Interactive Multiview Video System With Low Complexity 2D Look Around at Decoder

Thomas Maugey, *Member, IEEE*, and Pascal Frossard, *Senior Member, IEEE*

***Abstract*—Multiview video with interactive 2D look around at the receiver is a challenging application with several issues in terms of effective use of storage and bandwidth resources, reactivity of the system, quality of the viewing experience and system complexity. The impression of 3D immersion is highly dependent on the smoothness of the navigation and thus on the number of 2D viewpoints. The classical decoding system for generating virtual views first projects a reference or encoded frame to a given viewpoint and then fills in the holes due to potential occlusions. This last step still constitutes a complex operation with specific software or hardware at the receiver and requires a certain quantity of information from the neighboring frames for ensuring consistency between the virtual images. In this work we propose a new approach that shifts most of the burden due to interactivity from the decoder to the encoder, by anticipating the navigation of the decoder and sending auxiliary information that guarantees temporal and interview consistency. This leads to an additional cost in terms of transmission rate and storage, which we minimize by using optimization techniques based on the user behavior modeling. We show by experiments that the proposed system represents a valid solution for interactive multiview systems with classical decoders.**

***Index Terms*—2D look around viewing, interactivity, multiview video coding, view synthesis.**

I. Introduction

PROVIDING a three dimensional impression in multimedia applications is a challenging task that requires to properly study the sender/receiver interactions. The end-to-end system (*i.e.,* with capture, description, coding, transmission, decoding, display, see for example [1], [2]) sensibly varies depending on the target applications. This is especially true for the display configuration that strongly impacts on the design of the system. The actual configuration of the system has an influence on the definition of quality, where the sensation of immersion takes an important place. While a stereo display directly gives a 3D feeling to the user, a 2D display rather leads to the sensation of looking around objects in the presence of high quality interaction. In this work we assume that the receiver uses a 2D display and we consider the problem of compressing the multiview sequences for interactive delivery.

The coding of these multiview sequences have been widely explored in the scenario where the whole set of frames for all views is transmitted together to servers, edge-servers or client directly. In this configuration, increasing the coding efficiency leads to better exploitation of the interdependencies between the frames. This could be done by extending the motion estimation to inter-view prediction [3], [4]. In some approaches, the inter-frame correlation is exploited using the geometry of the scene, *e.g.,* with depth images [5]. Therefore, novel algorithms have been proposed lately to improve the depth information compression [6], [7] and to smartly balance the rate dedicated to texture and geometry information [8], [9].

Multiview video coding (MVC) schemes bring however strong dependencies between the frames due to predictive coding in the view and time directions. This implies that the extraction of a subset of frames from the multiview bitstream is hardly conceivable; it therefore limits the possibility of using an MVC solution for interactive delivery of multiview videos. One needs to define alternatives to the classical prediction structure of MVC. This can be achieved by limiting the dependencies in the multiview video coding algorithm, or by sending additional information to help navigation at the decoder. There exists a compromise between the level of interactivity (system delay, image quality, frame rate, number of available views, etc.) and the cost of this interactivity service (bandwidth or storage size). As this application typically targets simple devices such as mobile, TV decoder, personal computer, they should not involve too much complexity at the decoder side. However the majority of the existing interactive systems do not pay attention to the computational cost of the decoding algorithm, for example in the synthesis of virtual views. In contrary to what is stated in the majority of the papers related to interactive view switching systems, the design of synthesis algorithms is not obvious. One major issue is the consistency between the rendered image and the neighboring frames. In the MVC framework, this is usually handled by using the neighboring frames [10] for comparison and quality improvement. However, in interactive applications, these neighboring frames are not available and there is nowadays no solution to this consistency problem. However, generating good quality synthesized views and smooth transitions between the cameras is important to create a "look around effect". The latter is necessary to give the impression of immersion in the scene when the stereoscopic display is not available at the receiver, which is our assumption in this work.

In this paper, we propose a new system for multiview video transmission, which enables both a low complex interactivity

Manuscript received January 02, 2012; revised May 25, 2012 and September 05, 2012; accepted October 26, 2012. Date of publication February 08, 2013; date of current version July 15, 2013. This work was supported in part by the Swiss National Science Foundation, under grant 200021-126894. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Zhihai (Henry) He.

The authors are with the Signal Processing Laboratory (LTS4), Institute of Electrical Engineering, École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland (e-mail: thomas.maugey@epfl.ch; pascal.frossard@epfl.ch).

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/TMM.2013.2246147

and acceptable temporal and inter-view consistency. This original scheme is based on the idea of transmitting additive information in order to help the decoding process at the receiver. In the classical video coding schemes, the additive information (or residual information) is usually transmitted to enhance the decoding quality. Here we propose to study the cost and the efficiency of this residual information to decrease the complexity of interactive decoders. Our focus is to study the balance between rates, navigation capabilities and complexity in interactive multiview systems. For that purpose, we build a complete scheme that provides a very satisfying interactivity with low complexity and good viewing experience. We propose to construct and code residual frame information at the server which is used for interactive navigation at the decoder. We define a rate-distortion effective encoding of this information using the user behavior models. We finally show by extensive experiments that our scheme is a valid solution for low complexity interactive navigation systems, and presents an effective trade-off between interactivity and system resources.

The paper is organized as follows. We first introduce in Section II the original idea of our system that consists in encoding some additive residuals (called E frames) in order to help the decoder to reduce the calculation costs due to navigation. In Section III, we detail the complete system that permits the transmission of the multiview video and the E frames. Then, we propose in Section IV some rate-distortion optimization of our system. Finally we show in Section V the performance of our system with extensive experiments.

II. Low Complexity View Synthesis

A. Framework

The target of the proposed system is to deliver to a receiver (or to multiple receivers) a video sequence acquired in a multiview system with a fixed number of color+depth cameras. In addition the user should be able to choose the view and to change the viewpoint. In other words, the receiver only displays a 2D image on a classical video decoder. This image corresponds to one viewing angle in the multiview framework, and the user has the possibility to ask the server to change the viewpoint. The system thus has the objectives of minimizing the delay between the request and the actual viewpoint modification, of providing a high visual quality, of enabling the user to choose between a large number of viewpoints, of minimizing the required rate, and of finally keeping the decoding complexity at a reasonable level. This last requirement leads to non-classical system design, where the server has to prepare additional information used for interactivity.

B. Requirements Posed by Interactivity

For good visual quality, an interactive multiview system has to enable smooth transitions between the different views requested by the user, which is motivated by the need of immersion in the scene. This is called the look around effect [5] and requires a very high number of available views at the decoder. On the one hand it is important to propose a large amount of neighbor views to the user in order to satisfy this desire of immersion; on the other hand increasing the number of cameras is quite costly in terms of hardware. Smooth navigation thus comes through the generation of virtual views at the decoder. Usually, a virtual view synthesis (VVS) algorithm is composed of two steps: i) prediction and ii) error concealment. The prediction step consists in estimating the displacement d of each pixel p from the reference image I_{ref} to the target virtual view I_{vir}, using depth information $z(p)$. This operation is well described in [5], [10], [11] or [12]. The general idea of the process is to first project a pixel in the image plane coordinate (2D), then to the camera coordinate (3D) using depth information and intrinsic camera parameters, and finally to the world coordinate (3D) using the extrinsic camera parameters. In a second part, the inverse process is performed, and the pixel is projected from the world coordinate to its position in the virtual view with the target camera parameters. At the end, we have $I_{\mathrm{vir}}(p+d(p,z)) = I_{\mathrm{ref}}(p)$. If two reference cameras are used for VVS, the above projection is performed once for every camera; a fusion algorithm merges both projection results by considering distances to the reference cameras. This process leaves some holes in the image due to occlusions. They are usually filled by applying an inpainting algorithm, called f_{inp}. Inpainting algorithms [13] have been generally used in order to conceal image areas affected by manual object removal or any other type of local degradation. Some works have proposed adaptation of inpainting techniques to the occlusion filling problem [14], [15]; they use depth and neighboring view information in order to generate estimations that lead to time and view consistency in the reconstructed images. It finally reads

$$I_{\mathrm{vir}}(p') = \begin{cases} I_{\mathrm{ref}}(p) & \text{if } \exists p \text{ s.t. } p + d(p,z) = p' \\ f_{\mathrm{inp}}(p') & \text{otherwise.} \end{cases} \quad (1)$$

This classical VVS algorithm structure however has two major limitations that are generally not taken into account in the literature. First, the dense projection (pixel-based operation) and the inpainting algorithms (block-matching algorithm) are both very complex for a low power decoder [16]. Secondly, if the hole filling algorithm does not use any information taken from the neighboring frames, it reconstructs the images without really guaranteeing temporal or inter-view consistency. Yet, it is commonly admitted that flickering effects (due to inconsistency between frames) are very damageable for the visual quality. Instead of relying purely on VVS with received frames, the decoder thus requires some additional information transmitted by the encoder in order to enable a high quality reconstruction with possibly lower computational requirements. Finally, the implementation of an effective interactive system leads to a trade-off between transmission rate, visual quality and computational complexity at the decoder.

