USING DISTRIBUTED SOURCE CODING AND DEPTH IMAGE BASED RENDERING TO IMPROVE INTERACTIVE MULTIVIEW VIDEO ACCESS

Giovanni Petrazzuoli, M. Cagnazzo, Frederic Dufaux, Beatrice Pesquet-Popescu

USING DISTRIBUTED SOURCE CODING AND DEPTH IMAGE BASED RENDERING TO IMPROVE INTERACTIVE MULTIVIEW VIDEO ACCESS

Giovanni Petrazzuoli, Marco Cagnazzo, Frederic Dufaux, Béatrice Pesquet-Popescu

TELECOM-ParisTech, TSI department
46 rue Barrault, F-75634 Paris Cedex 13, FRANCE

ABSTRACT

Multiple-views video is commonly believed to be the next significant achievement in video communications, since it enables new exciting interactive services such as free viewpoint television and immersive teleconferencing. However the interactivity requirement (i.e. allowing the user to change the viewpoint during video streaming) involves a trade-off between storage and bandwidth costs. Several solutions have been proposed in the literature, using redundant predictive frames, Wyner-Ziv frames, or a combination of them.

In this paper, we adopt distributed video coding for interactive multiview video plus depth (MVD), taking advantage of depth image based rendering (DIBR) and depth-aided inpainting to fill the occlusion areas. To the authors' best knowledge, very few works in interactive MVD consider the problem of continuity of the playback during the switching among streams. Therefore we survey the existing solutions, we propose a set of techniques for MVD coding and we compare them. As main results, we observe that DIBR can help in rate reduction (up to 13.36% for the texture video and up to 8.67% for the depth map, wrt the case where DIBR is not used), and we also note that the optimal strategy to combine DIBR and distributed video coding depends on the position of the switching time into the group of pictures. Choosing the best technique on a frame-to-frame basis can further reduce the rate from 1% to 6%.

Index Terms— distributed video coding, interactive TV, 3D-TV, multiview video plus depth, depth map, depth image based rendering, inpainting

1. INTRODUCTION

In recent years, progress in cinema and TV is increasingly pushing towards immersive television [1] where users have the impression of being present at a real event. A very promising application is free viewpoint television (FTV) [2]. It gives to the user the 3D perception of the captured scene with the possibility of changing his viewpoint. This can be obtained by using N Z-cameras that capture the color image as well as a so-called depth map (multiview video plus depth, MVD). The latter can be used to generate intermediate views by depth image based rendering (DIBR) [3].

Obviously, MVD has a huge redundancy: not only in time, as ordinary mono-view video, but also among views (inter-view correlation) and between views and depth maps. All these kinds of redundancies have to be exploited in order to reduce the storage space on the server and the bandwidth used for transmission [4]. These two requirements are equivalent in the context of non-interactive scenarios (like TV broadcasting), when all the video stored on the server will be sent to the user. For example, all the views are sent when MVD is used on a autostereoscopic display. On the contrary, the interactive multiview video streaming (IMVS) [5] is a paradigm that

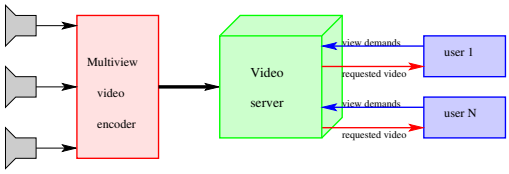

Fig. 1: In the IMVS paradigm the video is first pre-encoded, stored in a video server and afterwards sent to the users, according to their requests

enables the client to select interactively the view that he/she wants to display. Given this constraint, the server will send only the data needed to display the views according to the switch pattern decided by the user. However, the video is first encoded and stored in a server and afterwards it is sent to the clients (see Fig. 1). We would like to minimize both the storage space demanded by the compressed video and the bandwidth needed to interactively send the requested view to the user. These requirements are conflicting in the case of IMVS [5], which makes the problem challenging.

In this paper we deal with the problem of IMVS in the special case of MVD, which has been only sporadically treated in the literature. The problem background and the state of the art are given in section 2. Then, in section 3 we propose a number of techniques based on DIBR and on distributed video coding. Finally, in section 4 we compare them and we determine the best strategy for allowing view-switching in our context. In section 5 we draw some conclusions and present possible future works.

