A Virtual View PSNR Estimation Method for 3-D Videos

Hui Yuan, *Member, IEEE*, Sam Kwong, *Fellow, IEEE*, Xu Wang, *Student Member, IEEE*, Yun Zhang, *Member, IEEE*, and Fengrong Li

***Abstract*—In three-dimensional videos (3-DVs) with *n*-view texture videos plus *n*-view depth maps, virtual views can be synthesized from neighboring texture videos and the associated depth maps. To evaluate the system performance or guide the rate-distortion-optimization process of 3-DV coding, the distortion/PSNR of the virtual view should be calculated by measuring the quality difference between the virtual view synthesized by compressed 3-DVs with one synthesized by uncompressed 3-DVs, which increases the complexity of a 3-DV system. In order to reduce the complexity of 3-DV system, it is better to estimate virtual view distortions/PSNR directly without rendering virtual views. In this paper, the virtual view synthesis procedure and the distortion propagation from existing views to virtual views are analyzed in detail, and then a virtual view distortion/PSNR estimation method is derived. Experimental results demonstrate that the proposed method could estimate PSNRs of virtual views accurately. The squared correlation coefficient and root of mean squared error between the estimated PSNRs by the proposed method and the actual PSNRs are 0.998 and 2.012 on average for all the tested sequences. Since the proposed method is implemented row-by-row independently, it is also friendly for parallel design. The execute time for each row of pictures with 1024×768 resolution is only 0.079 s, while for pictures with 1920×1088 resolution it is only 0.155 s.**

***Index Terms*—Distortion estimation, 3DV, video coding.**

I. Introduction

With the improvements in high-speed networking, high-capacity storage, and high-quality auto-stereoscopic display technologies, extensive commercial applications of three-dimensional video (3DV) are becoming reality, *e.g.*, the well-known 3D television (3DTV) [1] and free viewpoint television (FTV) [2]. In a typical 3DV system, which includes capture, storage, transmission, and display, a 3D scene should first be represented efficiently by using a small amount of data [3]. Among 3D scene representation technologies [3], representations made by *n*-view texture videos plus *n*-view depth maps have been used extensively. For this kind of scene representation, virtual views should be rendered from the acquired *n*-view videos and their corresponding n-view depth maps by depth image-based rendering (DIBR) [4].

In order to obtain the distortion or quality of the virtual view for high-efficiency 3DV coding (3DVC) and 3DV quality assessment, the distortion or PSNR of the virtual view can be calculated by comparing a virtual view synthesized by compressed 3DVs and the one synthesized by uncompressed 3DVs, which increases the complexity of a 3DV system. A more economical way is to estimate the value of the virtual view's distortion/PSNR directly. In [5], *Zhang et al.* proposed a regional based virtual view distortion estimation method for depth maps coding. In [6], a linear model based virtual view distortion estimation method was proposed for depth maps coding. In our previous work [7], a planar model based virtual view distortion estimation method was proposed for joint bit allocation between texture videos and depth maps. Besides, similar distortion models and applications can also be found in [8]–[10]. The existing methods [5]–[10] can estimate the distortion variation tendency to some extent, but the estimated virtual view distortion may not be close to the actual distortion. In order to estimate the virtual view distortion accurately, a synthesis distortion estimation method is proposed in [11]. In this method, the effect of depth map distortion on synthesis distortion is broken down into 2 parts, spatial variant region and spatial invariant region, based on frequency domain analysis. However, the model cannot be used easily due to its high computational complexity.

From the basis of DIBR technology, it can be concluded that the distortion of the virtual view depends only on the distortion of the left and the right texture views and depth maps when the camera systems are calibrated well [7]. Since the DIBR technology is mathematically analytical, the distortion of virtual view can also be mathematical derived from the distortion of the left and the right texture views and depth maps. Motivated by this point, a fast and accurate virtual view distortion/PSNR estimation method is proposed by detailed analyzing of the virtual view synthesis procedure.

To estimate the distortion/PSNR for virtual view accurately with low complexity, DIBR procedure and distortion propagation from existing views to virtual views are analyzed in detail in Section II. During the analysis, it is necessary to mention that the DIBR procedure is equal to disparity compensation when all the cameras are well calibrated [7], thus a depth coding error can only affect the horizontal position of the projected pixels in the virtual view. In addition, for clear representation, a summary of some frequently used notations are given in Table I. Experimental results and conclusions are given in Section III and IV respectively.

Manuscript received June 18, 2015; revised September 30, 2015; accepted October 13, 2015. This work was supported in part by the National Natural Science Foundation of China under Grants 61571274, 61201211, 61471348, and 61501299; in part by the Young Scholars Program of Shandong University (YSPSDU) under Grant 2015WLJH39 in part by the Ph.D. Programs Foundation, Ministry of Education of China under Grant 20120131120032; in part by the Key laboratory of wireless sensor network and communication, Chinese Academy of Sciences under Grant 2013002; in part by the Shenzhen Emerging Industries of Strategic Basic Research Project under Grant JCYJ20150525092941043, in part by the City University of Hong Kong Applied Research Grant 9667094; and in part by the City University of Hong Kong Shenzhen Research Institute, Shenzhen, China.

H. Yuan is with the School of Information Science and Engineering, Shandong University, Jinan 250100, China (e-mail: yuanhui0325@gmail.com).

S. Kwong is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the City University of Hong Kong Shenzhen Research Institute, Shenzhen 5180057, China (e-mail: cssamk@cityu.edu.hk).

X. Wang is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: wangxu@szu.edu.cn).

Y. Zhang is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: yun.zhang@siat.ac.cn).

F. Li is with the Key Laboratory of Wireless Sensor Network and Communication, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, China (e-mail: lifengrongsim@mail.sim.ac.cn).

