	40.663	1.026	41.207	**0.482**
	25	34	38.653	36.393	2.260	37.954	0.699	38.415	**0.238**
	30	39	35.887	33.882	2.005	35.479	0.408	35.873	**0.014**
	35	42	33.401	31.643	1.759	33.162	0.240	33.519	**0.117**
	40	45	31.191	29.279	1.912	31.073	**0.118**	31.311	0.120
	45	48	29.131	26.834	2.297	29.127	**0.004**	29.256	0.125
	Average		36.364	34.194	2.170	35.810	0.555	36.174	**0.294**
GT-Fly	15	24	45.099	43.229	1.871	43.315	1.784	43.900	**1.199**
	20	29	43.224	41.190	2.034	41.768	1.455	42.282	**0.941**
	25	34	41.301	39.402	1.899	40.154	1.147	40.725	**0.576**
	30	39	39.436	37.640	1.795	38.622	0.814	39.075	**0.361**
	35	42	37.533	35.877	1.656	36.944	0.589	37.504	**0.028**
	40	45	35.291	33.722	1.568	34.924	0.366	35.590	**0.300**
	45	48	33.049	31.341	1.708	32.791	**0.258**	33.541	0.493
	Average		39.276	37.486	1.790	38.360	0.916	38.946	**0.557**

Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) [27], including the HHI's BookArrival (view 8 and 10 are used to render view 9) [28], the Nagoya University's Kendo (view 1 and 3 are used to render view 2) [29], the Poznan University of Technology's PoznanStreet (view 4 and 5 are used to render view 4.5) and PoznanHall2 (view 6 and 7 are used to render view 6.5) [30], the Nokia Corporation's UndoDancer (view 1 and 5 are used to render view 3) [31] and GT-Fly (view 5 and 9 are used to render view 7) [32].

The 3D video coding test platform version 9.2 (3D-HTM9.2) with configurations defined by JCT-3V common test condition is used to compress the sequences above with different texture/depth QP pairs [33], namely (15, 24), (20, 29), (25, 34), (30, 39), (35, 42),(40, 45), and (45, 48). The VSRS-1D-fast software of JCT-3V are used to synthesize virtual views [33]. Our proposed method is implemented in Matlab R2017a. All these experiments are tested on a laptop with Intel(R) Core(TM) i7-6700HQ CPU, 16.00GB memory, and 64-bit Windows Operating System.

A. Evaluation of the Accuracy at the Frame Level

In this subsection, we first apply our method to the entire frame at the frame level, and evaluate the accuracy of all these three methods. The estimated PSNRs by these three methods and the actual PSNRs of the synthesized views are compared in Fig. 4 and TABLE I. TABLE I shows the average results of the first 25 frames for each sequence. Fig. 4 shows the results of the first frame for each sequence. It should be noticed that $\triangle PSNR_i$ in TABLE I is the absolute value of the difference between PSNR and $PSNR_i$, where i is the index of the three methods. The smallest $\triangle PSNR_i$ means the best performance (highlighted in bold in TABLE I). According to the results,

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

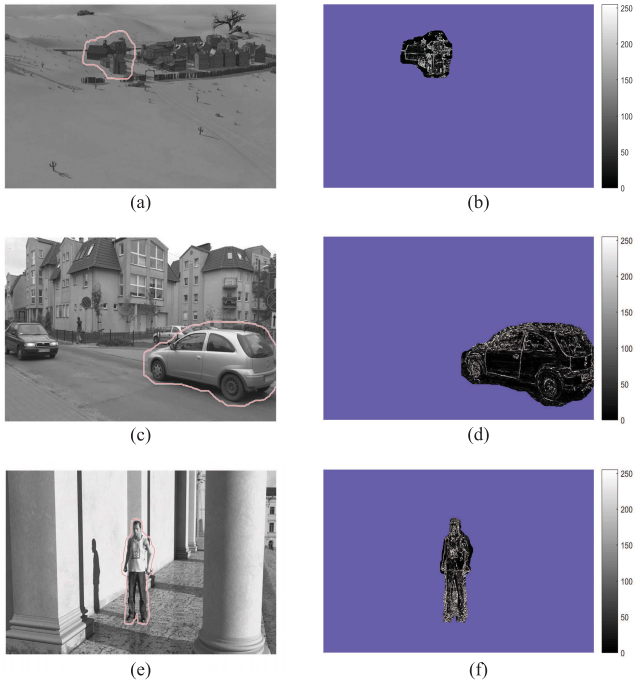

Fig. 7. Illustration of the distortion estimation of the POIs in the synthesized view with our method. The POIs are specified in (a), (c), and (e). The estimated SEs of the POIs are shown in (b), (d), and (f).

we can easily conclude that the accuracy of our method is substantially better than [15] and [13]. This verifies the benefit of pixel-level distortion estimation over frame-level estimation.

