RATE-COMPLEXITY TRADEOFF FOR CLIENT-SIDE FREE VIEWPOINT IMAGE RENDERING

Yu Gao [*], *Gene Cheung* [#], *Jie Liang* [*]

[*] Simon Fraser University, Canada, [#] National Institute of Informatics, Japan

ABSTRACT

Free viewpoint video enables a client to interactively choose a viewpoint from which to synthesize an image via depth-image-based rendering (DIBR). However, synthesizing a novel viewpoint image using texture and depth maps from two nearby views entails a sizable computation overhead. Further, to reduce transmission rate, recent proposals synthesize the second reference view itself using texture and depth maps of the first reference view via a complex inpainting algorithm to complete large disocclusion holes in the second reference image—a small amount of auxiliary information (AI) is transmitted by sender to aid the inpainting process—resulting in an even higher computation cost. In this paper, we study the optimal tradeoff between transmission rate and client-side complexity, so that in the event that a client device is computation-constrained, complexity of DIBR-based view synthesis can be scalably reduced at the expense of a controlled increase in transmission rate. Specifically, for standard view synthesis paradigm that requires texture and depth maps of two neighboring reference views, we design a dynamic programming algorithm to select the optimal subset of intermediate virtual views for rendering and encoding at server, so that a client performs only video decoding of these views, reducing overall view synthesis complexity. For new view synthesis paradigm that synthesizes the second reference view itself from the first, we optimize the transmission of AI used to assist inpainting of large disocclusion holes, so that some computation-expensive exemplar block search operations are avoided, reducing inpainting complexity. Experimental results show that the proposed schemes can scalably and gracefully reduce client-side complexity, and the proposed optimizations achieve better rate-complexity tradeoff than competing schemes.

Index Terms— Interactive multiview video, depth-image-based rendering, computation complexity

1. INTRODUCTION

Free viewpoint video [1] enables a client to interactively choose a virtual viewpoint from which to synthesize an image via *depth-image-based rendering* (DIBR) [2]. While observation of the 3D scene from different viewpoints can enhance depth perception in the viewer [3], the DIBR view synthesis process using texture and depth maps captured from two nearby views entails a sizable computation overhead. Further, to reduce the transmission rate of free viewpoint video, instead of explicitly encoding the second reference view for synthesis of intermediate views, recent proposals [4] call for view synthesis of the second reference view itself using only texture and depth maps of the first reference view. Using only one reference view for view synthesis typically results in large disocclusion holes in the target image, which necessitates transmission of a small amount of *auxiliary information* (AI) to assist more complex inpainting algorithms [5] to complete the image satisfactorily. For computation-constrained devices like tablets, this computation load may be overwhelming.

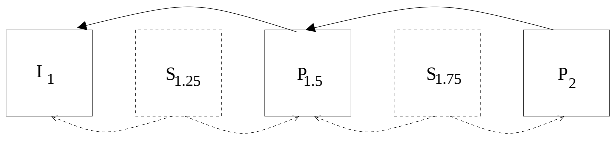


Fig. 1. Example of rate-complexity tradeoff for low-spec devices. In addition to reference views 1 and 2, virtual view 1.5 is rendered and encoded as a P-frame $P_{1.5}$ at server, so that only virtual views 1.25 and 1.75 are synthesized at client (each using two nearby encoded frames as reference, shown as dashed lines), reducing overall complexity.

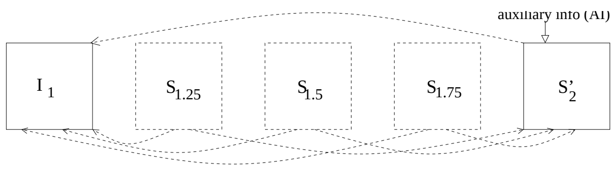


Fig. 2. Example of rate-complexity tradeoff for high-spec devices. Virtual views 1.25, 1.5, 1.75 are synthesized at client using views 1 and 2 as reference (shown as dashed lines). Reference view 2 is reconstructed via view synthesis from reference view 1, with the help of transmitted AI to inpaint disoccluded regions to complete the image.