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

Digital Object Identifier 10.1109/TBC.2015.2492461

TABLE I
NOTATIONS

Notations	Meaning
$B_{aseline}$	Baseline width (horizontal interval between 2 view points)
F_{ocal}	Focal length of cameras
Z_{near}, Z_{far}	Nearest and farthest object distance of the scene
α, β	Parameters depends on l, f, Z_{near} and Z_{far}
T_l, T_r	Original texture image of the left and the right view
T_l', T_r'	Reconstructed texture image of the left and the right view
I_l, I_r	Original depth map of left and right view
I_l', I_r'	Reconstructed depth map of left and right view
$d_{isp,l}$, $d_{isp,r}$	Original disparity (corresponds to depth map I_l and I_r)
$d_{isp,l}'$, $d_{isp,r}'$	Reconstructed disparity (corresponds to depth map I_l' and I_r)
V_l, V_r	Synthesized virtual view image from the left and the right view
V_m	Final output virtual view by blending V_l and V_r
x, y	Pixel positions
x_l, x_r	Pixel positions in the left and the right view
y_l, y_r	Pixel positions in the left and the right view
x_{vl}, x_{vr}	Pixel positions in V_l and V_r
$e_{d,l}$, $e_{d,r}$	Refer to depth coding errors (stochastic variable)
$e_{disp,l}$, $e_{disp,r}$	Refer to disparity errors (stochastic variable) correspond to $e_{d,l}$ and $e_{d,r}$
$e_{d,l}(x_l)$	The coding error of the pixel position x_l in the left view depth map

II. Proposed Virtual View Distortion/PSNR Estimation Method

In a 3DV system, depth maps are used for rendering virtual views. Depth map compression will affect the quality of virtual views indirectly. For a set of well-calibrated 3D videos captured by cameras with parallel arrangements, DIBR can be represented as a disparity compensation procedure [12] in which the relationship between a depth value and the corresponding disparity value can be written as (1) [13],

$$I = 255 \cdot \left(\frac{d_{isp}}{F_{ocal} \cdot B_{aseline}} - \frac{1}{Z_{far}} \right) \Big/ \left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}} \right), \tag{1}$$

where I denotes the pixel value of the depth map, d_{isp} denotes the corresponding disparity value, F_{ocal} is the focal length, $B_{aseline}$ is the baseline width (horizontal interval between 2 view points), and Z_{near} and Z_{far} are the nearest and farthest object distances from the scene. Equation (1) could be rewritten as a simple form,

$$I = \alpha \cdot d_{isp} + \beta, \tag{2}$$

where α and β depend on camera parameters, i.e., F_{ocal}, $B_{aseline}$, Z_{near}, and Z_{far}. When depth maps are compressed, the reconstructed pixel value I' could be represented as,

$$I' = \alpha \cdot d_{isp}' + \beta, \tag{3}$$

where d_{isp}' is the corresponding disparity value of I'. Therefore, the relationship between the depth error e_d and the disparity error can be written as,

$$e_d = \alpha \cdot \left(d_{isp} - d_{isp}' \right) = \alpha \cdot e_{disp}, \tag{4}$$

which means that the disparity error e_{disp} varies linearly with the depth error e_d.

In a typical virtual view synthesis procedure, as shown in Fig. 1, the existing texture videos (left view and right view) and the associated depth maps (left depth map and right depth map) are first warped to the virtual view position to generate 2 virtual views (generated from the left view and right view, respectively) and 2 virtual depth maps (generated from the left depth map and the right depth map).

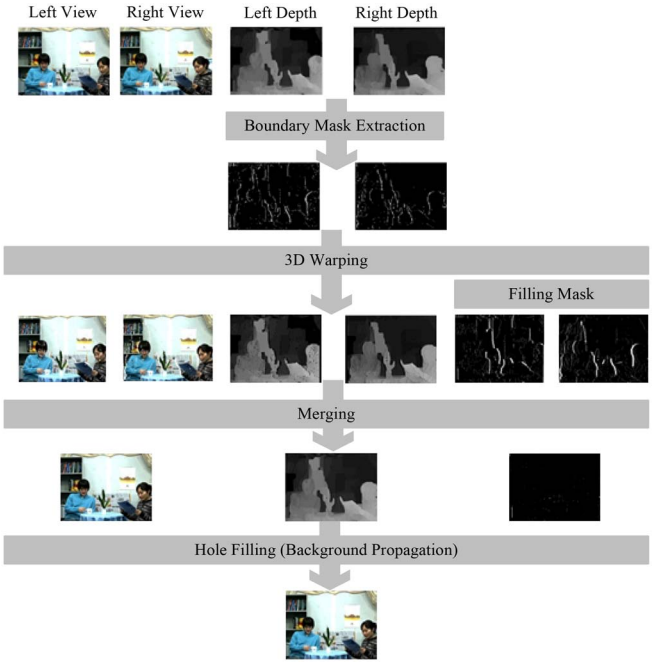

Fig. 1. Flow diagram for VSRS 1D mode [14].

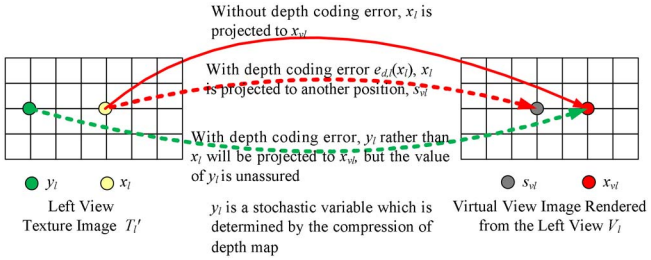

Fig. 2. Pixel projection with and without depth coding error.

