EVALUATION OF BACKWARD MAPPING DIBR FOR FVV APPLICATIONS

Daniel Berjón[1], *Alexander Hornung*[2], *Francisco Morán*[1], *Aljoscha Smolic*[2]

[1]Universidad Politécnica de Madrid
Av. Complutense 30, 28040 Madrid, Spain
{dbd,fmb}@gti.ssr.upm.es

[2]Disney Research, Zurich
Clausiusstrasse 49, 8092 Zurich, Switzerland
{hornung,smolic}@disneyresearch.com

ABSTRACT

In this paper, we explore the challenges posed by wide baseline camera configurations for depth-image-based rendering, which should provide greater freedom for choosing the virtual viewpoint in a Free Viewpoint Video context, compared with the usual camera configurations intended for use in 3DTV settings. We implement a backward mapping approach with a custom filtering scheme based on median filters. Whilst the results back our initial assumption that this camera configuration provides good mobility, we show that the usual encoding for depth information referred to a global reference system is wrong and reference systems local to each camera should be used instead.

Index Terms— DIBR, Backward mapping, Free Viewpoint Video, FVV, depth coding

1. INTRODUCTION

In the past few years the field of free viewpoint video (FVV) has seen much interest, both due to the emergence of 3D cinema and TV as a means to generate the several views that these systems need, or as a new creative tool to obtain so called "bullet-time" effects such as those of the famous scene of the sci-fi movie *The Matrix*. However, FVV is a term that describes a goal rather than a specific way to achieve it. Actually, there are a number of technologies that serve this purpose with varying degrees of success depending on the constraints of the specific application. Methods for achieving FVV range from the obtention of an explicit model based on geometric meshes or voxels [1, 2] to elemental manipulation of pixels from source images.

The former class of methods offers great flexibility for moving the camera because once the model has been obtained, rendering new views turns into standard computer graphics procedure. However, this flexibility comes at a price: there is no known general method to obtain high-quality geometric models of an unspecified scene and, even in those kinds of scenes for which models can be obtained [3], reconstruction methods are computationally expensive.

On the opposite side of the spectrum, there are methods in which every pixel in the virtual image results from a combination of contributions from pixels in the original images and no attempt is made at understanding the semantic structure of the 3D scene. These methods are collectively known as image-based rendering (IBR) or depth-image-based rendering (DIBR) if additional depth information is available.

The fundamental idea of DIBR algorithms [4, 5] is that, having images from one or more cameras containing both color and depth information, each pixel is easily backprojected to a 3D point, all pixels together forming a 3D point cloud. Then, this point cloud can be projected onto a new virtual camera in order to obtain an approximation of the image that a real camera with the same configuration would have seen.

There are several problems with DIBR synthesis that must be dealt with when mapping images to interpolate the new view:

- When looking at the scene from a different viewpoint, some parts of the original scene that were occluded from the original viewpoint should become visible. This is a fundamental problem that can only be objectively solved by placing enough reference cameras so that no area of interest is left uncovered. However, methods have been described for guessing how to cover those areas in a subjectively acceptable way [6].
- Each pixel from the original image(s) maps to at most (due to occlusions) one pixel in the target image. In general the coordinates of each 3D point, when projected onto the target image space, are non-integer, so some resampling is necessary. The fastest and therefore usual solution is to map the contribution of that point to the nearest neighbour, which usually results in cracks in the image. These effects can be corrected by interpolating neighboring pixels or using splatting techniques (i.e. covering more than one pixel in the target image per 3D point). However, all of these techniques amount to low-pass filtering the image, thereby imposing a penalty on sharpness. In the end, these are patches for what is a well known problem in the image-processing domain: in order to completely cover the target domain, geometric transformations must be performed backwards from the target to the reference image(s). The idea in this context was already proposed by [7] and has also been implemented for the MPEG View Synthesis Reference Software [8, 9].
- Pixels on the edges of objects have colours that are often combinations of foreground and background. However, it makes no sense to average depth information: it must be set to a definite value, either foreground or background. Consequently, when assigning that mixed colour information either to the foreground object or to the background, some strange halo or ghosting artifacts are produced. Some proposals have been made to mitigate this effect, such as perform splatting only for the contributions coming from edge pixels [9] or layering contributions coming from foreground, background or edges and prioritising them in order not to employ edge contributions if not absolutely necessary [10, 11].