C. E Frames

Based on the observations from the previous section, we propose to build and transmit auxiliary information in order to help the decoder for the creation of virtual views. This additional information needs to be simple to decode, unlike the hash information streams considered in some other schemes [17]. With this additional information, part of the calculation that is usually performed at the user side is shifted to the encoder. We call the additional information as E frames, e, which are built on

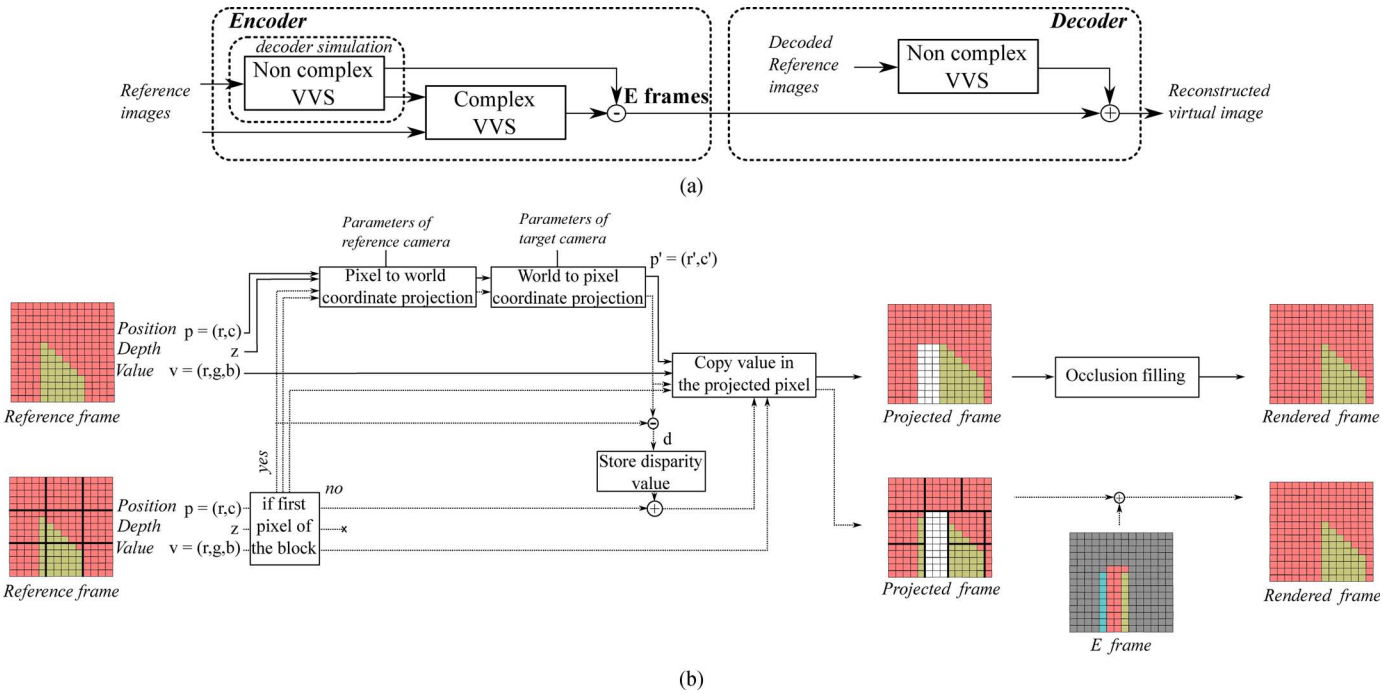


Fig. 1. Description on the E frame generation and their use at the decoder side. (a) The E frames are built by estimating the difference between a non complex VVS and a good quality virtual view. (b) Difference between the complex (plain arrows) and the non-complex (dashed arrows) VVS algorithms performed at the decoder. In the second case the E frames are used to enhance the virtual views.

residual information (see Fig. 1). In other words, the estimated virtual view I_{vir} is the sum of an estimation $\hat{I}_{\text{vir}}$ and a correction e. The idea of transmitting residual information to help the decoder has already been explored in the literature, but with the purpose of enhancing the decoding efficiency. We can cite for example the classical motion compensation residual in most of the common video codecs [18], [19]. We also refer to the layered depth video format [20], [21], where correction information resulting from depth-image based rendering (DIBR) is also considered. In all these methods the residual information is sent for quality enhancement and not necessarily for lowering the computational requirements at decoder. The residual construction is however similar so that our scheme is compatible with the classical decoders: the decoder is simulated at the encoder side and the residual information is the difference between the low complexity decoded version without auxiliary information and a "good quality" version of the signal (Fig. 1(a)).

The first idea for complexity reduction at the receiver side is to remove the very complex occlusion filling step, f_{inp}, from the decoding operation (right part of Fig. 1(b)). In other words, $I_{\text{vir}}(p)$ is no longer calculated with (1) but becomes equal to $\hat{I}_{\text{vir}} + e$ where:

$$\hat{I}_{\text{vir}}(p') = \begin{cases} I_{\text{ref}}(p) & \text{if } \exists p \text{ s.t. } p + d(p,z) = p' \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

and

$$e(p') = \begin{cases} 0 & \text{if } \exists p \text{ s.t. } p + d(p,z) = p' \\ f_{\text{inp}}(p') & \text{otherwise.} \end{cases} \quad (3)$$

The frame e is calculated at the encoder and transmitted to the decoder. Some preliminary results have been given in [22]. The extended study provided in this paper further shows that, for some configurations, sending E frames that contains occluded regions (Fig. 2(a)) decreases the complexity while keeping competitive performance. Whereas shifting the occlusion filling operations from the decoder to the encoder has already a significant impact on the decoding complexity, the projection operation in the construction of virtual views is still too complex for a light hardware: it involves a pixel-based image compensation that involves several matrix multiplications for each displacement calculation. In the scheme presented in this paper, we thus propose to also reduce the complexity of the projection operation at the decoder (left part of Fig. 1(b)). The approach is simple and consists in replacing the pixel precision by a block precision in the projection, where the block size is denoted by B.[1] In other words, instead of calculating displacements pixel by pixel with several matrix multiplications, the proposed low complexity decoder performs projection for each block of pixels and uses the same disparity value for every pixel in the block the disparity value is calculated once, stored and used for the rest of the block). The virtual view I_{vir} remains of the form $\hat{I}_{\text{vir}} + e$, with

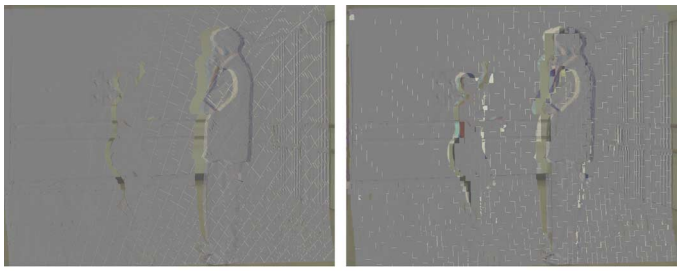
$$\hat{I}_{\text{vir}}(\mathbf{p}') = \begin{cases} I_{\text{ref}}(\mathbf{p}) & \text{if } \exists \mathbf{p} \text{ s.t. } \mathbf{p} + d(\mathbf{p},\mathbf{z}) = \mathbf{p}' \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

(where $\mathbf{p}$ and $\mathbf{p}'$ are $B \times B$ pixel blocks) and

$$e(p') = \begin{cases} I_{\text{vir}}(p') - \hat{I}_{\text{vir}}(p') & \text{if } \exists p \text{ s.t. } p + d(p,z) = p' \\ f_{\text{inp}}(p') & \text{otherwise.} \end{cases} \quad (5)$$

The dimension and thus the coding rate of the depth maps thus decrease in this case. As the quality of the projection is reduced in block-based approaches, we include in the E frames the resulting estimation error, so that the decoder can reconstruct

[1] In this paper, we consider block sizes of 4×4, 8×8 and 16×16.



(a) (b)

Fig. 2. Example of transmitted E frame involving (a) the occluded regions (b) the occluded regions and the blocking errors.

TABLE I
SYSTEM PARAMETERS

Name	Notation	Definition
GOP size	GOP	Size of the GOP used to compressed the reference sequences (color and depth) with JSVM [23]
request interval	N_T	Interval (in msecs) between two requests from the user to the server
request delay	N_D	Time (in msecs) between the request and the effective reception of the demanded frames
Block size	B	size of the blocks used at the projection step of the virtual view synthesis algorithms
No switching probability	p_1	probability that the user does not start any right or left switching
Continue switching probability	p_2	probability that the user continues his (right or left) switching
Stop switching probability	p_3	probability that the user stops his (right or left) switching

views of good quality. The E frames thus contains the error due to the block-based compensation, as shown in Fig. 2(b). At the receiver they are simply added to the projected view generated with the non complex VVS (in Fig. 1(a)).

III. INTERACTIVE MULTIVIEW SYSTEM

Equipped on the original E frame idea proposed in the previous section, we present here the general system that offers a non-complex interactivity to the user. Table I gathers the different notations.

A. User Interactivity

For multiview video transmission systems with interactive 2D display at the receiver, the purpose of enabling the user to change the viewpoint is twofold. First, it lets the user choose the camera position and angle used to observe a scene. This is especially interesting when watching scenes that contain some localized points of interest such as sport, concert or game events. In that purpose, any kind of interactivity may be considered. In other words, random access or smooth navigation in the multiview content can both be envisaged. On the other hand, interactivity can also provide a sensation of immersion in the scene that could replace stereo displays that are not available on every type of devices (mobiles, laptop, etc.). One classical way of rendering three dimensions to the user is to transmit stereo sequences. The problem is that it requires complex and expensive hardwares (glasses, specific screens, etc). However the 3D impression is also provided by the look around effect due to smooth transitions between the different views [24]. This does not require specific hardware on the client's side. It is exactly the objective of our interactive multiview system, where we consider that users might decide to gradually switch views in any direction. For that purpose we also consider the synthetic viewpoints, obtained thanks to the E frames, in order to offer smooth transitions between the captured sequences.