At the same time, 2 boundary masks are also generated from the left and right depth maps. Then, the 2 virtual views and virtual depth maps are merged to generate a virtual view and a virtual depth map, while the 2 boundary masks are merged to generate a hole mask. Finally, the merged virtual view and depth map are modified by the hole mask to generate the final virtual view and virtual depth map. From the virtual view synthesis procedure, we can observe that the occlusion and dis-occlusion areas are small and can be filled well. Therefore, in the research, we do not treat the occlusion and dis-occlusion areas specially.

From Fig. 1, the coding error of depth maps will first affect the quality of the 2 virtual views (denoted as V_l and V_r, respectively) generated from the left and right views, and then affect the quality of the final merged virtual view (denoted as V_m). Let $e_{d,l}(x)$ and $e_{d,r}(x)$ represent the coding error of pixel x in the left and the right depth maps, respectively; $I_l(x)$ and $I_r(x)$ represent the original depth pixel values of x; and $I_l'(x)$ and $I_r'(x)$ represent the reconstructed depth pixel values of x. Accordingly, the corresponding disparity of $I_l(x)$, $I_r(x)$, $I_l'(x)$, and $I_r'(x)$ could be represented as $d_{isp,l}(x)$, $d_{isp,r}(x)$, $d_{isp,l}'(x)$, and $d_{isp,r}'(x)$, respectively.

When the left view depth map is not compressed, as shown in the solid red line of Fig. 2, the pixel position x_l in the compressed left view (T_l') will be warped to pixel position x_{vl} in V_l, *i.e.*,

$$x_l\text{-}d_{isp,l}(x_l) = x_{vl}. \tag{5}$$

When the left view depth map is compressed, as shown in the dotted-red line of Fig. 2, due to the assured $e_d(x_l)$, we only confirm that x_l may be projected to another pixel position rather than x_{vl}, while another pixel position y_l will be projected to x_{vl}, as shown by the dotted- green line of Fig. 2, *i.e.*,

$$y_l\text{-}d'_{isp,l}(y_l) = x_{vl}. \tag{6}$$

However, y_l is not assured, and we only confirm that y_l depends on the compression of the left view depth map. Therefore, y_l is considered a **stochastic variable** for a certain pixel position x_{vl} in V_l. Subsequently, the squared error (SE_l) of the $x_{vl}{}^{th}$ pixel in a row of the virtual view can be written as,

$$SE(x_{vl}) = \left[T'_l(x_l) - T'_l(y_l)\right]^2, \tag{7}$$

in which the relationship between x_l and y_l can be written as,

$$x_l\text{-}d_{isp,l}(x_l) = y_l\text{-}d'_{isp,l}(y_l). \tag{8}$$

Taking (2) and (3) into consideration, (8) can be rewritten as,

$$\begin{aligned} y_l &= x_l - d_{isp,l}(x_l) + d'_{isp,l}(y_l) \\ &= x_l - \left[I_l(x_l) - \beta\right]/\alpha + \left[I'_l(y_l) - \beta\right]/\alpha \\ &= x_l - \left[I_l(x_l) - I'_l(y_l)\right]/\alpha \\ &= x_l - \left[I_l(x_l) - I'_l(x_l) + I'_l(x_l) - I'_l(y_l)\right]/\alpha \\ &= x_l - \left[e_{d,l}(x_l) + I'_l(x_l) - I'_l(y_l)\right]/\alpha \\ &= x_l - \Delta_l(x_l) - \tau_l(x_l, y_l), \end{aligned} \tag{9}$$

where

$$\Delta_l(x_l) = e_{d,l}(x_l)/\alpha, \tag{10}$$

$$\tau_l(x_l, y_l) = \left[I'_l(x_l) - I'_l(y_l)\right]/\alpha. \tag{11}$$

Accordingly, $SE(x_{vl})$ can be represented as,

$$\begin{aligned} SE(x_{vl}) &= \left[T'_l(x_l) - T'_l(y_l)\right]^2 \\ &= \left[T'_l(x_l) - T'_l(x_l - \Delta_l(x_l) - \tau_l(x_l, y_l))\right]^2. \end{aligned} \tag{12}$$

It can be observed from (12) that the $SE(x_{vl})$ can be calculated when $\Delta_l(x_l)$ and $\tau_l(x_l, y_l)$ are confirmed. Because $\Delta_l(x_l)$ can be obtained from the assured $e_{d,l}(x_l)$ and α directly, the main problem is how to obtain $\tau_l(x_l, y_l)$. From (9), we can observe that $\tau_l(x_l, y_l)$ depends on $I'_l(x_l)$-$I'_l(y_l)$, in which $I'_l(x_l)$ and $I'_l(y_l)$ represent the pixel values of positions x_l and y_l in the reconstructed left view depth map. Therefore, $\tau_l(x_l, y_l)$ is in turn determined by y_l. Since y_l can be considered a stochastic variable, $\tau_l(x_l, y_l)$ can also be considered a stochastic variable. Thus, in order to calculate (12), the expectation of $\tau_l(x_l, y_l)$ can be used to replace itself.

In order to calculate the expectation of $\tau_l(x_l, y_l)$, the probability density function of $\tau_l(x_l, y_l)$ must be confirmed first. Because y_l and $\tau_l(x_l, y_l)$ are affected directly by the position (disparity) error $\boldsymbol{e_{disp,l}}$, which is determined by the coding error of the depth map as shown in (4), the probability density function of y_l and $\tau_l(x_l, y_l)$ is the same as that of $\boldsymbol{e_{disp,l}}$, which is highly correlated with $\boldsymbol{e_{d,l}}$. Note that $\boldsymbol{e_{disp,l}}$ and $\boldsymbol{e_{d,l}}$ refer to **stochastic variables** whose probability density function could be obtained by statistic histogram of all the depth coding errors and the corresponding disparity errors in a row of the left view. Therefore, for a certain x_l, the expectation of $\tau_l(x_l, y_l)$ can be calculated by (13),

$$\begin{aligned} E\{\tau_l(x_l, y_l)\} &= \sum_{\tau_l(x_l,y_l)} f(\tau_l(x_l, y_l))\tau_l(x_l, y_l) \\ &= \sum_{y_l=-b_l}^{b_l} f(y_l)\tau_l(x_l, y_l) \\ &= \sum_{y_l=-b_l}^{b_l} f\big(e_{disp,l}\big)\tau_l(x_l, y_l), \end{aligned} \tag{13}$$