2. STATE OF THE ART

Let us consider a client switching from the view v_1 to v_2. The images of v_2 cannot be encoded using previous images from the same view, since the decoder will not have them. Therefore, two contrasting approaches emerges: on the one hand, we could reduce the bandwidth requirement if for any view, any image is coded N times, using any other views as reference. In this case the storage requirement is multiplied by a factor N (since at the encoding time we do not know which view the user will choose at any time), but the required bandwidth is minimized, since we can send, for the current image, only the residual with respect to the images we have already sent. This approach is called **Redundant P-frames**. On the other hand, we could encode each image only once but as an Intra frame, thus reducing the storage space. However in this case the bandwidth requirements are more demanding, since I-frames need much more bits to be encoded than P's do for the same quality. This approach is called **I-frames** [5].

An alternative approach exploits principles coming from distributed video coding (DVC) [6]. Each view is coded (independently from others) with a DVC technique, such as DISCOVER [7]: some images are intra-coded (they are called Key Frames, KF), while for some other (Wyner-Ziv Frames, WZF) we only send the parity bits of a systematic channel code. At the decoder side, KFs are decoded as usual, and used to estimate the missing WZFs. For example, one out of four frame is a KF, and the three middle WZFs are estimated using motion-compensated image interpolation between current and next KF (such as DISCOVER [7] or HOMI [8]). We explicitly remark that two images are needed in order to perform image interpolation. Finally, the parity bits are used to improve the quality of the estimation, which can be seen as a "noisy" version of the actual WZF. The differences between the estimation (called side information, SI) and the WZF are corrected using the parity bits. This approach can be effectively used in IMVS. When a user switches to a novel view, the next image to be displayed can either be a KF or a WZF. In the first case, it is decoded as usual; in the second one, we just have to adapt the mechanism of SI production, but we do not change the encoded representation of the frame stored on the server, *i.e.* the parity bits of the frame. This is interesting, since on one hand we improve the storage cost (only one version of any view is stored on the server), and on the other hand we do not send I-frames but only WZFs, which in theory could require as few bits as a P-frame, while in practice have an intermediate coding rate between I's ans P's [6]. Of course, any future novel and better DVC scheme will have an immediate impact on this kind of scheme for IMVS.

Cheung et *al.* [5] propose to insert the Wyner-Ziv Frames, used as **M-Frames (Merge Frames)**, in the video stream. In this case, the side information is the previous frame available at the decoder. They find an optimal combination of I-frames, Redundant P-frames and M-frames for the view switching, but they are interested in the case of multiview video coding *without* the depth information.

To the best of our knowledge, only a few papers deal with IMVS in MVD with the constraint of ensuring that the video playback is not interrupted during switching. In MVD the depth maps are usually coded independently of the texture images. Kurutepe et *al.* [9] propose to send to the user also lower bit-rate versions of a set of adjacent views (texture plus depth) in addition to the currently required view, in order to alleviate adverse effects of the unavoidable delay between the time a client requests a new stream and the time it becomes available. Yang et *al.* [10] propose to estimate the disparity map between different views. Differently from DIBR, disparity based algorithm can be applied also when camera parameters are not available. With this algorithm, they can simply generate virtual views in real time.

We conclude this section with some remarks about DIBR and the problem of occlusion filling. Without loss of generality, we consider a pair of cameras. Let (u_1, v_1) be the projection of a point on the camera 1 image plane. The position (u_2, v_2) of the same point on the camera 2 can be obtained by the knowledge of the intrinsic and extrinsic parameters of the two cameras. This allows to obtain a synthesis of the depth map and texture for the second view given the first one. The second camera can be a real camera or a virtual viewpoint. In this paper we refer only on the first case: then, this synthesization will be used only as estimation for the second camera. Obviously, there are points of the second view that are not visible in the first view (occlusion points): this causes the presence of occlusion areas in the synthesized image. These areas can be filled by inpainting techniques [11]. In particular, the depth maps, due to their smooth nature, can be inpainted by Bertalmio technique [12], which is based on isotropic diffusion by using the Navier-Stokes and fluid dynamics equations. On the other hand, for the texture image, one of the most popular technique is the Criminisi inpainting [13]: in this algorithm the texture is inpainted in the isophote direction. In the case of large baseline between the two cameras, Daribo and Pesquet-Popescu [14] modify the Criminisi inpainting by introducing a term related to the depth: in fact, the depth helps to distinguish if a pixel belongs to foreground or background and give higher priority to the patch that overlays at the same depth level. We will use this technique in order to perform an enhanced DIBR in our proposed methods. This will allow a better quality for both the texture image and the depth map.