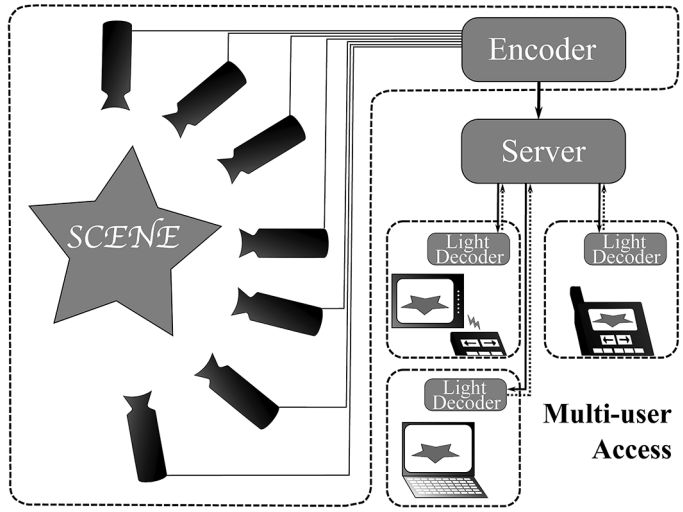


Fig. 3. General system structure.

B. Proposed System

The general structure of the system is composed by different functions: capture, encoding, storage on a server, transmission to the user and decoding, as shown in Fig. 3. After capture, the data (color and depth sequences) are compressed and transmitted to a central server called the *main server (MS)*. The server then processes these sequences before storage. Their stored version is a compressed scalable bitstream that the user could access at the quality (or rate) he wants. For this operation, we use the reference scalable video coder described in [23]. In addition the server generates, codes and stores E frames that correspond to additive information that can be sent to the decoder in order to enhance the virtual view synthesis operation. The E frames described in the previous section reduce the computational power requirements at decoder and increase the quality of the synthesis of virtual views.

At the user side, we assume that a standard video decoder accesses the information stored on the MS via a networks with feedback channel. On one hand the $\mathrm{communication\ user} \rightarrow \mathrm{MS}$ enables the server to get some informations about the user navigation, and on the other hand, the $\mathrm{communication\ MS} \rightarrow \mathrm{user}$ is used to transmit a bitstream that enables the user to navigate between the views. This bitstream corresponds to a group of images called as *set of frame (SoF)*. The $\mathrm{communication\ MS} \leftrightarrow \mathrm{user}$ depends on two parameters that define the level of interactivity in the system. First, we assume that the interval between two messages between client and server is equivalent to N_T (expressed in ms), called as *request interval*; its value is set by the network and can be either fixed or adaptive. This also fixes the interval between two communications from server to client. Second we denote the time spent to transmit the bitstream as *request delay*, N_D, expressed in ms). Note that a real time interactivity is possible as soon as $N_D < N_T$. Note also that N_T and N_D can easily be expressed in number of frames when multiplied by the frame rate.

The proposed system allows multiple users with different capabilities to access to multiview content. Indeed the data description on the MS is not specific to one user due to its scalability. The MS only needs to prepare and transmit data specific to each user as soon as it receives clients' requests. Fig. 5 shows

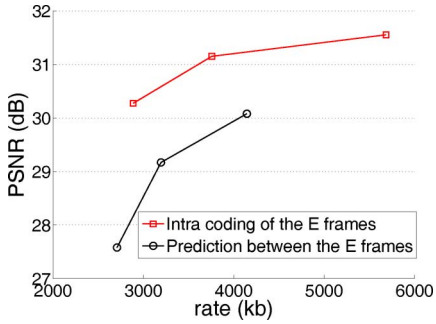


Fig. 4. Prediction-based coding versus Intra coding of the E frames.

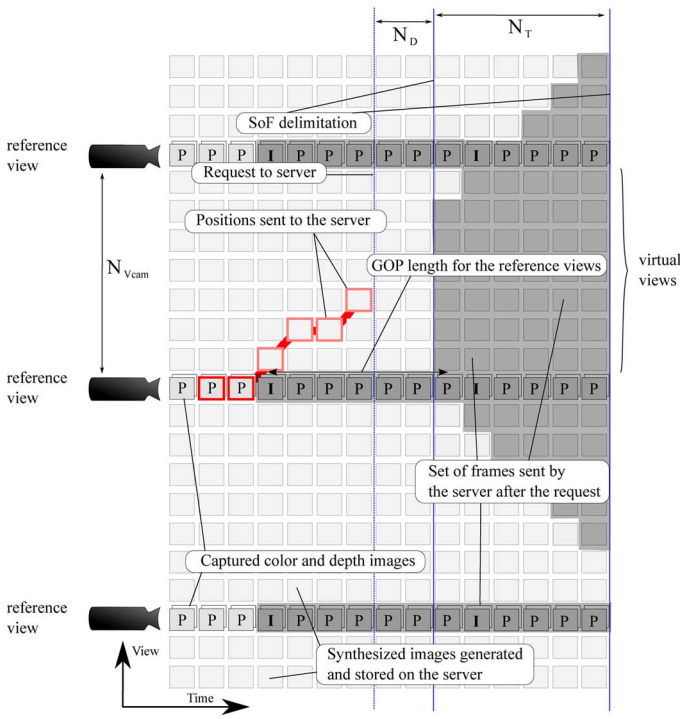


Fig. 5. An example of interaction between server and client. The past navigation path is given by the red images.

the detail of the server-client communication process. The highlighted frames correspond to the ones sent to the client after its request. The request happens N_D frames before the effective beginning of the SoF. The SoF contains all the achievable E frames and all the reference frames that are also achievable and/or involved in the E frame generation. Note that, when N_D becomes larger, the server needs to further anticipate user's navigation and the number of transmitted E frames increases; this is actually imposed by the network.

C. Server

We provide now more details about the multiview content that is present at the server. The reference sequences (color and depth) are stored in a H.264 scalable format [23]. The beginning of each GOP (*i.e.*, the first intra frame) is synchronized between the views, in other words the GOP length is fixed and the I frames occurs at the same time in every view. Then the server also stores additional information for low complexity view synthesis, in the form of E frames. The E frame generation process is summarized in Fig. 1 and is based on two VVS algorithms.

TABLE II
COMPARISON OF USING ONE OR TWO REFERENCE VIEWS FOR THE VVS

Number of reference view used for E frame generation	One	Two
amount of reference transmitted data	**low**	high
E frame size	higher	**lower**
decoding complexity	**lower**	higher
number of E frame stored version	two (left + right)	**one**

The so-called *non complex VVS* corresponds to the algorithm that is used at the decoder. It is designed such that it involves a low computational power for view synthesis. The *complex VVS* that is implemented at the server uses the output of the non-complex VVS, and the original input images in order to generate a higher quality synthesis of virtual images for navigation. This is considered as the target quality that users should experience. The E frame residual compressed on the server corresponds to the difference between the outputs of the non complex and the complex VVS block. They are used by the clients to replicate the output of the complex VVS algorithm, but with a low complexity decoder. The E frames are also coded and stored in an H.264 scalable format so that the users can access the level of quality they need. In contrary to the reference views, the E frames are coded and stored independently for two main reasons. Firstly, since we consider here an interactive scenario where a subset of the stored set of frames is sent to the user, the designed coding strategy tries to avoid dependencies between the frames. Otherwise the transmission of a frame requires the transmission of a set of images that are never displayed but act solely as reference frames, which is clearly suboptimal. This is why we design a scheme where the residuals are transmitted independently. Furthermore, in order to build a decoding strategy that is of low complexity and highly compatible with the current decoder, we mimic the structure of a classical video decoder that only performs the addition of an estimated frame and a residual that has been coded independently. The second reason is a question of performance. We could have designed a predicted coding algorithm based on classical video coding approaches instead of coding E frames with a H.264 Intra strategy. The residual would have been equal to the motion compensation of the previous residual plus another residual. Such an approach is not able to compete with the simple intra coding strategy that we use in our system (see Fig. 4). This is why we only propose an intra coding strategy for the E frames.

Note that the VVS algorithms require color and depth information extracted from one or several reference cameras. The number of reference views (*e.g.,* one or two) used for E frame generation impacts on the amount of data needed to be transmitted and on the storage size on the server. Using one reference view has the advantage of reducing the need of reference information at the decoder. On the other hand, using two views reduces the size of occlusions and then the rate of the E frames. Moreover, it makes the synthesis problem symmetric and then reduces the number of necessary E frame descriptions. With one reference view, the server has to store two versions per E frames (one per neighboring reference camera), while only one description per E frame is necessary with two reference views. These properties are summarized in Table II. We consider in this work that two reference frames are used for the view synthesis.

IV. Rate-Distortion Optimized E Frame Coding

As the server prepares and transmits auxiliary information for offering smooth navigation at the decoder, the storage or bandwidth resource requirements might become important. We propose in this section a method for coding the E frames in a rate-distortion effective way, where we exploit user behavior models.