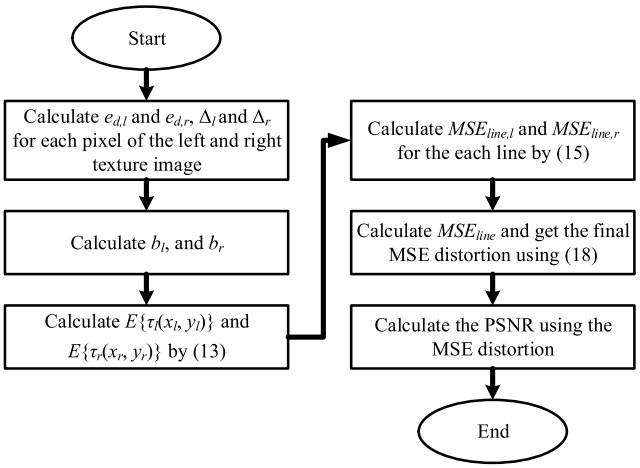

Fig. 3. Proposed virtual view distortion estimation procedure.

where $f(\tau_l(x_l, y_l))$ and $f(y_l)$ are the probability density functions of $\tau_l(x_l, y_l)$ and y_l, $f(e_{disp,l})$ is the probability density function of $e_{disp,l}$, and b_l is the maximum distance between x_l and y_l, and can be calculated by (14),

$$b_l = \max\{\Delta_l(0), \Delta_l(1) \cdots \Delta_l(W-1)\}, \tag{14}$$

where $\Delta_l(0)$, $\Delta_l(1)$, …, and $\Delta_l(W\text{-}1)$ are the disparity errors of each pixel position in a row, and W is the width of the picture.

As a result, the mean squared error of a row in V_l can be written as,

$$\begin{aligned} &MSE_{row,l} \\ &= \frac{1}{W}\sum_{x_{vl}=0}^{W-1} SE(x_{vl}) \\ &= \frac{1}{W}\sum_{x_l=0}^{W-1}\left[T'_l(x_l) - T'_l(x_l - \Delta_l(x_l) - \tau_l(x_l, y_l))\right]^2 \\ &\approx \frac{1}{W}\sum_{x_l=0}^{W-1}\left[T'_l(x_l) - T'_l(x_l - \Delta_l(x_l) - E\{\tau_l(x_l, y_l)\})\right]^2. \end{aligned} \tag{15}$$

Subsequently, by using the same method, $MSE_{row,r}$ can be calculated as,

$$\begin{aligned} &MSE_{row,r} \\ &\approx \frac{1}{W}\sum_{x_r=0}^{W-1}\left[T'_r(x_r) - T'_r(x_r - \Delta_r(x_r) - E\{\tau_r(x_r, y_r)\})\right]^2. \end{aligned} \tag{16}$$

Then, for each row of the merged virtual view, the mean squared error (denoted as MSE_{row}) can be written as,

$$MSE_{row} = \omega_l MSE_{row,l} + \omega_r MSE_{row,r}, \tag{17}$$

where ω_l and ω_r are weighted coefficients (defined as 0.5 for both ω_l and ω_r) for the left and the right view respectively. Finally, for the whole virtual view image, the mean squared error (MSE) can be written as,

$$MSE = \sum_{row=0}^{H-1} MSE_{row}, \tag{18}$$

where H is the height of the virtual view image.

Based on the above analysis, the virtual view distortion can be estimated by the following procedure, as shown in Fig. 3.

Step 1: Calculate the coding error, i.e., $e_{d,l}$ and $e_{d,r}$, as well as the corresponding Δ_l and Δ_r for all the pixels in the left and the

TABLE II
IMPORTANT ENCODER PARAMETERS

Encoder Parameters	Values	Encoder Parameters	Values
Unit Definition		**Motion Search**	
MaxCUWidth	64	FastSearch	1
MaxCUHeight	64	SearchRange	64
MaxPartitionDepth	4	BipredSearchRange	1
QuadtreeTULog2MaxSize	5	HadamardME	1
QuadtreeTULog2MinSize	3	FEN	1
QuadtreeTUMaxDepthInter	3	FDM	1
QuadtreeTUMaxDepthIntra	3		
Coding Structure		**Other Parameters**	
IntraPeriod	24	WaveFrontSynchro	1
DecodingRefreshType	1	UniformSpacingIdc	0
GOPSize	8	PCMEnabledFlag	0
		SliceMode	0
Quantization		**Deblock Filter**	
MaxDeltaQP	0	DeblockingFilterControlPresent	1
MaxCuDQPDepth	0	LoopFilterOffsetInPPS	0
DeltaQpRD	0	LoopFilterDisable	0 1
RDOQ	1	LoopFilterBetaOffset_div2	0
RDOQTS	1	LoopFilterTcOffset_div2	0
		DeblockingFilterMetric	0
Coding Tools		**Multiview Coding Tools**	
SAO	1 0	IvMvPred	1 1
AMP	1	AdvMultiviewResPred	1
TransformSkip	1	IlluCompEnable	1
TransformSkipFast	1	ViewSynthesisPred	1
SAOLcuBoundary	0	DepthRefinement	1
RateControl	0	IvMvScaling	1
		SubPULog2Size	3
Depth CodingTools		**View Synthesis Optimization (VSO)**	
VSO	1	WVSO	1
DMM	1	VSOWeight	10
SDC	1	VSDWeight	1
DLT	1	DWeight	1
QTL	1	UseEstimatedVSD	1
PC	1		
InterSDC	1	NA	
MPI	1		

right view depth maps, where Δ_r represents the disparity error of each pixel position of the right view;

Step 2: Calculate b_l and b_r based on the maximum value of Δ_l and Δ_r;

Step 3: Calculate $E\{\tau_l(x_l, y_l)\}$ by using (13);

Step 4: Calculate $E\{\tau_r(x_r, y_r)\}$ by the same way of Step 3.