We also compare the actual PSNRs with three estimated PSNRs provided by different methods for different frames. The results of sequences *BookArrival* and *PoznanStreet* with different texture/depth QP pairs are shown in Fig. 5. It could be seen that our method is the closest to the actual PSNRs, especially for larger QP pairs.

B. Evaluation of the Accuracy at the Pixel Level

Our method is the only one that can directly estimate the distortion of POIs in the synthesized view at pixel-level accuracy. This evaluation can be divided into two parts. The first part is to verify that our method can achieve the VSDE at pixel-level accuracy. The second part is to verify that only the distortion of the POIs is estimated in our method.

For the first part, we estimate each pixel's Squared Error (SE) (which cannot be achieved by the traditional frame level methods), which is shown in Fig. 6. The images in the first column ((a), (e), and (i)) are the first frames of the synthesized views in three different sequences. The second and third columns are the actual SEs of each pixel and its associated estimated SEs generated by our methods. Their absolute difference images are shown in the last column. It can be seen that our method can accurately achieve a VSDE at pixel-level accuracy. Besides, according to our results, we can locate the distortion regions accurately as well. However, our method still has some errors, since the hole filling process is ignored, e.g., we only use 128 to estimate the texture value at hole. Besides, some pre/post processing in view synthesis [17], [27], such as depth boundary filter, similarity enhancement, are also ignored in our method. All these factors will be studied in our future work to further improve the performance.

TABLE II

RUNNING TIME COMPARISON AMONG THREE METHODS (THE AVERAGE RESULTS OF THE FIRST 25 FRAMES)

Sequences	QPt		[15]	[13]	**Proposed**
	Texture	Depth	Time(s)	Time(s)	Time(s)
BookArrival	15	24	13.878	0.359	1.175
	20	29	13.807	0.359	1.170
	25	34	13.778	0.349	1.169
	30	39	13.682	0.352	1.172
	35	42	13.887	0.350	1.175
	40	45	13.810	0.355	1.174
	45	48	13.751	0.352	1.169
	Average		13.799	0.354	1.172
Kendo	15	24	12.626	0.358	1.293
	20	29	12.721	0.348	1.297
	25	34	12.721	0.347	1.288
	30	39	12.721	0.350	1.324
	35	42	12.716	0.348	1.304
	40	45	12.716	0.353	1.292
	45	48	12.769	0.350	1.292
	Average		12.713	0.351	1.299
PoznanStreet	15	24	73.948	0.733	4.500
	20	29	73.861	0.732	4.479
	25	34	73.446	0.730	4.498
	30	39	74.631	0.719	4.461
	35	42	73.775	0.730	4.491
	40	45	74.254	0.716	4.473
	45	48	74.255	0.729	4.498
	Average		74.024	0.727	4.486
PoznanHall2	15	24	73.050	0.730	5.408
	20	29	73.050	0.734	5.397
	25	34	73.299	0.728	5.395
	30	39	74.041	0.728	5.433
	35	42	73.414	0.713	5.415
	40	45	73.698	0.722	5.380
	45	48	73.453	0.719	5.398
	Average		73.429	0.725	5.404
UndoDancer	15	24	73.906	0.756	4.603
	20	29	73.697	0.758	4.629
	25	34	73.232	0.762	4.644
	30	39	73.724	0.781	4.640
	35	42	73.606	0.771	4.585
	40	45	73.804	0.785	4.646
	45	48	73.148	0.782	4.648
	Average		73.588	0.771	4.628
GT-Fly	15	24	74.148	0.729	3.185
	20	29	73.736	0.705	3.184
	25	34	73.446	0.708	3.212
	30	39	73.851	0.730	3.198
	35	42	72.938	0.727	3.205
	40	45	73.451	0.723	3.198
	45	48	73.848	0.724	3.208
	Average		73.631	0.721	3.199

For the second part, Fig. 7 shows some POIs and their estimated SEs, where we firstly select the POIs in the synthesized view, which could be a person or an object, as indicated by the image patch inside the red contour in the synthesized view. Then, only the distortions of the POIs are estimated at pixel-level accuracy in our method rather than the whole frame, as shown by the second columns in Fig. 7, demonstrating the flexibility of the pixel-level method.