Most of the DIBR algorithms that have been presented in the literature in the last years are intended for use in 3D TV applications [12, 10, 13]. As such, their main concern is to generate views appropriate for several viewers who are sitting in front of a TV screen

This work has been partially supported by the Ministerio de Ciencia e Innovación of the Spanish Government under projects TEC2010-20412 (Enhanced 3DTV) and TEC2007-67764 (SmartVision).

and look at the scene from only slightly different angles and roughly from the same horizontal plane. Following these contour conditions, they care mainly about situations in which the cameras are disposed in a 1D array, or can be reduced to that setup using the camera calibration parameters and simple image warping procedures. In fact, this configuration has been subject of special interest by the MPEG Video Group [9, 14] and has been shown to significantly reduce computational complexity by constraining all warps and interpolations to 1D operations [5].

If both source cameras and virtual camera share a common camera plane and their baseline distance is small, displacements of the virtual camera along the baseline do not usually result in significant changes of the apparent size of objects on the image plane. In this scenario, a simple forward mapping approach such as the one previously discussed works well enough, and does not usually result in very large cracks, although it usually requires some light postprocessing of the final image to work out whatever small holes might have been left uncovered.

If the baseline distance increases too much, though, parallel cameras cannot simultaneously see objects that are located near them and they must point in different directions in order to see the same objects. In this scenario, apparent sizes of both foreground objects and background can vary significantly with the choice of the virtual camera location. Specifically, if the apparent size of any element in the images increases in the virtual view, large cracks appear in the final image that are more difficult to fix.

However, we submit that DIBR is adequate not only for 3D TV applications but also for general FVV applications, even with camera configurations that do not lend themselves to the usual 1D reduction, this is, cameras widely spaced apart with their main axes describing big angles. However, care must be exercised when treating data: in particular, we found that some datasets choose to relate all depth information to a global coordinate system instead of relating each depth video to the local reference system of the corresponding camera. We will show how this affects to large baseline setups and why we think it is not a good choice.

We wanted to explore whether it is possible to correctly reconstruct the main characters in a scene using widely spaced cameras with DIBR techniques. However, most of the available sequences are from parallel camera setups and therefore not suitable for this purpose. We selected the well-known sequences *Ballet* and *Breakdancers*, kindly provided by the Interactive Visual Media Group at Microsoft Research, that are made up of eight views per scene, each of them carrying both colour and depth information. These sequences have been used in numerous papers, but usually pairs of cameras lying next to each other are selected. Instead, we will employ the outermost cameras of both setups for our study.

On Figure 1, a forward mapping rendition without postprocessing of the scene as seen from the midpoint between cameras 0 and 7 (see [11] for details on the configuration) is shown. Since the cameras have different fields of view, there are areas at the left and right of the scene where only one of the cameras provides information, accounting for the poor(er) quality of the image in those areas. However, there are also very noticeable cracks on the main characters that we aim to solve. To do so, we have used a backward mapping approach similar to that described in [15, 8], although instead of bilateral filters we have designed a custom filtering scheme based on median filters. Figure 2 shows a summary of the process.

Fig. 1. Forward mapping of *Ballet*. Note cracks on the foreground.

Fig. 3. Depth map estimation on the virtual camera obtained from only one reference camera.

2. STUDY OF A WIDE BASELINE SETUP

2.1. Backward mapping

The key idea to backward mapping is that, if a dense depth map can be estimated for the virtual camera, every target pixel can be traced to a single 3D point that can, in turn, be projected back to one or more of the reference images to compute the final colour of the target pixel. Even though the 3D points are likely to project onto floating point coordinates in the original images, these are dense and allow for interpolation, thereby eliminating the cracks.

In order to estimate this virtual depth map, the same principles of forward mapping are followed. Each of the pixels in the reference images is unprojected to a 3D point, forming a cloud that is then projected onto the virtual view, only now we do not care about colours and we just store the Z coordinate (local to the reference system of the virtual camera if occlusions are to be solved correctly) of each visible point in the cloud. Determining visibility is very easy since we are actually implementing a Z-buffer, that is arguably the best tool for this task. Note that, since this operation is a forward mapping, this virtual depth map (see Figure 3) exhibits cracks just the same as colour forward mappings do.

However, there is a crucial difference between depth and colour forward mapping: whereas it is very difficult to tell whether a partic-

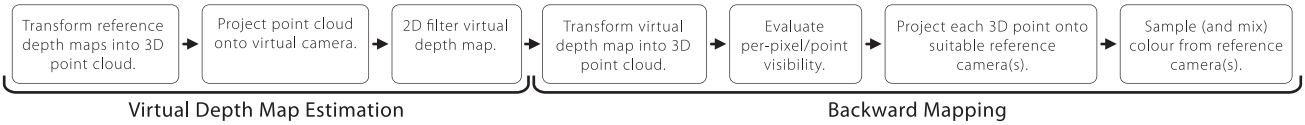

Fig. 2. Block diagram of the process pipeline.

Fig. 4. Depth map estimation on the virtual camera obtained from only one reference camera after a filtering stage.