Step 5: Calculate $MSE_{row,l}$ and $MSE_{row,r}$ for the each row based on (15) and (16).

Step 6: Calculate MSE_{row} based on (17) for each row, and get the final MSE of virtual view based on (18).

III. EXPERIMENTAL RESULTS

In order to verify the accuracy of the proposed PSNR estimation method, 8 3DV sequences [15] that are adopted by Joint Collaborative Team on 3D Video Coding (JCT-3V), i.e., *BookArrival* (view 10 and 8 are coded, view 9 is set as virtual view), *Newspapercc* (view 4 and 6 are coded, view 5 is set as virtual view), *Kendo* (view 1 and 3 are coded, view 2 is set as virtual view), *Balloons* (view 1 and 3 are coded, view 2 is set as virtual view), *GhostTwonFly* (view 5 and 9 are coded, view 7 is set as virtual view), *UndoDancer* (view 1 and 5 are coded, view 3 is set as virtual view), *PoznanStreet* (view 4 and 5 are coded, view 4.5 is set as virtual view), and *PoznanHall2* (view 6 and 7 are coded, view 6.5 is set as virtual view), with different content and camera parameters were used.

The sequences are encoded with texture/depth *QP* pairs of (15, 24), (20, 29), (25, 34), (30, 39), (35, 42), and (40, 45) by 3D video coding test platform version 9.2 (3D-HTM9.2) [16] with configurations defined by JCT-3V common test condition [17], as shown in Table II.

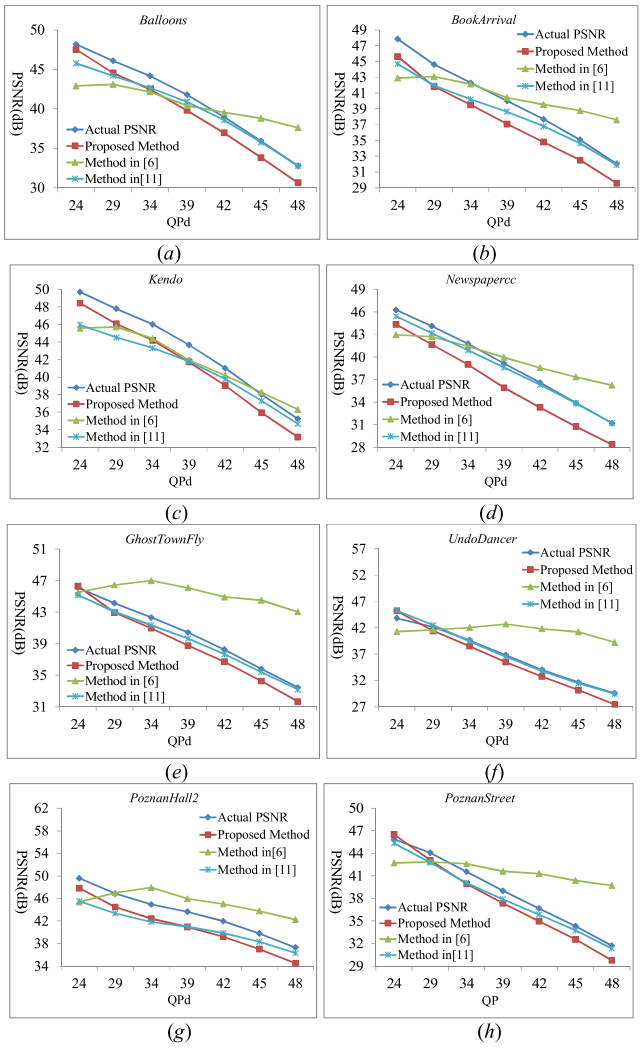

Fig. 4. Actual PSNRs and estimated PSNRs comparison, the 1st frame with different QPs.

The View Synthesis Reference Software [16] adopted by JCT-3V was used for rendering virtual views. The proposed distortion estimation method was implemented on Matlab Version 7.12.0 without any optimization and was executed on an Intel Core T5500 CPU with 2.0GHz basic frequency.

The estimated PSNRs of the proposed method, the method in [6], and the state-of-the-art method in [11], and the actual PSNRs of synthesized virtual views are compared in Fig. 4 and Table III.

Table IV shows the squared correlation coefficients (SCC) and the root mean squared error (RMSE) of the actual PSNRs and the estimated PSNRs calculated by different methods. From Table IV, it can be observed that the SCC and RMSE between the estimated PSNRs by the proposed method and the actual PSNRs are 0.998 and 2.012 on average, respectively, while those between the estimated PSNRs by the method in [6] and the actual PSNRs are 0.896 and 3.879, and those between the estimated PSNR by the state-of-art method in [11] and the actual PSNRs are 0.999 and 1.406. Therefore, it can be concluded that the proposed method is more accurate the method in [6] and is close to the state-of-the-art method in [11].