Let us denote by $F_{v,t}$ the t^{th} frame of the view v which can be a reference or virtual view. For the following we introduce the notion of *frame popularity*, $P(F_{v,t})$ that corresponds to the probability that the user chooses the view v at time t. Under this definition, we have $\sum_k P(F_{k,t}) = 1$ as we assume that users look at one frame and only one frame at each instant t. In this paper, we further assume, that every frame are *a priori* equiprobable. In other words, we assume that the user may watch the scene from every viewpoint with the same probability. Note that this assumption might not be exactly verified in practice because the views can have a different interest depending on the scene content. However, this choice does not limit the generality of our approach and the probability model can be modified without affecting the rest of the system.

For a given user, the frame popularity is however obviously conditioned by the current user position in the multiview context. Indeed, knowing that the user is watching the frame $F_{v',t'}$ obviously impacts on the probability of looking at every $F_{v,t}$ with $t > t'$. To the best of authors' knowledge, it does however not exist any work in the literature that proposes and validates a user navigation model that could help calculating these conditioned probabilities. Thereby, in this work, we propose a simple empiric model that relies on basic observations of user behavior. In other words, we assume that a good user behavior model is known at the server, but the actual instance of such a model is not critical in our optimization methodology.

The navigation model considered in this paper is based on the following intuitive observations. First, the information of the knowledge of current user position $F_{v,t}$ is not sufficient for predicting the probabilities of choosing the next frames; the system needs to know whether the user is already switching from a view to another one or not. Indeed, let us assume that the user is navigating from left to right, *i.e.,* from $F_{v-1,t-1}$ to $F_{v,t}$, the user will more likely continue switching ($F_{v+1,t+1}$) or remain on the current view ($F_{v,t+1}$) than go back in the other direction ($F_{v-1,t+1}$).[2] Besides, if the user has been looking at a particular view, he will more certainly continue to display this same view rather than switching to another view (left or right). Based on these observations, we introduce the following transition probabilities:

$$\begin{aligned} p(v|v,v) &= p_1 \\ p(v-1|v,v) &= p(v+1|v,v) = \frac{1-p_1}{2} \\ p(v+1|v,v-1) &= p(v-1|v,v+1) = p_2 \\ p(v|v,v-1) &= p(v|v,v+1) = p_3 \\ p(v+1|v,v+1) &= p(v-1|v,v-1) = 1-p_2-p_3 \end{aligned}$$

[2]An identical observation is to be done for a switching from right to left.

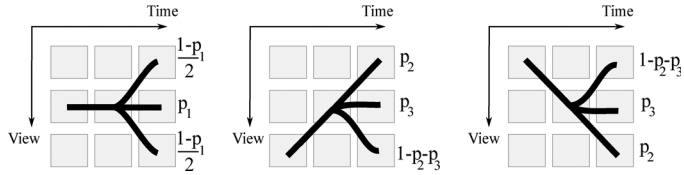


Fig. 6. Graphical representation of the transition probabilities for user navigation.

where $p(n_1|n_2,n_3)$ corresponds to the probability that the user chooses the view n_1 at time t knowing that he chose the view n_2 at $t-1$ and the view n_3 at $t-2$. We dropped the time dependency t in the notation for the sake of clarity, as the same transition probabilities are valid at any time t. They are graphically represented in Fig. 6.

These transition probabilities then permit to calculate the popularity of each frame, conditioned on initial state of the system. Let us assume that at a time t_0 a user is displaying the frame v_0, and at $t-1$ he was watching the view $v_{-1} \in \{v_0-1, v_0, v_0+1\}$. For a request interval N_T and a request delay N_D the set of achievable frames, *i.e.*, the images that can be displayed in the next N_T time instants, is defined by:

$$\mathcal{F}(F_{v_0,t_0}) = \{F_{v,t_0+\tau} \mid \tau \leq N_T + N_D,\ v_0 - \tau ! \leq v \leq v_0 + \tau\}\,.$$

The popularity of each of these frames is calculated as follows:

$$\forall\, t \geq t_0,\ \forall v,$$

$$P(F_{v,t}|F_{v_0,t_0}) = \begin{cases} 0 & \text{if } F_{v,t} \notin \mathcal{F}(F_{v_0,t_0}) \\ \displaystyle\sum_{\substack{v'=v-1 \\ F_{v',t-1} \in \mathcal{F}(F_{v_0,t_0})}}^{v+1} P(F_{v',t-1}|F_{v_0,t_0}) & \\ \cdot \left(\displaystyle\sum_{\substack{v''=v'-1 \\ F_{v'',t-2} \in \mathcal{F}(F_{v_0,t_0})}}^{v'+1} P(F_{v'',t-2}|F_{v_0,t_0}) p(v|v',v'') \right) & \\ \qquad \text{otherwise.} \end{cases}$$

In the following, we explain how we use the frame popularities in order to optimize different parts of the general scheme. In particular, we define a rate-distortion efficient coding strategy that gives more importance and typically more bits to the frames that have the highest popularity.

Let us assume that the server has calculated the frame popularity for every image of the future SoF sent at the receiver. The E frame encoding performance can be improved by the allocation of more bits to the frames that have higher chance to be displayed by the user. Based on the probabilities $P(F_{v,t}|F_{v_0,t_0})$ computed earlier, the encoder implements a rate allocation algorithm that adapts the quantization of the residual information in order to minimize the expected distortion at decoder. In other words, the encoder solves a problem of the form

$$\min_{\mathbf{r}} \sum_v \sum_t D(\mathbf{r}(v,t)) P(F_{v,t}|F_{v_0,t_0}) \quad \text{s.t.} \quad \sum_v \sum_t \mathbf{r}(v,t) \leq R_{\text{total}} \qquad (6)$$

where $\mathbf{r}$ is the rate distribution vector limited by a total bit budget R_{total} and $D(\mathbf{r}(v,t))$ is the distortion of the frame at

instant t in view v, encoded with the rate $\mathbf{r}(v,t)$. As the popularities P do not depend on the rate distribution $\mathbf{r}$, this criterion has a classical form well-known in the rate allocation problem, and thus can be written as:

$$\min_{\mathbf{r}} \sum_{v} \sum_{t} D(\mathbf{r}(v,t)) P(F_{v,t}|F_{v_0,t_0}) + \lambda ||\mathbf{r}||_1$$

where $\lambda > 0$ is the Lagrangian multiplier. The resolution of such a problem is simple since it is separable, *i.e.,* no dependencies between the distortions $D(\mathbf{r}(v,t))$.

In this allocation problem we focused on E frames problem. We leave for future works the search of the optimal balance between the rates of depth, reference texture and auxiliary information (as E frames), since it transcends the scope of the paper. Note that, as the reference frames are used to generate the virtual frames, they are coded with a good quality in order to limit the error propagation in the SoF. Further study on optimal balance between texture and color information can be found in [25], [26]. In our experiments we observe that we must keep a good quality for the depth maps since their compression impacts on the quality of the rendered views and consequently on the size of the residual.

V. Experimental Results

A. Experimental Setup

The experimental results provided in this section have been obtained with the two sequences of color and depth information provided by Microsoft Research [27], the *ballet* and *breakdancer* sequences (at a resolution of 768 pixels $\times$ 1024 pixels and 15 frame per second). Both sequences are 100 frames long and contain eight cameras. The RD curves correspond to an average of N_{path} experiments with different user navigation paths. The generation of the navigation paths is performed with the same model as the one explained in Section IV, which means that the user behavior model used at the server is assumed to match the actual user behavior. We study different aspects of the system performance, like the influence of the system constraints, the storage/bandwidth tradeoff, the role of the user behavior model and the decoding complexity. Most of the experiments have been run with 10 intermediary views between each of the eight reference views[3] (otherwise it is specified). The reference views (color and depth) are coded using the scalable mono-view video codec, JSVM [23]. The GOP are synchronized between the views and between the color and depth sequences. The adopted temporal prediction structures in the GOP consider P and B frames. Since all the views are a priori equiprobable, the quantization parameters adopted in the experiments are the same for every views. The E frames are coded independently with the intra mode of JSVM. We also compare the performance of the proposed systems to baseline solutions.

B. Influence of Network Constraints

As explained in Section III-B, two external system parameters impact on the coding performance. The request interval size N_T corresponds to the level of interactivity allowed by the system. This constraint is often mentioned in the literature but it is not clearly studied. For example, the authors in [17] consider a scheme with a request interval of 1 and state that, in case of larger values, the proposed scheme does not permit navigation during the time between two requests. This approach is not conceivable in our case since we want to provide a look around effect to the user. This is why we consider the request delay as an important parameter, and we enable a free navigation between two requests. Therefore the request interval impacts on the number of frames to be transmitted and thus on the quantity of data sent to the user. In Fig. 7, we plot the RD behavior of the system for N_T that is equal to 133, 266 and 533 ms (which respectively corresponds to 2, 4 and 8 frames). We observe that the penalty due to large values of N_T is however reasonable. This is explained by the low cost of the E frames that does not significantly impact the system preference when their number increases. The performance reduction between two configurations can be easily compensated by decreasing the number of intermediate views depending on the target application constraints. Note also that our scheme can adapt to variations of the request interval during the decoding process since no precalculation is performed on the server that precisely depends on this constraint.

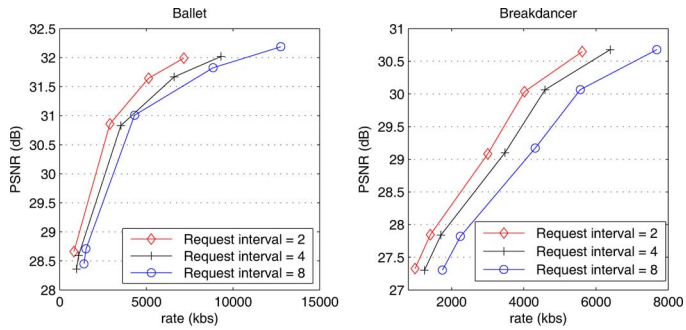


Fig. 7. RD results for different values of N_T (2, 4 and 8 frames which corresponds to 133, 266 and 533 ms) for the *ballet* and *breakdancer* sequences.