To address the client complexity problem, in this paper we study the optimal tradeoff between transmission rate and client-side complexity, so that in the event that a client device is computation-constrained, complexity of DIBR-based view synthesis can be gracefully and scalably reduced at the expense of a controlled increase in transmission rate. We first consider the rate-complexity (RC) tradeoff for standard virtual view synthesis paradigm that requires texture and depth maps from two neighboring reference views, as commonly done in the free viewpoint literature [2, 1]. The resulting synthesized image typically has small disoccluded regions that can be filled using simple standard procedures [2]. In this case, we design a dynamic programming (DP) algorithm to select the optimal subset of virtual views between two reference views for rendering and encoding at server, so that a client that desires a free viewpoint "look-around" from the first reference viewpoint to the second can perform video decoding of these frames, reducing overall synthesis complexity. As an example, in Fig. 1 virtual view 1.5 is chosen to be rendered and encoded as a P-frame $P_{1,5}$, so that only virtual views 1.25 and 1.75 are synthesized at client.

For new view synthesis paradigm [4] that synthesizes the second reference view using texture and depth map of the first reference, we study the RC tradeoff for the construction of the second reference view by controlling the selection of AI to assist inpainting of large disocclusion holes. More specifically, if a missing block requires high-complexity search in the filled-in region to identify a suitable exemplar block for copying, then an intra block can be transmitted instead to complete the block at a cost of AI transmission rate increase. See Fig. 2 for an illustration. The RC-optimal sequence of AIs is selected at sender by finding the shortest path in a trellis. Experimental results show that the proposed schemes can scalably and gracefully reduce client-side complexity, and the proposed optimizations achieve better RC tradeoff than competing schemes.

The outline of the paper is as follows. We first discuss related work in Section 2. We then overview our system model in Section 3. We formulate our RC optimization for two types of devices in Section 4. Finally, experimental results and conclusions are presented in Section 5 and 6, respectively.

2. RELATED WORK

In *interactive multiview video streaming* (IMVS) [6], only the single video view currently selected by a client is transmitted from server, lowering transmission rate. The technical challenge in [6] is to design a frame structure that efficiently compresses multiview video *and* provides periodic view-switching mechanisms in the encoded bitstream at the same time. [7] extends the work in [6] by considering interactive streaming of free viewpoint video, where texture and depth maps of two coded views that sandwich the virtual view currently selected by a client are transmitted from server, so that the virtual view can be synthesized at client via DIBR. While we follow the same IMVS setup in [6, 7], we focus exclusively on the RC tradeoff in client-side view synthesis, which is not considered in [7].

To the best of our knowledge, only [8] studies the complexity of DIBR-based view synthesis at decoder in a formal manner. The common assumption in [8] and this work is that overall system computation load (or power consumption) can be reduced by trading off view synthesis complexity with an increase in transmission bitrate. The increase in transmission rate will not cause the same proportional increase in power consumption; this is true for 3G network data transmission, for example, since energy consumption is dominated by the *tail energy* at the end of a transfer, not the actual amount of data transmitted [9]. The key idea in [8] is to compute and transmit from server the disoccluded pixels in an DIBR-synthesized image from a virtual viewpoint, so that the client can simply integrate these server-computed pixels without performing any inpainting. Our RC optimization for standard view synthesis paradigm is similar in that we also seek to minimize the computation load due to disoccluded pixels, but optimize at a frame level. Note that [8] requires pre-encoding and storage of inpainted pixels for every possible virtual view, resulting in a large storage cost.

3. SYSTEM OVERVIEW

Like [7], we consider an IMVS scenario where a server pre-encodes and stores a multiview video content of V captured views, $v \in \{1, \ldots, V\}$, where v corresponds to the physical location of a camera in a 1D array. The IMVS interactivity we provide is *static view-switching*, which means a user can stop the playback of the video in time and navigate to neighboring virtual views of the paused 3D scene[1]. Specifically, a client observes one virtual view at a time denoted by $v + k/K$, where $k \in \mathbb{I}$ and $1 \leq k \leq K-1$. In essence, the client observes a "look-around" of the static 3D scene by viewing virtual views from v to $v+1$. Upon arriving at view $v+1$, the client then has the option of either continuing the static 3D scene "look-around" to camera view $v+2$, or starting temporal video playback at view $v+1$. Dependent on the baseline distance between neighboring cameras, K should be large enough to support a smooth-switching user experience.