Fig. 5 shows the comparison of the estimated PSNRs for different frames with a same pair (QP_t=25, QP_d=34) for all the tested sequences. It can be observed that the proposed method could estimate the PSNR trends (along with frames) accurately. The estimated PSNRs are close to the actual PSNRs, and the accuracy of

TABLE III
COMPARISONS AMONG DIFFERENT METHODS

Sequences			Actual		Proposed Method			Method in [6]			Method in Method [11]		
	QPt	QPd	PSNR	MSE	PSNR	MSE	Time(s)	PSNR	MSE	Time(s	PSNR	MSE	Time(s)
Balloons	15	24	48.175	0.990	47.508	1.154	55.182	44.485	2.315	1.445	45.768	1.723	110.792
	20	29	46.081	1.603	44.551	2.280	60.667	44.587	2.261	1.283	44.174	2.487	109.228
	25	34	44.179	2.484	42.484	3.671	60.417	43.719	2.762	1.342	42.667	3.519	82.571
	30	39	41.780	4.316	39.796	6.815	59.752	41.227	4.902	1.648	40.886	5.302	85.773
	35	42	38.970	8.242	36.948	13.130	56.730	39.850	6.732	1.427	38.550	9.079	83.771
	40	45	35.919	16.643	33.827	26.941	57.231	38.132	9.996	1.365	35.764	17.246	85.081
	45	48	32.781	34.274	30.634	56.197	64.764	36.640	14.095	1.459	32.768	34.378	85.548
BookArrival	15	24	47.866	1.063	45.648	1.771	60.376	42.908	3.328	1.313	44.710	2.198	83.405
	20	29	44.628	2.240	41.817	4.280	56.654	43.095	3.189	1.347	41.968	4.133	83.899
	25	34	42.304	3.826	39.502	7.293	56.944	42.146	3.967	1.344	40.217	6.186	84.997
	30	39	40.017	6.477	37.074	12.754	56.719	40.418	5.905	1.326	38.640	8.894	84.568
	35	42	37.678	11.099	34.784	21.609	60.902	39.541	7.228	1.302	36.809	13.557	85.635
	40	45	35.079	20.193	32.492	36.634	58.618	38.791	8.589	1.321	34.626	22.411	85.664
	45	48	32.031	40.732	29.577	71.681	59.995	37.607	11.282	1.320	31.893	42.053	86.157
Kendo	15	24	49.704	0.696	48.433	0.933	65.846	45.578	1.800	1.343	45.966	1.646	89.005
	20	29	47.777	1.085	46.070	1.607	63.682	45.713	1.745	1.369	44.525	2.294	82.498
	25	34	46.014	1.628	44.184	2.481	61.898	44.403	2.359	1.314	43.310	3.035	83.014
	30	39	43.676	2.789	41.730	4.366	62.165	41.948	4.152	1.274	41.799	4.297	83.143
	35	42	41.018	5.144	39.036	8.118	56.042	40.197	6.215	1.400	39.776	6.846	84.838
	40	45	38.024	10.248	35.935	16.579	58.675	38.264	9.699	1.280	37.301	12.104	84.752
	45	48	35.231	19.498	33.154	31.457	57.375	36.315	15.192	1.370	34.665	22.214	84.505
Newspapercc	15	24	46.259	1.539	44.350	2.388	67.260	42.958	3.290	1.538	45.420	1.867	84.822
	20	29	44.098	2.531	41.649	4.448	60.143	42.688	3.502	1.401	43.164	3.138	85.956
	25	34	41.771	4.325	38.992	8.201	63.161	41.421	4.688	1.374	40.891	5.297	83.367
	30	39	39.141	7.925	35.921	16.635	68.857	39.949	6.579	1.364	38.627	8.920	84.898
	35	42	36.600	14.227	33.302	30.399	59.482	38.576	9.025	1.473	36.296	15.258	87.032
	40	45	33.870	26.674	30.746	54.758	66.996	37.336	12.010	1.401	33.852	26.785	84.665
	45	48	31.187	49.473	28.373	94.579	65.346	36.252	15.414	1.462	31.219	49.111	84.402
GhostTownFly	15	24	46.114	1.591	46.317	1.519	175.636	45.511	1.828	2.634	45.175	1.975	225.690
	20	29	44.148	2.502	42.947	3.299	168.210	46.446	1.474	2.625	43.049	3.222	219.648
	25	34	42.331	3.802	40.957	5.217	157.782	47.007	1.295	2.765	41.355	4.759	229.282
	30	39	40.435	5.883	38.769	8.634	170.680	46.080	1.603	2.666	39.649	7.050	221.161
	35	42	38.247	9.735	36.702	13.894	175.140	44.925	2.092	2.796	37.675	11.107	220.872
	40	45	35.790	17.145	34.265	24.354	157.906	44.514	2.300	2.646	35.412	18.704	224.723
	45	48	33.457	29.335	31.657	44.404	157.204	43.036	3.232	2.847	33.192	31.183	221.155
UndoDancer	15	24	43.794	2.714	45.208	1.960	152.120	41.249	4.878	2.705	45.231	1.950	219.770
	20	29	42.133	3.979	41.401	4.710	164.235	41.606	4.493	2.797	42.510	3.648	218.688
	25	34	39.613	7.108	38.497	9.192	179.730	42.037	4.068	2.655	39.446	7.388	223.098
	30	39	36.805	13.569	35.481	18.409	165.684	42.694	3.497	2.717	36.502	14.551	231.174
	35	42	34.010	25.830	32.736	34.635	158.699	41.804	4.292	2.610	33.758	27.373	200.141
	40	45	31.618	44.804	30.166	62.581	161.013	41.214	4.917	2.610	31.431	46.769	200.424
	45	48	29.524	72.552	27.431	117.485	164.775	39.217	7.787	2.664	29.424	74.245	198.994
PoznanHall2	15	24	49.595	0.714	47.835	1.070	191.541	45.395	1.877	3.007	45.522	1.824	203.141
	20	29	46.876	1.335	44.500	2.307	184.197	46.998	1.298	2.624	43.360	3.000	200.886
	25	34	44.958	2.076	42.425	3.720	177.513	47.952	1.042	2.584	41.827	4.269	200.827
	30	39	43.671	2.792	40.939	5.238	169.815	45.964	1.647	2.771	41.008	5.155	200.330
	35	42	41.975	4.127	39.236	7.753	175.529	45.022	2.046	2.640	39.876	6.691	198.733
	40	45	39.817	6.782	37.028	12.891	163.642	43.765	2.732	2.814	38.333	9.545	202.633
	45	48	37.299	12.112	34.513	23.005	172.433	42.261	3.864	2.780	36.333	15.128	202.833
PoznanStreet	15	24	45.881	1.679	46.518	1.450	196.398	42.708	3.485	2.721	45.346	1.899	215.575
	20	29	44.075	2.544	43.075	3.203	151.296	42.864	3.363	2.402	42.788	3.422	202.921
	25	34	41.557	4.543	39.925	6.616	170.817	42.606	3.569	2.790	40.072	6.396	209.518
	30	39	39.028	8.134	37.365	11.928	170.170	41.613	4.485	2.757	37.902	10.542	208.817
	35	42	36.672	13.991	34.975	20.683	159.152	41.283	4.839	2.379	35.863	16.857	206.276
	40	45	34.328	24.004	32.550	36.152	162.717	40.378	5.960	2.561	33.759	27.367	205.769
	45	48	31.712	43.838	29.763	68.678	160.528	39.720	6.935	2.542	31.354	47.605	206.936