In this work, we also consider the constraint N_D that corresponds to the time between a request and the effective transmission of the corresponding data. This parameter depends on the network latency and the time that the server needs to respond to clients' requests. This latter delay could be considerably high for all the methods that consist in transcoding or re-encoding the data in function of the user requests. In our work, everything is prepared and stored on the server beforehand. Then, the response time of the server is negligible since it only corresponds to the time needed to extract the appropriate bitstream from the scalable description stored on the server. The delay is thus dominated by the network latency. We measure the influence of the parameter N_D and we show the results in Fig. 8. Obviously an increasing value of N_D penalizes the performance, but the consequences are not very important because of the reasonable cost of the E frames. As usual, this performance reduction problem can be handled by design tradeoffs, like decreasing of the navigation smoothness with smaller number of views or by limiting the number of available paths.

[3]This corresponds to the lowest number of view that enables smooth transitions between the reference views.

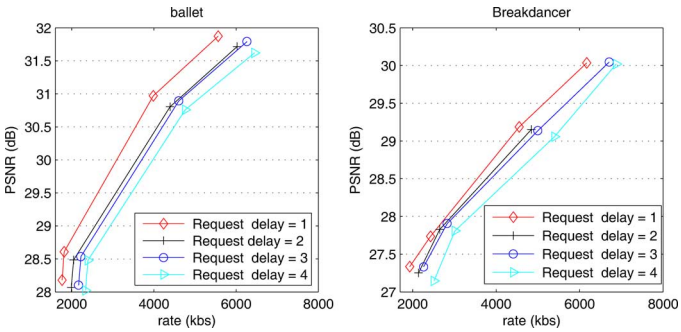


Fig. 8. RD results for different values of N_D (1, 2, 4 and 4 frames which corresponds to 66, 133, 200 and 266 ms) for the *ballet* and *breakdancer* sequences.

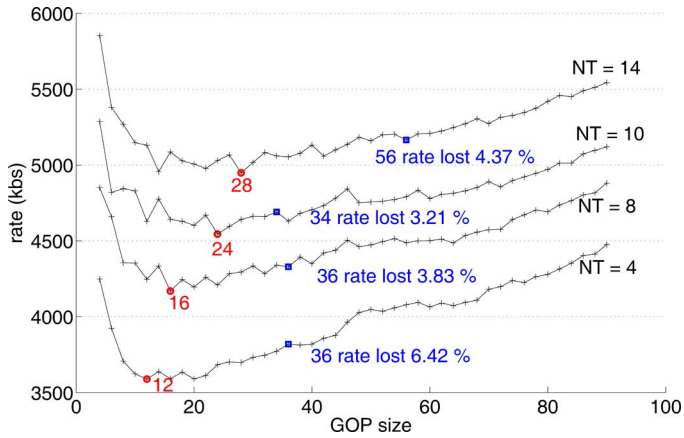


Fig. 9. Transmission rate for reference views versus GOP size for different values of N_T (4, 8, 10 and 14 frames which corresponds to 266, 533, 666 and 933 ms). In red: the optimal GOP size values, when the user behavior model is considered. In blue: the optimal GOP size chosen without user behavior model, along with rate penalty.

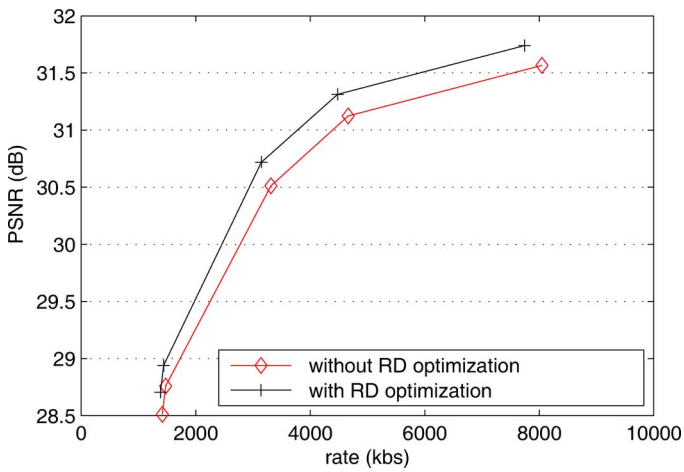


Fig. 10. Quality versus encoding rate (E frames + reference views) for *ballet* sequence.

C. Compromise Between Bandwidth and Storage

In a scenario where a user or multiple users simultaneously receive video sequences, the coding strategy has to deal with a compromise between the storage size on the server and the bandwidth of the transmitted data. A naive scenario in coding all the frames with numerous dependencies and effective prediction with JMVM [19] (*i.e.,* most efficient codec for compressing a whole multiview sequence). However, the coding rate would be tremendous since the display of one frame would require the transmission of numerous other reference frames. This is not efficient in terms of bandwidth. On the opposite, one could consider a situation where the server could store sequences corresponding to all the possible prediction paths in order to optimize the amount of transmitted data. The storage cost becomes huge. These two examples show the intuition that reducing the bandwidth is often obtained by increasing the storage on the server (and *vice versa*) for a given level of interactivity. Some works (*e.g.,* [17]) aim at finding the coding approach that could give the optimal compromise between storage size and bandwidth. In our work, we do not optimize the prediction structure between the frames. Nevertheless, the tradeoff between bandwidth and storage can be achieved by proper coding of additional information, or by adapting the GOP size of reference views. Since the GOP does not have to be aligned on the value N_T, it should be as large as possible for more effective compression, without penalizing delays. However, from a bandwidth point of view, the GOP size should be small in order to reduce the number of reference frames that are not directly used by the clients. In fact, the optimal GOP size depends on the rate of the intra and predicted images in the reference views. Indeed, if the intra frames are much heavier than the P-frames, the GOP size should be longer in order to reduce the number of I-frames. On the contrary, if the I-frames do not cost too much rate, the GOP should be shorter in order to be adapted to the user navigation.

Finally, another important element to consider in the GOP size selection is the behavior of the user. For example, the GOP size should be short if the user often changes views. Given a user behavior, we find the best GOP size that minimizes the transmission rate without penalizing the compression efficiency of the reference frames. In Fig. 9, we show an example of GOP size selection with and without taking the user behavior into account. For a high number of paths, we have simulated the transmission of reference views sequences for different values of N_T and different GOP sizes. For every values of N_T and GOP sizes we have averaged the transmission rate over different navigation paths, and then we have determined the best GOP size for a given value of N_T. In order to analyze the influence of the user behavior model in the decision, we have first determined the GOP size as if all the frames are equiprobable. In a second time, we have generated the path with the transition probabilities defined in Section IV. We have compared the bandwidth optimal GOP sizes in both cases. We can observe that the consideration of the user behavior in the selection of the GOP size leads to a rate saving of up to 6% in some situations.

D. RD Optimized Coding

We analyze now the benefits of considering the user behavior model in the rate-distortion optimized coding of the E frames. We fix the values of $N_T = 533$ ms (8 frames) and $N_D = 0$ ms; we transmit the reference and virtual frames such that the view synthesis is performed with two reference images at the receiver, with a block size of 8. These reference sequences are coded with a GOP size of 16. Fig. 10 shows the comparison

TABLE III
INFLUENCE OF THE PROBABILITY MODEL ON THE BJONTEGAARD GAIN BETWEEN THE CASES WITH AND WITHOUT MODEL-BASED RATE ALLOCATION

p_1	p_2	p_3	type of trajectory	in practice	Rate saving [28]	Rate saving [28] with error of 0.1 in the model
0.9	0.1	0.9	almost no switching	the user remains on a nice viewpoint	-13 %	-7 %
0.3	0.3	0.3	almost random navigation	the user is looking for the best viewpoint	-9 %	-8 %
0.1	0.9	0.1	long switch in the same direction	the user completely changes the viewpoint	-10 %	-9 %
0.1	0.1	0.1	zigzag	the user tests the look around effect	-10 %	-9 %

of the system efficiency with and without the proposed E frame rate allocation introduced in Section IV. We can observe that the consideration of the user behavior model in the E frame coding brings a sensible improvement in terms of average RD performance compared to an encoding that ignores frame popularities.

We then vary the probabilities p_1, p_2, p_3 in the user behavior model of Section IV; for each configuration, we measure the performance for the systems with and without optimized rate allocation, while the decoding process follows the user behavior model. We propose four illustrative or extreme situations that could be seen on actual user navigation processes. The results are presented in Table III. The first remark concerns the very interesting gain that we obtain for every scenario when encoding is optimized by considering a user behavior model. In all the cases, taking into account the user behavior model leads to a non-negligible rate saving greater than 9% (in terms of Bjontegaard metric [28]). Moreover, the different scenario considered in this experiment leads to quite different gains. In the situation where the user performs a random navigation, the gain is less important than in the situation where the navigation is almost deterministic (first line). In real situations, it is obvious that the user behavior would follow different modes depending on the scene content. Our results show that our rate allocation solution leads to interesting rate-distortion performance even with a more evolved probabilistic model that could detect the different navigation modes.