Unlike [7] that seeks to minimize expected transmission rate while facilitating application-required periodic view-switching, the challenge in this paper is to design *additional* coded data for transmission, so that complexity of view synthesis at client can be scalably reduced. To understand the computation complexity for the new view synthesis paradigm, we first overview a representation scheme in [4] where texture and depth maps for the second reference view $v+1$ is synthesized using texture and depth maps of a first reference view v, with the help of transmitted AI to aid the inpainting process of large disoccluded regions. It was demonstrated that such a representation has better rate-distortion (RD) performance compared to multiview video coding (MVC) [11] and layered depth video (LDV) [12].

3.1. Auxiliary Information (AI) for Hole-filling

In [4], texture map for the second reference view is constructed as follows. (Depth map can be constructed similarly.) First, texture pixels from the first reference view are mapped to their corresponding locations in the second reference view according to their disparity values[2]. These are the *known pixels* in the *source region* Φ. To fill in values in the *disoccluded pixels* in the *target region* Ω, the Criminisi's algorithm [5] is employed, with the help of additional transmitted AI described below.

Criminisi's algorithm designates a filling order for code blocks with center on the boundary $\delta\Omega$ between source Φ and target region Ω. We do not change the block order, but merely how each block with missing pixels are completed. There are four types of AI to assist in block completion: `skip`, `MV`, `intra` and `pred`. According to the statistics in [4], `pred` is seldom used, and thus we will describe only the other three:

1. `skip` means missing pixels on a block with center on boundary $\delta\Omega$ can be capably inpainted (copied) by the most similar block in Φ, found using the Criminisi's search. Thus, no further information need to be transmitted.
2. `MV` means that Criminisi's search cannot identify a good quality block in source region Φ as replacement. Thus a transmitted motion vector (MV) can help directly locate a best-matched block in Φ for copying.
3. `intra` means no similar block exists in source region Φ. Thus an intra coded block is transmitted for pixel completion.

For our AI implementation, we implemented these three types of AI with three additional modifications beyond [4]. First, `skip` can specify a search range θ, so that complexity for the Criminisi's search can be scalably reduced. Second, if the previous block is coded as `MV`, then the current block also coded as `MV` can have its MV differentially coded, reducing transmission overhead. Third, there is a one-bit flag in `intra` indicating the type of intra-prediction performed. Specifically, if the previous block is also coded as `intra`, then the one bit indicates whether the intra prediction should be performed using the previous block, or using the pixels across boundary $\delta\Omega$, with the isophote[3] as computed in the Criminisi's algorithm [5]

[1][10] showed that humans prefer the visual effects of static view-switching over *dynamic view-switching* [6], where the video is played back in time uninterrupted as users interactively switch to neighboring views. The latter produces effects similar to single-camera pan, which is not novel.

[2]Since there is a one-to-one correspondence between texture and depth, we will use these terms interchangeably.

[3]Isophote is basically the gradient at a pixel rotated by 90 degrees [5], which empirically we found to be a good intra-prediction direction.

as prediction direction. If the previous block is not coded as `intra`, then the one bit indicates whether intra prediction should not be performed, or performed using the pixels across $\delta\Omega$ as described earlier. The properties of the three types of AI are summarized in Table 1.

3.2. Choosing AI for RC Tradeoff

From Table 1, one can find the tradeoff between bit-rate and client-side complexity for different AIs. Suppose it is required that one must achieve a certain reconstruction quality for every block. `intra` is capable of reconstructing to any desired quality (specified by the quantization parameter (QP)) at low decoder complexity. However, the transmission cost of AI `intra` is the highest. If a good matching block does exist in the source region (one that satisfies the quality requirement), its location can be explicitly specified by AI `MV`, with a medium transmission cost, or search using the Criminisi's algorithm, which will incur a large computation cost at decoder. The RC-optimal selection of AI will be formulated in Section 4.2.

4. PROBLEM FORMULATION

We divide the RC optimization of client-side virtual view synthesis into two sections. We first formulate the RC optimization for standard view synthesis paradigm, where each virtual view is synthesized using two nearby reference views. We then formulate RC optimization for a new view synthesis paradigm, where the second reference view is first synthesized from the first reference, and then disoccluded region is constructed using our proposed complexity-scalable inpainting algorithm.