Name	Description
m	switching time: user wants view 1 up to $m-1$ and then view 2
k	time of the KF of the GOP affected by the switch on view 2. $k \leq m$
N	GOP size; for simplicity we only consider the case $N = 4$
I_n	decoded frame for the first view at instant n
J_n	an estimated frame of the target view, taken at time n and used as reference for motion interpolation for the remaining WZFs of the GOP
$\widetilde{I}_n$	estimation of the second view at time n obtained by depth-aided DIBR [14] on I_n
$\widehat{I}_n$	$\widetilde{I}_n$ corrected by parity bits

Table 1: Notation

3. PROPOSED METHODS FOR INTERACTIVE MVD

For the sake of simplicity, we consider the case where we have two Z-cameras, there is a KF out of N images in time domain (i.e. a GOP size of N), and the KFs on the two views are shifted by half a GOP. For each view the texture and the depth maps are coded independently from the other view, e.g. using DISCOVER. This will assure an efficient use of server storage, since DISCOVER performances are better than all-Intra coding [7]. However we note that, since the encoding process is performed *off-line*, at that time we do not know which views will be used by the decoder to estimate the WZFs, so we do not know how many bits it will demand. Therefore we have to store all the parity bits, which can be more than the bits that will actually be demanded at the decoding time. However this is unavoidable in the Wyner-Ziv Frame approach. Still, this represents less than storing all possible P residuals in a classical coding framework.

We have to choose what information is send to the decoder when the user makes a switch between two views, and how this is used. We propose six different techniques, exploiting DIBR and DVC. In order to describe the proposed methods, we introduce the following notation. We call m the instant of the switch: that is the user wants view 1 up to the time $m-1$, and then he/she wants the other view. We call k the instant when the GOP of frame m begins in view 2: in other words the previous KF for the target view happens to be in k. Because of the periodical GOP structure, we have to consider only the cases $m = k, k+1, k+2, \ldots k+N-1$, namely the cases when the switch corresponds to a KF or to one of the $N-1$ WZF of the GOP. However, since the GOP structures of the views are shifted by half a GOP, the frames globally concerned by the switch are those from $k-N/2$ (beginning of the current GOP on view 1) to $k+N-1$ (end of the current GOP on view 2).

Moreover, we call J_n the reconstructed image for the second view *used for creating the side information*. This image will not necessary be displayed, but will be used, together with the KF $k+N$ that will always be sent ($m < k+N$ by definition), to produce the estimation for the WZF of the current GOP in the target view. We call this image the *reference image*. Finally we denote by $\widetilde{I}_n$ the estimation of the target view at time n obtained by only using

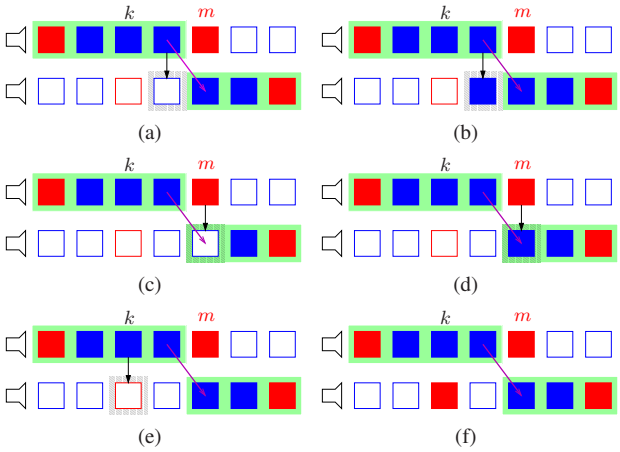

Fig. 2: Solutions for $N = 4$ and $m = k + 2$: (a) A-DIBR; (b) A-DIBRc; (c) I-DIBR; (d) I-DIBRc; (e) GOPp; (f) noDIBR. The KFs are in red and the WZFs in blue. The frames with green background are displayed. The filled frames are sent by the video server to the user. The black arrow shows that DIBR is applied.

depth-aided DIBR [14] on the first view, and with $\widehat{I}_n$ an improved version of this image, obtained by using the parity bits of the second view image to correct $\widetilde{I}_n$. The introduced notation is summarized in Table 1 for ease of reading.