The model considered in the R-D optimization does not always capture the actual user behavior. Even if finding a accurate model is out of the scope of the paper, we propose experiments where the actual behavior at the receiver is slightly different than the one assumed at the server. More precisely, we add an error of 0.1 on the transitions probabilities. Results are shown in Table III. The error in the model reduces the benefits of the RD optimized E frame coding. However, this performance penalty is still reasonable for an error of 10% in the model. It is however interesting to note that the larger penalty is reached in the case of "almost no user switching", which represents an extreme case in view switching scenarios.

E. Decoding Complexity

We now analyze the performance of our system in terms of computational complexity at decoder, which was one of the main motivations for the construction of E frames. The machine used for these experiments is a quad cores, Intel(R) Xeon(R) (2.66 GHz). We consider in these experiments that the network delay, N_D, is zero. In the first column of Table IV, we present the computational time savings of the proposed low-complexity VVS algorithm (block-based disparity compensation and summation of residual information). We consider different block size configurations (1, 4, 8 and 16 pixels) in the disparity compensation and we calculate the computational time savings in our decoder with respect to the complex VVS techniques involved in the classical decoding schemes. The results demonstrate that our scheme leads to computational complexity savings that are really significant. The second column shows the complexity reduction for the whole decoding process, *i.e.,* the reference and E frames decoding processes. The complexity reduction results are pretty convincing about the interest of transmitting additional information as the E frames in interactive multiview systems.

TABLE IV
CALCULATION TIME FOR THE VVS ALGORITHM (SECOND COLUMN) AND THE WHOLE DECODING PROCESS (THIRD COLUMN) FOR DIFFERENT VALUES OF THE BLOCK SIZE B. THE THIRD COLUMN CORRESPONDS TO THE VARIANCE OF THE RESIDUAL INFORMATION

Configuration	Speed up factors of our VVS technique (projection + summation of a residual) wrt to the complex VVS algorithm (dense projection + inpainting)	Speed up factors of frame decoding time of our system (projection + residual decoding + summation of the residual) wrt to the complex decoding approach (dense projection + inpainting)	Variance of the residual
dense projection	45.5	23,5	224.48
$B = 4 \times 4$	625.0	70.4	273.71
$B = 8 \times 8$	2.1×10^5	123.5	292.88
$B = 16 \times 16$	5.8×10^5	163.9	333.02

This considerable decoding time reduction does however not come for free as the third column of Table IV shows it, since the variance of the residual information increases with B. The effective cost of the E frames is shown in Figs. 11 and 12. These figures represent the storage sizes on the server of the three following entities: the reference color sequence, the depth images, and the E frames. In Fig. 11 (resp. Fig. 12), the evolution of these quantities is given in function of VVS block size B (resp. the GOP size of the reference frames) and in function of the number of intermediate views that we considered between the reference views. One can see that the E frames storage cost is not negligible but remains reasonable considering the number of virtual views that they can generate. For instance, the E frame size is slightly higher than twice the size of the color image reference, whereas they permit to generate 10 times more views, which considerably improves the smoothness of the navigation to produce the look around effect. It is also important to note that the storage size does not exactly correspond to the transmission rate. The cost of the E frames during the transmission between the server and a client is given in Table V. In these experiments, we transmit all the information needed for an interactive navigation at the receiver, and we measure, at medium bitrate, the weight of each entity (reference color, reference depth and E frames) in the total bit budget. We still assume here that

TABLE V
RATE DISTRIBUTION IN A SYSTEM WITH 10 INTERMEDIARY VIEWS ($N_T = 133$ ms, 2 FRAMES)

	% of the total bit budget			% of the bit budget per frame		
Configuration	color rate (%)	depth rate (%)	E frames rate (%)	color rate (%)	depth rate (%)	E frames rate (%)
$B = 4 \times 4 + GOP$ 16	61.2	18.7	20.1	74.7	22.8	2.5
$B = 8 \times 8 + GOP$ 16	68.0	10.5	21.5	84.3	13.0	2.7
$B = 16 \times 16 + GOP$ 16	68.5	6.1	25.4	88.8	7.9	3.3
$B = 4 \times 4 + GOP$ 8	56.7	17.4	25.9	73.9	22.7	3.4
$B = 8 \times 8 + GOP$ 8	64.9	9.0	26.0	84.8	11.8	3.4
$B = 16 \times 16 + GOP$ 8	62.4	6.4	31.2	86.8	8.9	4.3

TABLE VI
RATE DISTRIBUTION IN A SYSTEM WITH 5 INTERMEDIARY VIEWS ($N_T = 133$ ms, 2 FRAMES)

	% of the total bit budget			% of the bit budget per frame		
Configuration	color rate (%)	depth rate (%)	E frames rate (%)	color rate (%)	depth rate (%)	E frames rate (%)
$B = 4 \times 4 + GOP$ 8	65.4	20.5	14.1	73.7	23.1	3.2
$B = 8 \times 8 + GOP$ 8	72.4	11.3	16.4	83.3	12.9	3.8
$B = 16 \times 16 + GOP$ 8	73.1	7.4	19.5	86.6	8.8	4.6

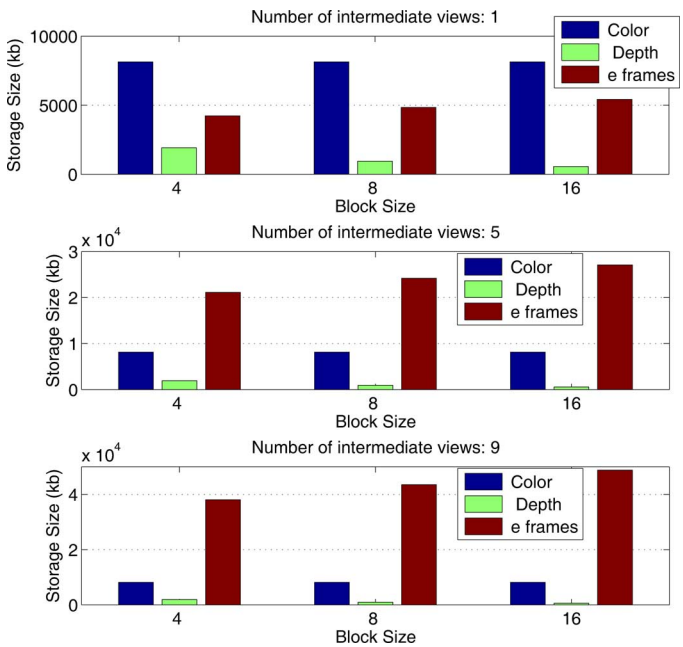


Fig. 11. Storage size on the server as a function of the disparity compensation block size B.

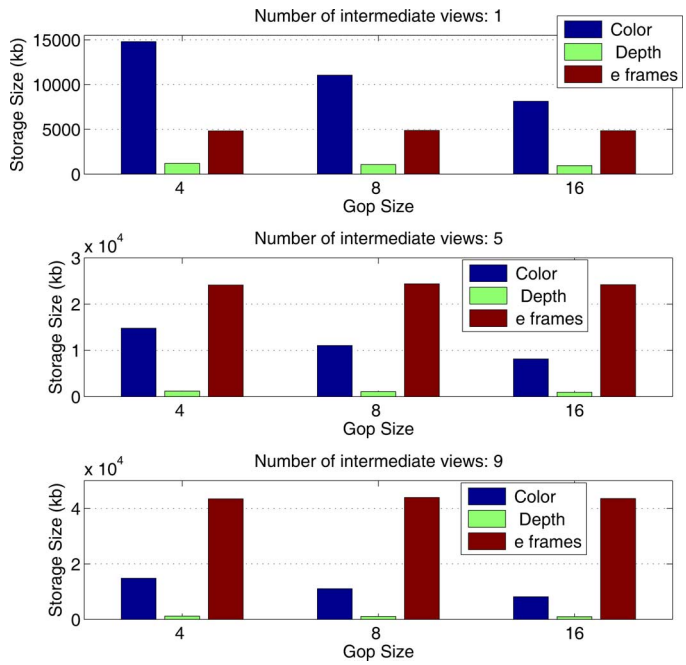


Fig. 12. Storage size on the server as a function of the GOP size in coding the reference views.

the network delay, N_D, is zero. Although this cost is not negligible, it is still smaller than one third of the total bit budget in the case of very smooth view transitions, (*i.e.,* 10 intermediary views). If this cost is too high for given bandwidth constraints, one can reduce the smoothness of the navigation and consider a smaller number of intermediary views. For example, the results in Table VI show that, when the number of intermediary views is set to 5, the relative rate of E frames is sensibly reduced and never increases beyond 1/5 of the total bitrate.

Overall, the experiments shown in this section demonstrate that our scheme manages to provide a considerable complexity reduction with respect to the existing decoding schemes for interactive multiview navigation. The cost of this low complexity decoding is reasonable and further reducible by adapting the interactivity and navigation quality levels.