4.1. Rate-Complexity Tradeoff for Standard View Synthesis

For decoders that adopt the standard view synthesis paradigm [2], reference view v and $v+1$, will be encoded as video frames, so that intermediate views between them can be synthesized using *two* references via DIBR. The RC tradeoff is how to select *additional* virtual views between them for rendering and encoding at server, so that complexity at client can be optimally reduced. More specifically, given network bandwidth can support M additional encoded frames for transmission, how to select M intermediate virtual views for encoding so that the complexity of synthesizing the remaining views is minimized.

For simplicity, we assume complexity of synthesizing a view in DIBR is a weighted sum of the number of translated pixels[4] *plus* the number of inpainted pixels due to disocclusion. More precisely, let $\psi(u,w)$ be the complexity of synthesizing intermediate virtual views $u+1/K, \ldots, w-1/K$ between u and w, if each is synthesized via DIBR using encoded views u and w as left and right reference views, respectively. We write:

$$\psi(u,w) = \sum_{y=u+1/K}^{w-1/K} g_y(u,w) + \mu\, h_y(u,w) \tag{1}$$

where $g_y(u,w)$ and $h_y(u,w)$ are the numbers of translated and inpainted pixels in virtual view y respectively, given left and right reference views u and w are used during synthesis, and μ is a weighting parameter. If a computation-intensive inpainting method [13] is used for disocclusion hole-filling, then μ is assigned a value $\gg 1$.

[4]A translated pixel means a texture pixel copied from left and/or right texture map(s) to target view, where the copied location is indicated by the depth map(s). See [2] for details of DIBR.

The objective is to find M additional encoded views, $u_1, \ldots, u_M$, between reference views v and $v+1$, so that the overall complexity is minimized:

$$\min_{u_1,\ldots,u_M} C_L = \sum_{m=0}^{M} \psi(u_m, u_{m+1}) \quad \text{s.t.} \begin{cases} u_0 = v \\ u_{M+1} = v+1 \\ u_m < u_{m+1} \end{cases} \tag{2}$$

(2) can be solved recursively as follows. We first define $C_L(u,m)$ as the minimum complexity from intermediate view $u+1/K$ till $v+1-1/K$, given view u is an encoded view and m remaining intermediate views can be selected for encoding. $C_L(u,m)$ can be defined recursively as:

$$C_L(u,m) = \begin{cases} \min_w \psi(u,w) + C_L(w, m-1) & \text{if } m \geq 1 \\ \psi(u, v+1) & \text{o.w.} \end{cases} \tag{3}$$

Using (3), a recursive call to $C_L(v,M)$ will yield the optimal solution to (2). Further, the complexity of computing (3) can be reduced via *dynamic programming* (DP): each time $C_L(u,m)$ or $\psi(u,w)$ is computed, the solution is stored in entry $[u,m]$ or $[u,w]$ of DP tables, so that repeated calls to the same sub-problem can be looked up instead. This is particularly helpful if (2) needs to be solved multiple times for different M's.

4.2. Rate-Complexity Tradeoff for New View Synthesis

We now formulate the RC optimization problem for the new view synthesis paradigm [4], where the virtual views between references v and $v+1$ are synthesized via DIBR using the two reference images, but the second reference $v+1$ is first synthesized using texture and depth maps of the first reference v only. The problem we pose is how to first construct the second reference image $v+1$ given the first reference v, using a complexity-scalable inpainting algorithm.

First, we assume that all translated pixels in reference view $v+1$ are mapped using texture and depth maps at reference v. The remaining disoccluded pixels in the target region Ω need to be filled with the help of transmitted AI. Specifically, each block b with center on the target / source region boundary $\delta\Omega$ needs to be completed using one of three AI: $\varphi_b \in \{\texttt{skip}(\theta), \texttt{MV}, \texttt{intra}\}$. To select AI φ_b for block b, we write our objective of overall bit-rate as:

$$\min_{\{\varphi_b\}} \sum_b R(\varphi_b, \varphi_{b-1}) \tag{4}$$

where $R(\varphi_b, \varphi_{b-1})$ is the encoding rate for block b if AI φ_b and φ_{b-1} are used for blocks b and $b-1$, respectively. There is a dependency on the AI chosen for previous block $b-1$, because: i) MV can be differentially coded if the consecutive blocks use `MV` as AI, and ii) available intra-prediction types depend on whether previous block is also coded as `intra`, as described in Section 3.1.