C. Evaluation of the Efficiency

We compare the complexity of these three methods in this subsection when performing frame-level distortion estimation.

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

The average running time of the first 25 frames are listed in TABLE II. The unit is second. From this perspective, method in [13] is the fastest method. However, both method in [15] and our method are friendly for parallel processing. For these two methods, each row can be independently performed, e.g., by a separate thread of the CPU or GPU. Besides, our method can be processed in parallel at the pixel level, which will further save the execution time. Moreover, considering that our method is designed for the region-based 3D video coding, only the distortion of the POIs in the virtual view will be estimated, which can also save time compared to frame-based methods. With these flexible designs, our method can easily achieve up to 10 times speedup during parallel processing, which is fast enough to beat method in [13].

VI. Conclusion

In this paper, we develop a pixel-level distortion estimation of the POI in the virtual view to better serve the emerging region-based 3D video coding algorithms. The pixel-level view synthesis distortion is firstly defined. Then, the CFBP method is developed to estimate the pixel-level view synthesis distortion. With such method, we can easily build the relationship between each pixel in the warped view and its associated pixel in the reference view. The CFBP achieves an accurate backward prediction by taking the 3D warping, rounding operation and warping competition into account. To speed up our coarse-to-fine backward prediction method, a table-lookup method and a warping property are exploited. After integrating these two models together, only the necessary pixels in the reference view, which are related to the POI in the warped view, are used to calculate the distortion at pixel-level accuracy. Experimental results demonstrate that our proposed method has obvious advantages in both accuracy and efficiency compared with the latest frame level VSDE methods. Moreover, our method is the first VSDE algorithm designed for the region-based 3D video coding schemes.

References

[1] M. Irani, T. Hassner, and P. Anandan, "What does the scene look like from a scene point?" in *Proc. Eur. Conf. Comput. Vis.*, May 2002, pp. 883–897.

[2] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *Proc. IEEE Int. Conf. Image Process.*, San Antonio, TX, USA, Sep. 2007, pp. I-201–I-204.

[3] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," *Proc. SPIE*, vol. 5291, pp. 93–105, May 2004.

[4] M. Domanski *et al.*, "3D video compression by coding of disoccluded regions," in *Proc. 19th IEEE Int. Conf. Image Process.*, Orlando, FL, USA, Sep. 2012, pp. 1317–1320.

[5] M. Domanski *et al.*, "High efficiency 3D video coding using new tools based on view synthesis," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3517–3527, Sep. 2013.

[6] M. S. Farid, M. Lucenteforte, and M. Grangetto, "Panorama view with spatiotemporal occlusion compensation for 3D video coding," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 205–219, Jan. 2015.

[7] Z. Xing, A. Wang, J. Jin, and Y. Wu, "Synthesis-aware region-based 3D video coding," in *Proc. Pacific Rim Conf. Multimedia*, Dec. 2015, pp. 400–409.

[8] C. Lin, Y. Zhao, J. Xiao, and T. Tillo, "Region-based multiple description coding for multiview video plus depth video," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1209–1223, May 2018.

[9] J. Jin, A. Wang, Y. Zhao, C. Lin, and B. Zeng, "Region-aware 3D warping for DIBR," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 953–966, Jun. 2016.

[10] P. Merkle *et al.*, "The effects of multiview depth video compression on multiview rendering," *Signal Process., Image Commun.*, vol. 24, nos. 1–2, pp. 73–88, Jan. 2009.

[11] B. T. Oh and K.-J. Oh, "View synthesis distortion estimation for AVC-and HEVC-compatible 3D video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 6, pp. 1006–1015, Jun. 2014.

[12] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[13] L. Fang, N.-M. Cheung, D. Tian, A. Vetro, H. Sun, and O. C. Au, "An analytical model for synthesis distortion estimation in 3D video," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 185–199, Jan. 2014.

[14] P. Gao, Q. Peng, and W. Xiang, "Analysis of packet-loss-induced distortion in view synthesis prediction-based 3D video coding," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2781–2796, Jun. 2017.

[15] H. Yuan, S. Kwong, X. Wang, Y. Zhang, and F. Li, "A virtual view PSNR estimation method for 3D videos," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 134–140, Mar. 2016.