Fig. 5. Final depth map estimation on the virtual camera obtained from both reference cameras.

ular pixel in a colour buffer is correct or not (e.g., how to automatically distinguish a crack from an actual striped texture?), this buffer actually holds a description of the physical structure of the scene; this allows us to make several useful assumptions: real-world objects are usually connected and, barring grids, nets and strings (and actual cracks in the wall!), there are not many objects that could fit the physical description that cracks in the depth image imply. Therefore, it is relatively safe to apply 2D reconstruction techniques such as median filters, which would eliminate cracks and smooth the surfaces while retaining sharpness in the actual boundaries of objects.

2.2. Filtering depth maps

In order to close the holes in the depth maps, we selected a 5×5 median filter, which is known to perform in a way which is similar to a low-pass filter in smooth areas while retaining sharp contours. However, using the assumptions discussed earlier we can improve upon the basic filter. Since reference depth data is bounded and we know the calibration parameters of the cameras, we can surely know in which range to expect legitimate depth values in the depth buffer of the virtual camera. We previously initialise the Z-buffer on the virtual camera to a huge value well beyond expected depth values. This allows us to make the following modifications to the median filter:

- Instead of using all values in the window, we compute the median of only the valid samples within the window, disregarding pixels known to contain invalid depth values.
- To prevent spurious isolated points completely surrounded by invalid points from growing, which would be an undesired consequence of the preceding modification, we require at least one quarter (experimental value) of the pixels in the window to be valid to actually modify the value of the pixel in the Z-buffer.
- To prevent valid depth values from *creeping* into areas that rightfully contain invalid values because they represent actual occlusions, we require at least half of the pixels in the perimeter of the mask to hold valid values. Thus, we can close small holes but not affect large blanks, which are probably correct.

The results of this filtering stage can be seen on Figure 4. All the cracks have all but disappeared, and the remaining holes are mainly due to occlusions, either real or due to noise in the reference depth estimation. In our particular setup, we have the contributions of two cameras as inputs, so we have to merge the depth maps resulting of their reprojections. Since we have only two, there is no objective way to tell which one is right when they hold different values for the same pixel (a possible solution to this issue has been recently proposed [16]). Thus, in such a case, we just take the lesser value, because it could represent an object which is not visible from the other camera and that, if looked at from the new virtual viewpoint, occludes whatever was.

Figure 5 shows the final depth estimation obtained from the combination of the contributions of both reference cameras as described plus another additional filtering step to fill small holes. Most of the artifacts due to the forward mapping process have disappeared, and only those areas which are not seen from any of the cameras are missing information. While the quality and coverage of the depth map could surely be improved by adding information from more cameras, it is actually good enough to reconstruct at least the main characters in the scene.

2.3. Pixel colouring

Once a dense depth map has been obtained, all that is left is tracing back each pixel to the appropriate coordinates in the original images to get its colour. However, since not all pixels in the virtual camera come from points seen by both cameras, we need to evaluate in a per-pixel basis to which camera(s) to refer to compute the final colour.

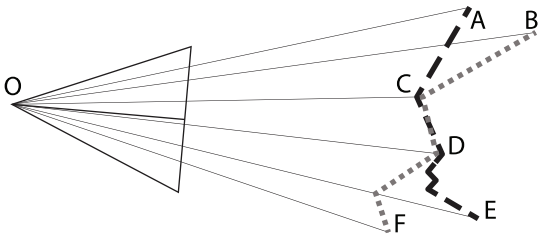

Fig. 6. Z masking

Fig. 7. Final results of backward mapping on sequence *Ballet*.

To solve these occlusion problems, we have followed a simple strategy. During the virtual depth map estimation phase, we had already produced separate partial depth maps from each reference camera such as the one shown on Figure 4, so we can use them to determine whether each pixel should be traced back to a particular camera.

Figure 6 illustrates a cross-section of the depth map estimation process; the dashed line denotes the right camera contribution and the dotted line denotes the left camera contribution. There are three possible cases:

- Angular sectors that are only seen by one camera ($\widehat{AOB}$ and $\widehat{EOF}$). We shall obviously not trace those points back to the other camera.
- Angular sectors seen by both cameras but in which the contribution of one of them is clearly closer than the other ($\widehat{BOC}$ and $\widehat{DOE}$). We disregard the camera with the farthest contribution for colour mapping because it is clearly occluded.
- Angular sectors seen by both cameras and at a comparable depth ($\widehat{COD}$). We cannot reliably rule out either of the cameras, so we will combine their contributions to compute the final colour.

The results of the whole backward mapping process can be seen on Figures 7 and 8. Excluding areas with too much noise (e.g. the floor) or that are occluded from both reference cameras, the macroscopic geometry of the scene is correct. In particular, the main characters are well reconstructed and completely defined despite the significant perspective change.