the proposed method is comparable to that of the start-of-the-art method in [11].

Moreover, Table V compares the complexity of the proposed method with the method in [6], and the state-of-the-art method [11]. From Table V, we can observe that the average execute time of the proposed method for pictures with 1024×768 resolution is 60.781s, while that for pictures with 1920×1088 resolution is 168.377s. It should be noted that the proposed method is friendly for parallel design, i.e., each row can be processed independently. When using parallel processing (e.g., one row corresponds to one core

TABLE IV
SCC AND RMSE BETWEEN THE ACTUAL PSNRS AND THE ESTIMATED PSNRS

Sequences	Proposed Method		Method in [6]		Method in [11]	
	RMSE	SCC	RMSE	SCC	RMSE	SCC
Balloons	1.800	1.000	2.297	0.947	1.348	1.000
BookArrival	2.684	0.999	3.283	0.908	1.862	0.999
Kendo	1.862	1.000	2.027	0.983	2.323	1.000
Newspapercc	2.836	0.997	2.809	0.939	0.622	1.000
GhostTownFly	1.419	0.999	6.232	0.892	0.774	1.000
UndoDancer	1.396	0.993	6.482	0.672	0.590	0.999
PoznanHall2	2.554	1.000	3.409	0.876	2.762	0.993
PoznanStreet	1.544	1.000	4.493	0.946	0.965	0.998
Average	**2.012**	**0.998**	**3.879**	**0.896**	**1.406**	**0.999**

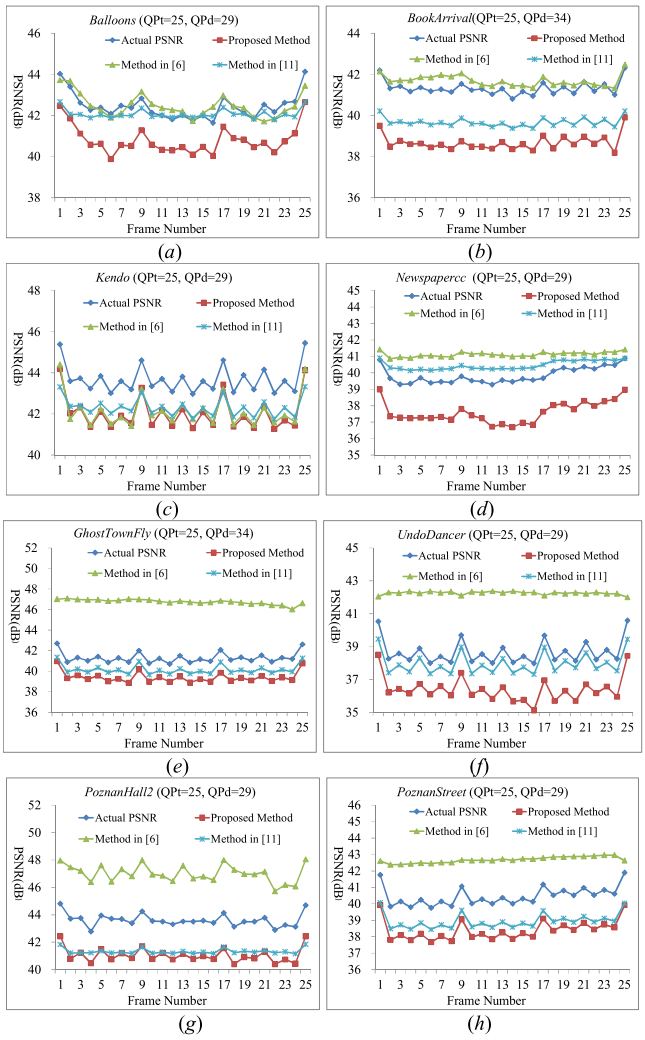

Fig. 5. Actual PSNRs and estimated PSNRs comparison, successive 25 frames with the same QP pair (QP_t=25,QP_d=34).

of a Graphic Process Unit), the execute time for each row of pictures with 1024×768 resolution is only 60.7813/768=0.079s, while for pictures with 1920×1088 resolution it is only 168.3772/1088=0.155s. Although the complexities of the method in [6] is smaller than those of the proposed method, the estimated PSNRs of those methods are not accurate enough. Compared with the state-of-the-art method [11], the proposed method could give similar estimation accuracy with much lower complexity and is more suitable for parallel design.