[16] D. Zhang and J. Liang, "View synthesis distortion estimation with a graphical model and recursive calculation of probability distribution," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 5, pp. 827–840, May 2015.

[17] D. Tian, P. L. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," *Proc. SPIE*, vol. 7443, Sep. 2009, Art. no. 74430T.

[18] J. Jin, J. Liang, Y. Zhao, C. Lin, C. Yao, and A. Wang, "A depth-bin-based graphical model for fast view synthesis distortion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[19] Y. Zhao, C. Zhu, Z. Chen, D. Tian, and L. Yu, "Boundary artifact reduction in view synthesis of 3D video: From perspective of texture-depth alignment," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 510–522, Jun. 2011.

[20] K.-Y. Chen, P.-K. Tsung, P.-C. Lin, H.-J. Yang, and L.-G. Chen, "Hybrid motion/depth-oriented inpainting for virtual view synthesis in multiview applications," in *Proc. 3DTV-Conf., True Vis.-Capture, Transmiss. Display 3D Video*, Jun. 2010, pp. 1–4.

[21] P. Ndjiki-Nya *et al.*, "Depth image-based rendering with advanced texture synthesis for 3D video," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 453–465, Jun. 2011.

[22] C. Yao, Y. Zhao, J. Xiao, H. Bai, and C. Lin, "Depth map driven hole filling algorithm exploiting temporal correlation information," *IEEE Trans. Broadcast.*, vol. 60, no. 2, pp. 394–404, Jun. 2014.

[23] S. Farid, M. Lucenteforte, and M. Grangetto, "Depth image based rendering with inverse mapping," in *Proc. IEEE 15th Int. Workshop Multimedia Signal Process.*, Oct. 2013, pp. 135–140.

[24] D.-H. Li, H.-M. Hang, and Y.-L. Liu, "Virtual view synthesis using backward depth warping algorithm," in *Proc. Picture Coding Symp. (PCS)*, Dec. 2013, pp. 205–208.

[25] R. Hartley and A. Zisserman, *Multiple View Geometry Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[26] J. B. Lu, H. Cai, J.-G. Lou, and J. Li, "An epipolar geometry-based fast disparity estimation algorithm for multiview image and video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 6, pp. 737–750, Jun. 2007.

[27] D. Rusanovskyy, K. Müller, and A. Vetro, *Common Test Conditions of 3DV Core Experiments*, document JCT3VA1100.doc, JCT-3V, Jul. 2012.

[28] (Sep. 2013). *Fraunhofer Heinrich Hertz Institute*. 3DV Sequences of HHI. [Online]. Available: ftp://ftp.hhi.de/HHIMPEG3DV

[29] U. Nagoya. (Mar. 2008). *3DV Sequences of Nagoya University*. [Online]. Available: http://www.tanimoto.nuee.nagoya-u.ac.jp/mpeg/mpeg-ftv.html

[30] M. Domañski *et al.*, *Poznan Multiview Video Test Sequences and Camera Parameters*, document MPEG 2009/M17050, ISO/IEC JTC1/SC29/WG11, Xian, China, Oct. 2009.

[31] D. Rusanovskyy, P. Aflaki, and M. M. Hannuksela, *Undo Dancer 3DV Sequence for Purposes of 3DV Standardization*, document ISO/IEC JTC1/SC29/WG11, Doc. M20028, Geneva, Switzerland, Mar. 2011.

[32] J. Zhang, R. Li, H. Li, D. Rusanovskyy, and M. M. Hannuksela, *Ghost Town Fly 3DV Sequence for Purposes of 3DV Standardization*, document ISO/IEC JTC1/SC29/WG11, Doc. M20027, Geneva, CH, Mar. 2011.

[33] (Mar. 2013). *Joint Collaborative Team for 3DV*. 3D-HTM Software Platform. [Online]. Available: https://hevc.hhi.fraunhofer.de/

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.

Jian Jin received the B.S. and M.E. degrees in economics management and electronic engineering from the Taiyuan University of Science and Technology, Shanxi, China, in 2011 and 2014, respectively. He is currently pursuing the Ph.D. degree with Beijing Jiaotong University, Beijing, China.

He was as a Visiting Ph.D. Student with Simon Fraser University, Burnaby, BC, Canada. His research interests include image/video compression, 3D video synthesis, machine learning, and deep learning. He has served as a Reviewer for the IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP), the IEEE TRANSACTIONS ON MULTIMEDIA (TMM), and the *EURASIP Journal on Image and Video Processing*.