Now we can conceive several methods for IMVS using DIBR. They are shown in Fig. 2 for $N = 4$ and $m = k + 2$, but this can be easily generalized to any N and m. A first one, that we call **Advance DIBR (A-DIBR)** consists in computing J_{m-1} at the decoder by using DIBR on the last received image from the first view. Next images in the GOP will be interpolated using $J_{m-1} = \widetilde{I}_{m-1}$ and the next KF on the target view, and this side information will be corrected with the parity bits of the target view. A variation of this method, that we can use only when $m \neq k+1$ consists in using the parity bits in order to improve the reference image: $J_{m-1} = \widehat{I}_{m-1}$. This second method is not necessarily better than the first one, since the higher reference quality is traded-off with a higher coding rate (the parity bits were not sent in the previous case). We call this method **Advance DIBR + correction (A-DIBRc)**.

Another couple of methods is obtained if we perform DIBR at switch time instead of the previous instant. This means that we send to the decoder the parity bits (or the key frame) of the first view at time m, so that it can reconstruct the latter and use it to perform DIBR, without (**Immediate DIBR, I-DIBR**) or with sending the parity bits for view 2 at time m (**Immediate DIBR + correction, I-DIBRc**).

We can also think about preserving the GOP decoding structure in the second view. If we compute the previous KF on the GOP of the target view by DIBR, then we can use it to perform image interpolation in the GOP. In this case we just need to compute J_k as $\widetilde{I}_k$. We call this method **GOP preserving with DIBR (GOPp)**.

The last method does not use DIBR. This method consists in directly sending the key frame of the current GOP for the target view. With respect to the GOPp method, it demands more rate but provides a better representation of the reference frame. This last method is called **GOP preserving without DIBR (noDIBR)**.

The proposed methods are summarized in Table 2, by describing the time chosen to compute the reference image and the technique used to perform this operation. The operation mode of the six methods for the case $m = k + 2$ is shown in Fig. 2(a). Note that not all the methods can be applied for each m, in fact A-DIBRc (resp. I-DIBRc) are applicable only if in $m - 1$ (resp. m) there is a WZF for the second view.

method	time for reference image	computation of the reference image
A-DIBR	$m-1$	$\widetilde{I}_{m-1}$
A-DIBRc	$m-1$	$\widehat{I}_{m-1}$
I-DIBR	m	$\widetilde{I}_m$
I-DIBRc	m	$\widehat{I}_m$
GOPp	k	$\widetilde{I}_k$
noDIBR	k	KF of target view

Table 2: Summary of methods

Now we compute the rate and the distortion provided by the different method for the frames concerned by the switch, namely from $k - N/2$ to $k + N - 1$. We note with $R_n^{(i)}$ the rate for the frame at instant n of the i-th view, with R_{ST} the rate needed for storage, and with R_{BW} the bandwidth needed for sending the requested video, and with R_{SW} the frames that are sent to the decoder during the switch, but not necessarily visualized from the user:

$$R_{\mathrm{ST}} = \sum_{n=k-N/2}^{k+N-1} R_n^{(1)} + \sum_{n=k-N/2}^{k+N-1} R_n^{(2)} \quad (1)$$

$$R_{\mathrm{BW}} = \sum_{n=k-N/2}^{m-1} R_n^{(1)} + \sum_{n=m+1}^{k+N-1} R_n^{(2)} + R_{k+N/2}^{(1)} + R_{\mathrm{SW}} \quad (2)$$

Eq. (1) accounts for the fact that all encoded frames have to be stored (independently of the IMVS method). However, only some of them have to be sent (see Eq. (2)), namely frames from $k-2$ to $m-1$ for the first view and from $m+1$ to the end of the GOP for the second. Moreover, we always send the next KF of the first view, since it is needed to obtain $I_{k-2}, \ldots, I_{m-1}$. Then, according to the chosen technique, some other frames should be sent (see R_{SW} in Eq. (1)). In particular, we always send the encoded frame of target view at time m (excepted for I-DIBR, where it is obtained as $\widetilde{I}_m$). Moreover we send the parity bits for the frame obtained by DIBR in methods A-DIBRc and I-DIBRc (if the corresponding frame is a WZF), the KF k of second view for the noDIBR method and possibly, for GOPp, frame $m-1$ of the second view [1].