It is subject to a distortion constraint for each block b:

$$D(\varphi_b) \leq \bar{d}, \quad \forall b \tag{5}$$

where $D(\varphi_b)$ is the resulting distortion of block b if mode φ_b is used, and an overall complexity constraint for all blocks b's:

$$\sum_b C(\varphi_b) \leq \bar{C} \tag{6}$$

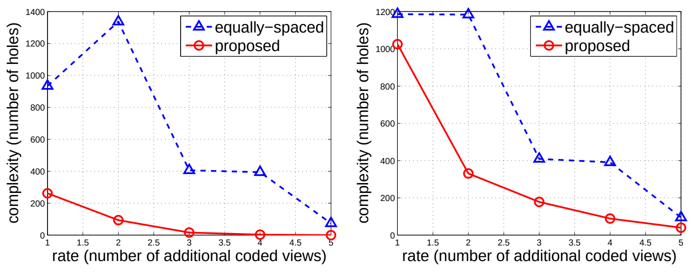
where $C(\varphi_b)$ is the processing time of block b if mode φ_b is used.

Instead of solving (4) directly, we solve the equivalent Lagrangian instead:

$$\min_{\varphi_b \mid D(\varphi_b) \leq \bar{d}} \sum_b R(\varphi_b, \varphi_{b-1}) + \lambda C(\varphi_b) \tag{7}$$

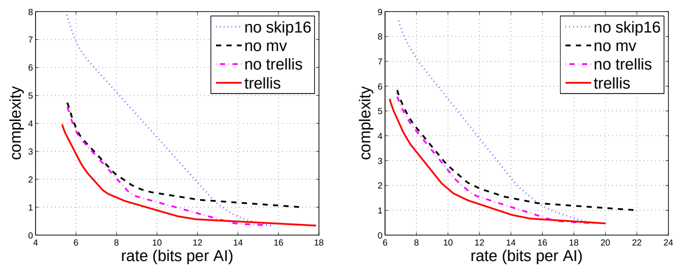
Table 1. Rate-complexity of three types of AI. MV is short for motion vector.

AI	Server Side	Client Side	Bit Rate	Decoder Complexity
`skip`(θ)	signal the search range θ	perform Criminisi's search inside specified range	low	moderate or high depending on θ
`MV`	motion estimation, encode, send MV	decode MV, block copy	moderate	low
`intra`	(intra-prediction), encode, send block	(intra-prediction), decode block	high	low



(a) `Kendo` (b) `Balloons`

Fig. 3. Tradeoff between the number of encoded virtual views and total number of holes after DIBR for standard synthesis.



(a) `Kendo` (b) `Balloons`

Fig. 4. RC tradeoff of view synthesis with the AI-aided hole-filling for new synthesis paradigm.

where λ is a Lagrangian multiplier, selected so that complexity constraint (6) is met for the entire frame. (7) can be solved by first constructing a trellis where each column b of three states correspond to the three types of AI that can be chosen for block b. The edge cost from state φ_{b-1} of column $b-1$ to state φ_b of column b is $R(\varphi_b, \varphi_{b-1}) + \lambda C(\varphi_b)$, except when $D(\varphi_b) > \bar{d}$, in which case the edge cost is infinity. Once the trellis is constructed, one can find find the shortest path in the trellis using the known Viterbi algorithm, which corresponds to the optimal set of AIs for all blocks b's.

5. EXPERIMENTATION

We now demonstrate the performance of our proposed RC optimizations for standard and new synthesis paradigms, respectively. The test sequences used are `Kendo` and `Balloons`[5].

For standard synthesis paradigm, we select a subset of virtual views between two reference views for rendering and encoding at server. A naïve method to select M intermediate virtual views is to pick views that are equally spaced, *i.e.*, insert M encoded views at locations $v + 1/(M + 1)$, $v + 2/(M + 1)$, etc. This method is denoted by "equally-spaced" in Fig. 3. Also shown is our "proposed" scheme. Note that M is a simple proxy for rate. One can alternatively consider actual rates of the M encoded views in the RC optimization, at the cost of a more complex optimization algorithm. This is left for future work.