Jie Liang (S'99–M'04–SM'11) received the B.E. and M.E. degrees from Xi'an Jiaotong University, China, in 1992 and 1995, the M.E. degree from the National University of Singapore (NUS) in 1998, and the Ph.D. degree from Johns Hopkins University, Baltimore, MD, USA, in 2003.

Since 2004, he has been with the School of Engineering Science, Simon Fraser University, Canada, where he is currently a Professor and the Associate Director. In 2012, he visited the University of Erlangen-Nuremberg, Germany, as an Alexander von Humboldt Research Fellow. From 2003 to 2004, he was with the Video Codec Group of Microsoft Digital Media Division. From 1997 to 1999, he was with Hewlett-Packard Singapore and the Center for Wireless Communications, NUS. His research interests include image and video coding, multimedia communications, sparse signal processing, computer vision, and machine learning.

Dr. Liang received the 2014 IEEE TCSVT Best Associate Editor Award, the 2014 SFU Dean of Graduate Studies Award for Excellence in Leadership, and the 2015 Canada NSERC Discovery Accelerator Supplements (DAS) Award. He has served as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCSVT), the IEEE SIGNAL PROCESSING LETTERS, *Signal Processing: Image Communication*, and the *EURASIP Journal on Image and Video Processing*.

Yao Zhao received the B.S. degree from the Radio Engineering Department, Fuzhou University, Fuzhou, China, in 1989, and the M.E. degree from the Radio Engineering Department, Southeast University, Nanjing, China, in 1992, and the Ph.D. degree from the Institute of Information Science, Beijing Jiaotong University (BJTU), Beijing, China, in 1996.

He became an Associate Professor with BJTU in 1998 and became a Professor in 2001. From 2001 to 2002, he was a Senior Research Fellow with the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands. In 2015, he visited the Swiss Federal Institute of Technology, Lausanne, Switzerland (EPFL). From 2017 to 2018, he visited the University of Southern California. He is currently the Director of the Institute of Information Science, BJTU. His current research interests include image/video coding, digital watermarking and forensics, video analysis and understanding, and artificial intelligence.

Dr. Zhao serves on the Editorial Boards of several international journals, including as an Associate Editor of the IEEE TRANSACTIONS ON CYBERNETICS, a Senior Associate Editor of the IEEE SIGNAL PROCESSING LETTERS, and an Area Editor of the *Signal Processing: Image Communication*. He was named as a Distinguished Young Scholar by the National Science Foundation of China in 2010, and was elected as a Chang Jiang Scholar of the Ministry of Education of China in 2013. He is a fellow of the IET.

Chunyu Lin was born in Liaoning, China. He received the Ph.D. degree from Beijing Jiaotong University, Beijing, China, in 2011. From 2009 to 2010, he was a Visiting Researcher at the ICT Group, Delft University of Technology, Delft, The Netherlands. From 2011 to 2012, he was a Post-Doctoral Researcher with the Multimedia Laboratory, Gent University, Gent, Belgium. He is currently an Associate Professor with Beijing Jiaotong University. His research interests include image/video compression and robust transmission, 2D-to-3D conversion, 3D video coding, and virtual reality video processing.

Chao Yao received the B.S. degree in computer science from Beijing Jiaotong University (BJTU), Beijing, China, in 2009, the Ph.D. degree from the Institute of Information Science, BJTU, in 2016. From 2014 to 2015, he was a Visiting Ph.D. Student with the Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. Since 2016, he has been a Post-Doctoral Fellow with the Beijing University of Posts and Telecommunications (BUPT), Beijing, China. His current research interests include image and video processing and computer vision.

Lili Meng received the B.E. degree from Shandong University, Jinan, China, in 2005, and the Ph.D. degree from Beijing Jiaotong University, Beijing, China, in 2013, under the supervision of Prof. Y. Zhao. In 2010, she visited the National Kaohsiung University of Applied Sciences, Taiwan. From 2010 to 2011, she was a Visiting Student with Simon Fraser University, Canada. From 2017 to 2018, she was a Visiting Scholar with Simon Fraser University, Canada. Since 2014, she has been with the School of Information Science and Engineering, Shandong Normal University, Jinan.

Authorized licensed use limited to: Univ of Science and Tech Beijing. Downloaded on February 17,2023 at 07:24:29 UTC from IEEE Xplore. Restrictions apply.