As far as the distortion is concerned, we consider only frames that are visualized by the user. Therefore, the distortion is computed as $D = \sum_{n=k-N/2}^{m-1} D_n^{(1)} + \sum_{n=m}^{k+N-1} D_n^{(2)}$.

4. EXPERIMENTAL RESULTS

In our tests we use the first 100 frames of the MVD sequences Ballet and Breakdancers (views 3 and 4). The texture sequence and the depth maps are independently encoded for both cameras with the DISCOVER algorithm [7]. In order to compare the algorithms, five switchs from one view to the other are performed for each sequence. The results are presented in Table 3, where we report the Bjontegaard metric [15] of the first five methods with respect to the sixth (noDIBR).

We remark that, according to the position of the switching point within the GOP (that is supposed to be equal to 4), the performance of the methods change, and some of them become equivalent, while

[1] Only if it is needed in hierarchical temporal interpolation, e.g. for $m = k + 3$ and $N = 4$.

ballet				
	texture		depth	
method	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	13.40	-0.80	13.76	-0.93
$m = k + 1$				
A-DIBR/GOPp	**4.81**	**-0.27**	**1.76**	**-0.13**
I-DIBR	10.07	-0.59	9.69	-0.62
I-DIBRc	3.24	-0.18	1.73	-0.11
$m = k + 2$				
A-DIBR	0.54	-0.02	-1.54	0.08
A-DIBRc	-0.42	0.02	-2.11	0.13
I-DIBR	7.34	-0.43	4.58	-0.29
I-DIBRc	**-2.22**	**0.12**	**-3.57**	**0.21**
GOPp	-0.77	0.07	-1.67	0.09
$m = k + 3$				
A-DIBR	-5.53	0.33	-6.67	0.41
A-DIBRc	**-5.82**	**0.34**	**-6.67**	**0.41**
I-DIBR	18.29	-1.04	17.21	-1.05
I-DIBRc	8.93	-0.50	8.65	-0.52
GOPp	-1.53	0.09	-3.05	0.18
breakdancers				
$m = k$				
I-DIBR/GOPp	1.52	-0.08	0.48	-0.01
$m = k + 1$				
A-DIBR/GOPp	**-6.21**	**0.33**	**-5.82**	**0.36**
I-DIBR	0.89	-0.04	-0.30	0.03
I-DIBRc	-4.42	0.22	-4.38	0.26
$m = k + 2$				
A-DIBR	-5.76	0.29	-4.64	0.28
A-DIBRc	-5.77	0.29	-5.01	0.29
I-DIBR	-3.55	0.18	-2.88	0.19
I-DIBRc	**-11.01**	**0.56**	**-8.04**	**0.50**
GOPp	-10.98	0.58	-4.94	0.33
$m = k + 3$				
A-DIBR	-9.26	0.48	-8.42	0.50
A-DIBRc	**-13.36**	**0.68**	**-8.67**	**0.55**
I-DIBR	12.38	-0.59	17.60	-0.99
I-DIBRc	5.18	-0.22	12.80	-0.71
GOPp	-10.48	0.54	-5.18	0.34

Table 3: Rate-distortion performance of all techniques wrt noDIBR by Bjontegaard metric [15]