F. Comparisons With Other Solutions

Our proposed system is a complement to the existing schemes rather than a completely different alternative, since it explores the virtual view synthesis with a low-power decoder assumption. Nevertheless, we present in Fig. 13 some tests that provide hints about the benefits of the E frames solution. In these experiments, we use the following coding parameters: $N_T = 533$ ms (8 frames), $N_D = 0$ ms, GOP size of 8. The E frames are transmitted using the RD optimized coding strategy. The proposed scheme (blue curve, squares) corresponds to a configuration with the block size $B = 16$. In order to measure the importance of E frames in the reconstruction quality, we plot the curves corresponding to the situation where the block size is identical (black line and crosses), but where the E frames are not transmitted and rather replaced at the decoder by a simple

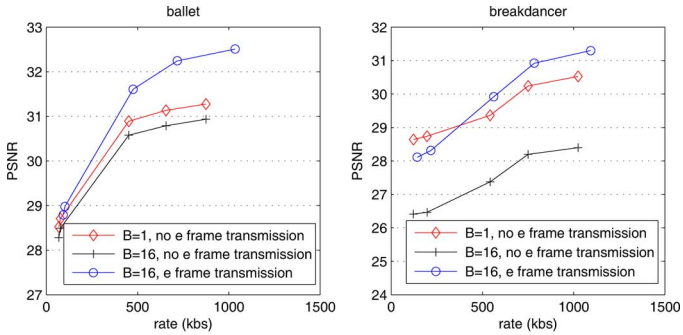


Fig. 13. Comparison of decoding performance when E frames are transmitted or not. (a) *ballet*. (b) *breakdancer*.

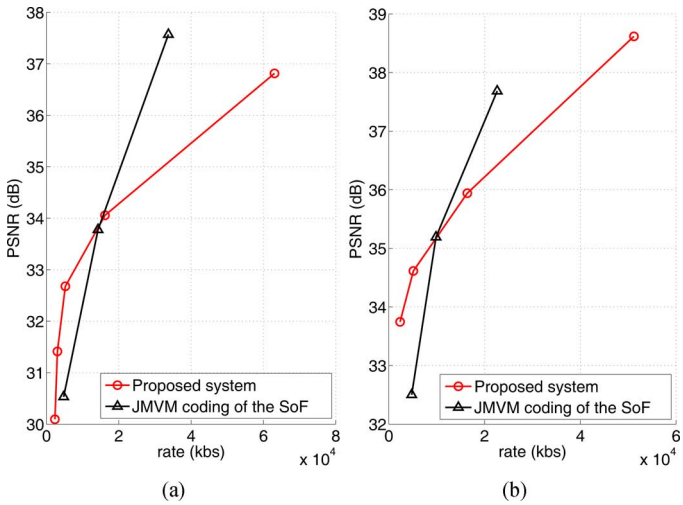


Fig. 14. Comparaison with the JMVM encoding (H.264 MVC) of the requested set of frames (SoF). (a) *ballet*. (b) *breakdancer*.

inpainting method (averaging of the neighboring pixels). This alternative system mimics the behavior of a decoder that cannot afford medium to high complexity in VVS. The resulting curves clearly highlight the benefits of E frames in terms of visual quality. Another interesting comparison is to consider that the decoding process is a bit more powerful and is able to calculate a dense projection ($B = 1$) with a similar inpainting as in the previous scheme (without E frame transmission). The results (represented in red line, losange) shows that for medium and high bitrate, it is worth sending residual information rather than having a very precise projection and bitrate savings. For lower bitrate, the relative performance is sometimes different. This is explained by the fact that the cost of the E frames is proportionally higher at low bitrate, exactly like the motion vectors in classical video coding.

Finally, we compare our scheme with the solution that consists in encoding the whole block of requested frames (set of frames, SoF, that the user would need for navigation) with H.264 MVC. Note that this scheme is not feasible in practice since the server cannot store all the possible blocks of frame nor even encode them in real-time. The comparison is still interesting since it shows the compression efficiency with respect to the classical MVC algorithm. The results are illustrated in Fig. 14. Even if, as expected, H.264 MVC—that relies on very efficient and optimized compression algorithm—outperforms our solution at high rate, it is very interesting to highlight that, at low bitrate, our scheme outperforms H.264 MVC. This shows the efficiency of the compression obtained with our method.

VI. Related Work

Our work serves as a complement to the current literature that tackles the interesting problem of interactive multiview video coding or delivery. We see in this section that the existing methods address the problem of reference view transmission while our system rather studies the question of sending information to help the virtual view synthesis. This is not considered in the techniques detailed below. We review in this section the most relevant works that address the design of interactive video services.

The introduction of interactivity in video systems has first been explored for mono-view video where the problem consists in enabling the user to access every frame in the sequence with a minimum delay. With the classical coding schemes (*e.g.*, H.264) if a user accesses a frame randomly, and if this frame is a predicted frame, the decoder has to receive and decode a set of intermediary frames, which leads to a non-negligible decoding delay. It requires the transmission of useless frames with a penalty in rate-distortion performance. Some solutions have been proposed in order to tackle this problem. One of them is based on SI and SP frames [29], which are images added in H.264 bitstream that help for switching between two bitstreams or for random access. These SP/SI frames are constructed with motion prediction with reasonable encoding sizes. This solution is then less costly than simple solutions that transmit intra frames at the switching instants. Another technique [30] uses a similar idea of building predicted frames that do not depend on the reference image they are predicted from. It uses distributed source coding techniques and transmits hash information in order to construct the side information at the Slepian-Wolf decoder. The solutions proposed to solve the problem of providing interactivity in mono-view scenario lays the foundations of a general problem of adapting the encoding strategy to the user behavior. The general idea is to anticipate the user behavior with two possible alternatives: i) to send additional information or ii) construct a complete prediction structure between the images.

A straightforward extension of mono-view interactivity to multiview systems has been proposed in [31], which adapts the concept of SP/SI frames to view switching. As in mono-view system, these frames constitutes additional information to help the transition between two predefined GOPs. While this approach is appropriate in the case of mono-view video (since the user does not switch too often), it becomes limited for view switching because the user may change the displayed viewpoint frequently, which requires a high quantity of additional information with SP/SI frames. Another approach has been proposed in [32] and reviewed in [33]. It consists in describing the signal in different layers with different levels of prediction. In other words, the encoder provides different descriptions of the signal that can enhance the frame reconstruction when the user changes the viewpoint. The user position is predicted using a Kalman filter. The authors in [34] alternatively propose to store

multiple encodings on the server and to adapt the transmission to the user position. This however brings a high storage cost on the server.

As in the mono-view scenario, some other works adapt the prediction structure to the user behavior. In [35] the system performs real-time encoding and enables the user to switch at precise instants (when the target frame is intra coded). To tackle the limitation of real-time encoding, other works have been proposed such as in [36], [37] where the multiview sequence is encoded with a GoGOP structure, that corresponds to a set of GOP. Inside a GoGOP the frames are coded using different predictions in order to preserve the compression efficiency. On the other hand, the GoGOP are coded independently in order to enable view switching without transmitting large sets of useless frames. The limitation of such methods in the fixed encoding structure, which cannot be easily adapted to different configurations. In some situation, the user may indeed change viewpoints more frequently than in other cases. Interested readers may refer to [24] and [33] that give a good overview of these interactive multiview decoding techniques. The first work that provides an optimization of the prediction structure for interactive decoding has been developed in [38], [39]. The problem is formulated so that the proposed prediction structure reaches a compromise between storage and bandwidth. The possible type of frames are intra frames and predicted frames (with the storage of different motion vectors and residuals). Petrazzuoli *et al.* [40] have recently introduced the idea of using distributed source coding and inter-view prediction for effective multiview switching.

Both ideas of adapting the frame prediction structure and creating additional information have been merged in [17], [41] that extend the work in [38], [39] by adding another possible frame description type based on distributed source coding techniques. This has been recently extended in [42] by taking into account the network delay constraints. With this approach the description of the multiview sequence becomes quite efficient, but this solution does not deal with the question of view synthesis. The scheme proposed in this paper offers a complementary solution to such techniques.

Finally, another important issue in multiview video streaming is the design of systems that enable the transmission of 3D information to multiple heterogeneous users with data representation described above. The purpose of these systems is to meet users' requests under different constraints (*e.g.,* delay, bandwidth, power resources, etc.). In this context, only a few works address at the decoder the problem posed by the limitation of computational complexity during the view rendering at the decoder. In [43], the system contains intermediary servers that performs the virtual view rendering in the place of the light decoders and transmit the resulting images to decoders. However, this approach leads to high processing delays which can be addressed by choosing the appropriate remote rendering systems [44]. The SyncCast system [45] moreover enables the user to interact with each other for improved decoding performance. All of these works can lead to interesting extensions of our solution, where only one server currently delivers the video sequence to multiple users. The format of the data that we have considered moreover considerably reduces this processing delay because everything can be distributed to multiple servers.

VII. Conclusion

In this paper we have studied the question of reducing the required power (or increasing battery lifetime) at the receiver side of an interactive multiview video coding system. Our original idea consists in sending residual frame information that helps smooth view navigation at the decoder. We have shown that the cost of this additional information is reasonable and that it can be even reduced by integrating the user behavior in effective rate allocation strategies. Our work interestingly provides a system that could be readily implemented on the nowadays decoding devices. Finally, it introduces the idea of sending residual information for virtual views, which could trigger some future research work with additional purposes such as the improvement of the compression efficiency.

References

[1] P. Benzie, J. Watson, P. Surman, I. Rakkolainen, K. Hopf, H. Urey, V. Sainov, and C. von Kopylow, "A survey of 3DTV displays: Techniques and technologies," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1647–1658, Nov. 2007.

[2] N. Holliman, N. Dodgson, G. Favalora, and L. Pocket, "Three-dimensional displays: A review and applications analysis," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 362–371, Jun. 2011.

[3] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1461–1473, Nov. 2007.

[4] A. Vetro, T. Wiegand, and G. Sullivan, "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 avc standards," *Proc. IEEE*, vol. 99, no. 4, pp. 626–642, Apr. 2011.