TABLE V
EXECUTE TIME COMPARISONS

Sequences	Proposed Method	Method in [6]	Method in [11]
	Time(s)	Time(s)	Time(s)
Balloons	59.249	1.424	91.824
BookArrival	58.601	1.325	84.903
Kendo	60.812	1.336	84.536
Newspapercc	64.463	1.430	85.020
Average	**60.781**	**1.379**	**86.571**
GhostTownFly	166.080	2.711	223.219
UndoDancer	163.751	2.679	213.184
PoznanHall2	176.381	2.746	201.341
PoznanStreet	167.297	2.593	207.973
Average	**168.377**	**2.682**	**211.429**

Although the execute time may be reduced by using a simplified method to calculate the power spectrum density in [11] of the texture picture, some complex operations, i.e., boundary detection, Fourier Transform, etc., are still needed; while in the proposed method, the distortions/PSNRs could be calculated row by row directly in the spatial domain that is parallel friendly.

IV. CONCLUSION

In this paper, a fast and parallel-design-friendly virtual view distortion/quality (evaluated by PSNR) estimation method is proposed based on detailed analysis of virtual view synthesis procedure. Experimental results show that the proposed method can estimate the PSNRs of virtual views accurately. The SCC and RMSE between the PSNRs estimated by the proposed method and the actual PSNRs are 0.998 and 2.012 on average. In addition, the execution time for each row of pictures with 1024×768 resolution is only 0.079s, while for pictures with 1920×1088 resolution it is only 0.155s.

Since the main objective of depth coding is to compress depth maps efficiently while guaranteeing the quality of synthesized virtual views, the future work is to design novel rate distortion optimization algorithms, efficient bit allocation, no-reference virtual view quality assessment methods, etc., for 3DV application systems by employing the model. In addition, in the proposed method, the estimated distortions are calculated based on the probability distribution of depth coding errors; therefore, for block-based depth map coding, when the pixel numbers are small (e.g., 16×16, 8×8, 4×4), the probability distribution may not be accurate enough. In the future, we will design a more flexible distortion estimation method for block-based depth map coding.

ACKNOWLEDGMENT

The authors would like to thank the editors and anonymous reviewers for their valuable comments. They would also like to thank JCT-3V Group for providing their 3D video sequences and their valuable work on 3DV.

REFERENCES

[1] L. Onural, A. Gotchev, H. M. Ozaktas, and E. Stoykova, "A survey of signal processing problems and tools in holographic three-dimensional television," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1631–1646, Nov. 2007.

[2] M. Tanimoto and T. Fujii, "FTV—Free viewpoint television," in *Proc. 61st Meeting ISO/IEC JTC1/SC29/WG11, Doc. M8595*, Flagenfurt, Austria, Jul. 2002.

[3] A. Alatan *et al.*, "Scene representation technologies for 3DTV—A survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1587–1605, Nov. 2007.

[4] C. Fehn, "Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV," in *Proc. SPIE Stereoscopic Image Process. Render.*, vol. 5291. San Jose, CA, USA, Jan. 2004, pp. 93–104.

[5] Y. Zhang *et al.*, "Regional bit allocation and rate distortion optimization for multiview depth video coding with view synthesis distortion model," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3497–3512, Jun. 2013.

[6] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map distortion analysis for view rendering and depth coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Cairo, Egypt, Nov. 2009, pp. 721–724.

[7] H. Yuan, Y. Chang, J. Huo, F. Yang, and Z. Lu, "Model based joint bit allocation between texture videos and depth maps for 3D video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 4, pp. 485–497, Apr. 2011.

[8] J. Xiao, T. Tillo, and H. Yuan, "Macroblock level bits allocation for depth maps in 3D video coding," *J. Signal Process. Syst.*, vol. 74, no. 1, pp. 127–135, Jan. 2014.

[9] S. Hu, S. Kwong, Y. Zhang, and C.-C. J. Kuo, "Rate-distortion optimized rate control for depth map-based 3D video coding," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 585–594, Feb. 2013.

[10] F. Shao, G. Jiang, W. Lin, M. Yu, and Q. Dai, "Joint bit allocation and rate control for coding multi-view video plus depth based 3D video," *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 1843–1854, Dec. 2013.

[11] L. Fang *et al.*, "An analytical model for synthesis distortion estimation in 3D video," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 185–199, Jan. 2014.

[12] P. Lai, A. Ortega, C. C. Dorea, P. Yin, and C. Gomila, "Improving view rendering quality and coding efficiency by suppressing compression artifacts in depth-image coding," in *Proc. Visual Commun. Image Process. (VCIP)*, vol. 7257. San Jose, CA, USA, Jan. 2009, pp. 72570O.1–72570O.10.

[13] H. Yuan, J. Liu, H. Xu, Z. Li, and W. Liu, "Coding distortion elimination of virtual view synthesis for 3D video system: Theoretical analyses and implementation," *IEEE Trans. Broadcast.*, vol. 58, no. 4, pp. 558–568, Dec. 2012.

[14] L. Zhang, K. Wegner, and S. Yea, "3D-HEVC test model 5," in *Proc. 5th Meeting Joint Collaborative Team 3D Video Coding Ext. Develop. ITU-T SG16 WP3 ISO/IEC JTC1/SC29/WG11, Doc. JCT3V-E1005*, Vienna, Austria, Aug. 2013.

[15] (Sep. 2013). *Joint Collaborative Team for 3DV, 3D Video Test Sequences*. [Online]. Available: ftp://ftp.hhi.fraunhofer.de

[16] (Mar. 2013). *Joint Collaborative Team for 3DV, 3D-HTM Software Platform*. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/

[17] D. Rusanovskyy, K. Müller, and A. Vetro, "Common test conditions of 3DV core experiments," in *Proc. 3rd Meeting ITU-T/ISO/IEC Joint Collaborative Team 3D Video Coding (JCT-3V), Doc. JCT3V-C1100*, Jan. 2013.