some other are not available. For example, for $m = k$ the best solution is noDIBR, which was expected: when the switching point coincides with a KF, the best is to send it directly. Moreover in this case, the A-DIBR/A-DIBRc methods would not be possible, since the frame $m-1$ belongs to another GOP. For $m = k+1$ the best solution is A-DIBR/GOPp (The two methods coincide since the frame in $m - 1$ is the previous KF). For $m = k + 2$ the best solution is I-DIBRc: in this case A-DIBR is less effective because the reference frame is farther apart. For $m = k + 3$ the most effective method is A-DIBRc because it demands the smallest number of non-displayed frames. Note that, when parity bits are available, it is always better to send them since the quality of the frame J_n used for creating the side information for the rest of the GOP is improved. Therefore next estimated frames are improved as well less parity bits are needed to correct them. So, I-DIBRc and A-DIBRc should be used instead of I-DIBR and A-DIBR, respectively, whenever the parity bits are available. As a conclusion, we remark that no technique is always the best. We highlight this by comparing the 6 proposed methods (see Table 4) with the optimal combination which uses the best method for each value of m, that is I-DIBR/GOPp for $m = k$, A-DIBR/GOPp for $m = k + 1$, I-DIBRc for $m = k + 2$ and A-DIBRc for $m = k + 3$, and averaging over all possible values of m. We see that if we want to use only one method, it is better to use A-DIBRc. If we want to gain a further almost 1% we have to use the optimal combination.

method	Δ_R [%]	Δ_{PSNR} [dB]
A-DIBRc	0.88	-0.06
I-DIBRc	6.04	-0.44
GOPp	2.94	-0.17
no DIBR	4.22	-0.22

Table 4: Rate-distortion performance for texture video (averaged over the two sequences) by the Bjontegaard metric [15] wrt the optimal combination.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we explore interactive multiview video video access in the context of free viewpoint television where both the texture images and the depth maps are coded. Our goal is to assure that during the switching the temporal playback is not interrupted. The video stream is coded by DVC techniques. Since the depth maps are available, DIBR can be applied in order to have an estimation of the other view. Several algorithms are proposed and compared w.r.t. a solution that does not use DIBR algorithm. The results are encouraging: the rate reduction for texture image is up to 13.36%. We have also shown that there is not an optimal method for any switching instant: the best solution is choosing a different method according to it. As future work, we want to apply extrapolation techniques for side information generation instead of interpolation techniques in order to reduce the number of frames that are sent to the decoder and are not displayed by the user, because interpolation introduces a small delay for the reconstruction at the decoder.

6. REFERENCES

[1] C. Fehn, P. Kauff, O. Schreer, and R. Schäfer, "Interactive virtual view video for immersive TV applications," in *Proc. of Intern. Broadcast Conf*, 2001, pp. 14–18.

[2] M. Tanimoto, M.P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," *Signal Processing Magazine, IEEE*, vol. 28, no. 1, pp. 67 –76, 2011.

[3] C. Fehn, "A 3D-TV Approach Using Depth-Image-Based Rendering (DIBR)," in *Proceedings of 3rd IASTED Conference on Visualization, Imaging, and Image Processing*, Benalmádena, Spain, Sept. 2003, pp. 482–487.

[4] M. Flierl and B. Girod, "Multiview video compression," *Signal Processing Magazine, IEEE*, vol. 24, no. 6, pp. 66–76, Nov. 2007.

[5] G. Cheung, A. Ortega, and N. M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *Image Proc., IEEE Trans.*, 2010.

[6] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proc. IEEE*, vol. 93, no. 1, pp. 71–83, Jan. 2005.

[7] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The DISCOVER codec: Architecture, techniques and evaluation, picture coding symposium," in *Coding of Audio-Visual Objects, Part 10: Advanced Video Coding, 1st Edition*, 2007.

[8] G. Petrazzuoli, M. Cagnazzo, and B. Pesquet-Popescu, "High order motion interpolation for side information improvement in DVC," in *Acoustics Speech and Signal Processing (ICASSP) IEEE Int. Conf. on*, 2010, pp. 2342 –2345.

[9] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Interactive transport of multiview videos for 3dtv applications," in *Packet VideoWorkshop*, Hangzhou, China, 2006.

[10] Y. Yang, M. Yu, G.Y. Jiang, and Z.J. Peng, "A transmission and interaction oriented free viewpoint video system," *Int. J. Circuits Syst. Signal Process*, vol. 1(4), pp. 310–316, 2007.

[11] M.Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. ACM Conf. Comp. Graphics (SIGGRAPH)*, New Orleans, USA, 2000.

[12] M. Bertalmio, A.L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2001.

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

[14] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Multimedia Signal Processing (MMSP), 2010 IEEE International Workshop on*, 2010, pp. 167 –170.

[15] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," in *VCEG Meeting*, Austin, USA, Apr. 2001.