[5] K. Müller, P. Merkle, and T. Wiegand, "3D video representation using depth maps," *Proc. IEEE*, vol. 99, no. 4, pp. 643–656, Apr. 2011.

[6] H. Oh and Y. Ho, "H.264-based depth map sequence coding using motion information of corresponding texture video," *Adv. Image Video Technol.*, vol. 4219, no. 1, pp. 898–907, Dec. 2006.

[7] Y. Morvan, D. Farin, and P. De With, "Depth-image compression based on an RD optimized quadtree decomposition for the transmission of multiview images," in *Proc. IEEE Int. Conf. Image Process.*, San Antonio, TX, USA, Sep. 2007.

[8] P. Merkle, A. Smolic, K. Müller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *Proc. IEEE Int. Conf. Image Process.*, Atlanta, GA, USA, Oct. 2007.

[9] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Müller, P. de With, and T. Wiegand, "The effect of depth compression on multiview rendering quality," in *Proc. 3D TV Conf.*, Istanbul, Turkey, May 2008.

[10] D. Tian, P. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," in *Proc. SPIE, Int. Soc. for Optical Engineer*, 2009, vol. 7443.

[11] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand, "View synthesis for advanced 3D video systems," *EURASIP J. Image Video Process.*, vol. 2008, 2008.

[12] [Online]. Available: http://www.cse.unr.edu/~bebis/CS791E/.

[13] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[14] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Saint Malo, France, Oct. 2010.

[15] K. Oh, S. Yea, and Y. Ho, "Hole filling method using depth based inpainting for view synthesis in free viewpoint television and 3-D video," in *Proc. Picture Coding Symp. (PCS)*, Chicago, IL, USA, May 2009.

[16] F. Shao, G. Jiang, M. Yu, and Y. Zhang, "Object-based depth image-based rendering for a three-dimensional video system by color-correction optimization," *Opt. Eng.*, vol. 50, pp. 047 006–047 006-10, 2011.

[17] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Trans. Image Process.*, vol. 3, no. 3, pp. 744–761, Mar. 2011.

[18] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[19] Joint Multiview Video Model (JMVM) ISO/IEC MPEG & ITU-T VCEG, Marrakech, Morocco, 2007.

[20] B. Bartczak, P. Vandewalle, O. Grau, G. Briand, J. Fournier, P. Kerbiriou, M. Murdoch, M. Müller, R. Goris, and R. van der Vleuten, "Display-independant 3D-TV production and delivery using the layer depth video format," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 477–490, Jun. 2011.
[21] I. Daribo and H. Saito, "A novel inpainting-based layered depth video for 3D-TV," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 533–541, Jun. 2011.
[22] T. Maugey and P. Frossard, "Interactive multiview video system with low decoding complexity," in *Proc. IEEE Int. Conf. Image Process.*, Bruxelles, Belgium, Sep. 2011.
[23] "Joint Scalable Video Model jsvm-8.6", Tech. Rep., 2007, ITU-T and I. JTC1.
[24] A. Smolic and P. Kauff, "Interactive 3D video representation and coding technologies," *Proc. IEEE*, vol. 93, no. 1, pp. 98–110, Jan. 2005.
[25] W. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map coding with distortion estimation of rendered views," in *Proc. SPIE, Int. Soc. for Optical Engineer*, 2010.
[26] V. Davidoiu, T. Maugey, B. Pesquet-Popescu, and P. Frossard, "Rate distortion analysis in a disparity compensated scheme," in *Proc. Int. Conf. Acoust., Speech and Signal Process.*, Prague, Czech Republic, 2010.
[27] [Online]. Available: http://research.microsoft.com/en-us/um/people/sbkang/3dvideodownload/.
[28] G. Bjontegaard, "Calculation of average PSNR differences between RD curves," in *Proc. 13th VCEG-M33 Meeting*, Austin, TX, USA, Apr. 2001.
[29] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 637–644, Jul. 2003.
[30] N. Cheung, H. Wang, and A. Ortega, "Video compression with flexible playback order based on distributed source coding," in *Proc. SPIE Visual Commun. Image Process.*, San Jose, CA, USA, Nov. 2006.
[31] Y. Chen, Y. Wang, K. Ugur, M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standards for 3D video services," *EURASIP J. Adv. Signal Process.*, vol. 2009, pp. 1–13, 2009.
[32] E. Kurutepe, M. Civanlar, and A. Tekalp, "Client-driven selective streaming of multiview video for interactive 3DTV," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1558–1565, Nov. 2007.
[33] A. Tekalp, E. Kurutepe, and M. Civanlar, "3DTV over IP: End-to-end streaming of multiview videos," *IEEE Signal Process. Mag.*, no. 6, pp. 77–87, Nov. 2007.
[34] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, "RD-optimized interactive streaming of multiview video with multiple encodings," *J. Visual Commun. Image Represent.*, vol. 21, no. 5-6, pp. 1–10, Jul. 2010.
[35] J. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *Proc. ACM Int. Conf. Multimedia*, Singapore, 2005, pp. 161–170.
[36] H. Kimata, M. Kitahara, K. Kamikura, and Y. Yashima, "Free-viewpoint video communication using multi-view video coding," *NTT Tech. Rev.*, vol. 2, no. 8, pp. 21–26, Aug. 2004.
[37] S. Shimizu, M. Kitahara, H. Kimata, K. Kamikura, and Y. Yashima, "View scalable multiview video coding using 3-d warping with depth map," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1485–1495, Nov. 2007.
[38] G. Cheung, A. Ortega, and T. Sakamato, "Coding structure optimization for interactive multiview streaming in virtual world observation," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Cairns, Queensland, Australia, Oct. 2008.
[39] G. Cheung, N. Ortega, and A. Cheung, "Generation of redundant frame structure for interactive multiview streaming," in *Proc. Int. Packet Video Workshop*, Seattle, WA, USA, May 2009.
[40] G. Petrazzuoli, M. Cagnazzo, F. Dufaux, and B. Pesquet-Popescu, "Using distributed source coding and depth image based rendering to improve interactive multiview video access," in *Proc. IEEE Int. Conf. Image Process.*, Brussels, Belgium, 2011.
[41] G. Cheung, N. Cheung, and A. Ortega, "Optimized frame structure using distributed source coding for interactive multiview video streaming," in *Proc. IEEE Int. Conf. Image Process.*, Cairo, Egypt, Nov. 2009.
[42] X. Xiu, G. Cheung, and J. Liang, "Frame structure optimization for interactive multiview video streaming with bounded network delay," in *Proc. IEEE Int. Conf. Image Process.*, Brussels, Belgium, Sep. 2011.
[43] S. Shi, W. Jeon, K. Nahrstedt, and R. Campbell, "Real-time remote rendering of 3D video for mobile devices," in *Proc. ACM Int. Conf. Multimedia*, Beijing, China, Oct. 2009.
[44] S. Shi, N. K. a. H. J. M. Kanmali, and R. Campbell, "A high-quality low-delay remote rendering system for 3D video," in *Proc. ACM Int. Conf. Multimedia*, Firenze, Italy, Oct. 2010.
[45] Z. Huang, W. Wu, K. Nahrstedt, R. Rivas, and A. Arefin, "Synccast: Synchronized dissemination in multi-site interactive 3D tele-immersion," in *Proc. ACM Conf. Multimedia Syst. (MMSys '11)*, San Jose, CA, USA, Feb. 2011.

Thomas Maugey (S'09–M'11) graduated from École Supérieure d'Électricité, Supélec, Gif-sur-Yvette, France in 2007. He received the M.Sc. degree in fundamental and applied mathematics from Supélec and Université Paul Verlaine, Metz, France, in 2007. He received his Ph.D. degree in Image and Signal Processing at TELECOM ParisTech, Paris, France in 2010. Since October 2010, he is a postdoctoral researcher at the Signal Processing Laboratory (LTS4) of Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland.

His research interests include monoview and multiview distributed video coding, 3D video communication, data representation, video compression, network coding and view synthesis.

Pascal Frossard (S'96–M'01–SM'04) received the M.S. and Ph.D. degrees, both in electrical engineering, from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1997 and 2000, respectively. Between 2001 and 2003, he was a member of the research staff at the IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, where he worked on media coding and streaming technologies. Since 2003, he has been a faculty at EPFL, where he heads the Signal Processing Laboratory (LTS4). His research interests include image representation and coding, visual information analysis, distributed image processing and communications, and media streaming systems.

Dr. Frossard has been the General Chair of IEEE ICME 2002 and Packet Video 2007. He has been the Technical Program Chair of EUSIPCO 2008, and a member of the organizing or technical program committees of numerous conferences. He has been an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2004-), the IEEE TRANSACTIONS ON IMAGE PROCESSING (2010-) and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006–2011). He is the Vice-Chair of the IEEE Image, Video and Multidimensional Signal Processing Technical Committee (2007-). He is an elected member of the IEEE Visual Signal Processing and Communications Technical Committee (2006-) and of the IEEE Multimedia Systems and Applications Technical Committee (2005-). He has served as Vice-Chair of the IEEE Multimedia Communications Technical Committee (2004–2006) and as a member of the IEEE Multimedia Signal Processing Technical Committee (2004–2007). He received the Swiss NSF Professorship Award in 2003, the IBM Faculty Award in 2005, the IBM Exploratory Stream Analytics Innovation Award in 2008 and the IEEE Transactions on Multimedia Best Paper Award in 2011.