We see in Fig. 3 that for the same number of encoded views, "proposed" can achieve much lower complexity (measured in number of disocclusion holes). The reason can be explained as follows. In practice, the non-stationary geometrical information of the 3D scene means that disoccluded pixels are not evenly distributed across views, but rather skewed towards views with cameras that are closer to objects in the scene. Our proposed recursive optimization is robust to this non-stationarity and can smartly select the views with more disoccluded pixels, resulting in a dramatic decrease in client-side complexity.

In the second experiment for new synthesis paradigm, we verify the performance of our complexity-scalable DIBR-based view synthesis algorithm by smartly selecting AI to assist inpainting of the second reference view. In Fig. 4, trellis-based optimization ("trellis") can achieve noticeable gain over separate optimization that selects AI for each block individually ("no trellis"), by exploiting rate dependency between two consecutive AIs. Note that since the shortest path in trellis is searched at encoder, there is no client-side complexity increase when using "trellis" instead of "no trellis".

Fig. 4 also shows the performance of trellis-based optimization without `skip`(16)—only `skip`(32) and `skip`(64) are used for AI `skip`—and optimization without `MV` ("no mv"). Without `MV`, we can have comparable performance in low-bitrate region, but there is a larger performance gap in high-bitrate region, which implies that when complexity is more a concern, `MV` is an important AI to effectively reduce complexity. Without `skip`(16), the AI-aided method cannot perform well at low bitrate region, where the complexity is dramatically increased. This observation means that a well-defined search range is critical to scalably reduce the complexity at low bitrate.

6. CONCLUSION

DIBR-based view synthesis at client entails a sizable computation overhead, which may be too costly for computation-constrained devices. In this paper, we propose two techniques to scalably reduce the complexity of view rendering at client, at the expense of a controlled increase of transmission rate from server. For standard view synthesis paradigm, we propose a dynamic programming algorithm to identify subset of virtual views for rendering and encoding at server, so that the client is only required to decode the encoded rendered images with no view synthesis overhead. For a new view synthesis paradigm, we propose to tune the selection of auxiliary information (AI) used to aid inpainting of large disoccluded regions, to optimally trade off transmission rate and inpainting complexity. Experimental results show that the proposed schemes can scalably and gracefully reduce client-side complexity, and the proposed optimizations achieve better rate-complexity tradeoff than competing schemes.

[5]http://www.tanimoto.nuee.nagoya-u.ac.jp/ fukushima/mpegftv/

7. REFERENCES

[1] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," in *IEEE Signal Processing Magazine*, January 2011, vol. 28, no.1.

[2] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," in *Applications of Digital Image Processing XXXII, Proceedings of the SPIE*, 2009, vol. 7443 (2009), pp. 74430T–74430T–11.

[3] C. Zhang, Z. Yin, and D. Florencio, "Improving depth perception with motion parallax and its application in teleconferencing," in *IEEE International Workshop on Multimedia Signal Processing*, Rio de Jeneiro, Brazil, October 2009.

[4] I. Daribo, G. Cheung, T. Maugey, and P. Frossard, "RD optimized auxiliary information for inpainting-based view synthesis," in *3DTV-Conference*, Zurich, Switzerland, October 2012.

[5] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," in *IEEE Transactions on Image Processing*, September 2004, vol. 13, no.9, pp. 1–13.

[6] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," in *IEEE Transactions on Image Processing*, March 2011, vol. 20, no.3, pp. 744–761.

[7] X. Xiu, G. Cheung, and J. Liang, "Delay-cognizant interactive multiview video with free viewpoint synthesis," in *IEEE Transactions on Multimedia*, August 2012, vol. 14, no.4, pp. 1109–1126.

[8] T. Maugey and P. Frossard, "Interactive multiview video system with low decoding complexity," in *IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[9] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: A measurement study and implications for network applications," in *9th ACM SIGCOMM Conference on Internet Measurement Conference*, Chicago, IL, November 2009.

[10] J.-G. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *ACM International Conference on Multimedia*, Singapore, November 2005.

[11] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, November 2007, vol. 17, no.11, pp. 1461–1473.

[12] J. Shade, S. Gortler, L. He, and R. Szeliski, "Layered depth images," in *ACM SIGGRAPH*, New York, NY, September 1998.

[13] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *IEEE Multimedia Signal Processing Workshop*, Saint-Malo, France, October 2010.