Fig. 8. Final results of backward mapping on sequence *Breakdancers*, using the same parameters as camera 4 of the original setup.

Fig. 9. Detail of foreground texture registration errors. Left to right: Wide baseline unfiltered forward mapping, wide baseline backward mapping, short baseline backward mapping.

3. REPRESENTATION OF DEPTH DATA

Although the wide camera configuration performs well for the reconstruction of the geometry of the main characters throughout the whole path between cameras, there are very noticeable registration errors between the texture contributions from the reference cameras. Figure 9 shows a detail comparing forward and backward mapping in a wide baseline configuration, and backward mapping in a short baseline configuration; both methods show similar alignment problems in the wide baseline configuration, even though forward mapping data are not filtered in any way. On the other hand, the short baseline configuration does not exhibit these symptoms, so it would appear that there is a trade-off between freedom of movement and image quality, and indeed there is if the depth reference data are not high quality or are misrepresented. In this section we will show that the errors are not due to the camera configuration but to the reference data in themselves.

The dataset used for this study provides depth images for each of the eight cameras of the setup. However, the depth data is related to the world coordinate system (which is fixed to the local coordinate system of camera 4) for all of them, which means that all pixels with the same grey level represent points on a plane $Z_{world} = k$. Being these data quantized to fit into 8-bit integers, the choice of a common reference system results in unexpected side effects.

Figure 10 shows a 2D analogue of the problem; we want to describe the circumference using both cameras and a common reference system fixed to the local system of camera A. When the camera is aligned with Z (camera A), the information that is actually being

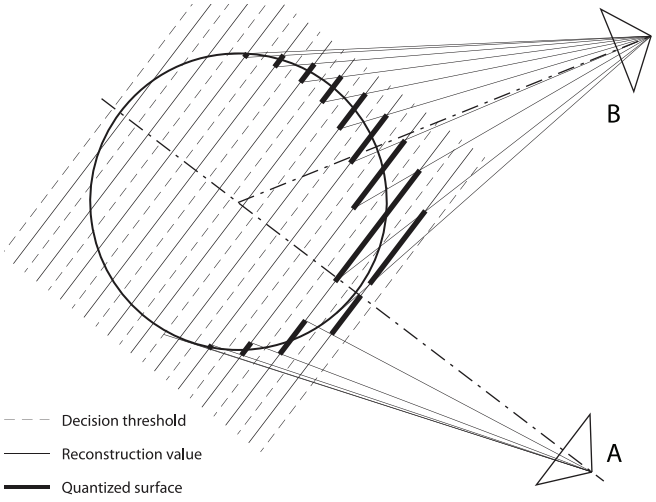

Fig. 10. Side effects of Z quantization.

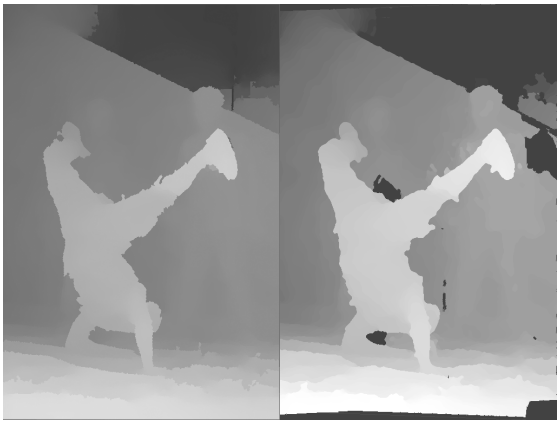

Fig. 11. Left to right: original depth estimation and virtual depth reconstruction, both for camera 4. Note that the original depth is *not* a ground truth.

encoded (only the lower half depicted) is reasonably faithful to the original object. However, when the camera is severely off-axis (camera B), what is actually being encoded significantly differs from the original surface. If data for the reconstruction comes from two cameras that lie on opposite sides of the neutral alignment, as it is the case in our study, both are biased towards opposite directions, so the reconstructed geometry is painfully wrong. Note that depth quantization in this figure is uniform for the sake of simplicity, whereas it is non-linear in the *Ballet* and *Breakdancers* datasets, but this does not affect the reasoning above.

Figure 11 shows a comparison between the original depth data estimation for camera 4—the only reference camera which provides local depth data and therefore the only one for which this comparison makes sense—of the *Breakdancers* dataset and the reconstruction of the virtual depth from the same viewpoint using cameras 0 and 7 as sources. Inconsistencies and errors in the depth estimation of reference cameras are augmented by the choice of a common reference system, resulting in the foreground object being much bigger than it should. However, it must be noted that the depth estimation of camera 4 is *not* a ground truth, so this comparison is merely illustrative.

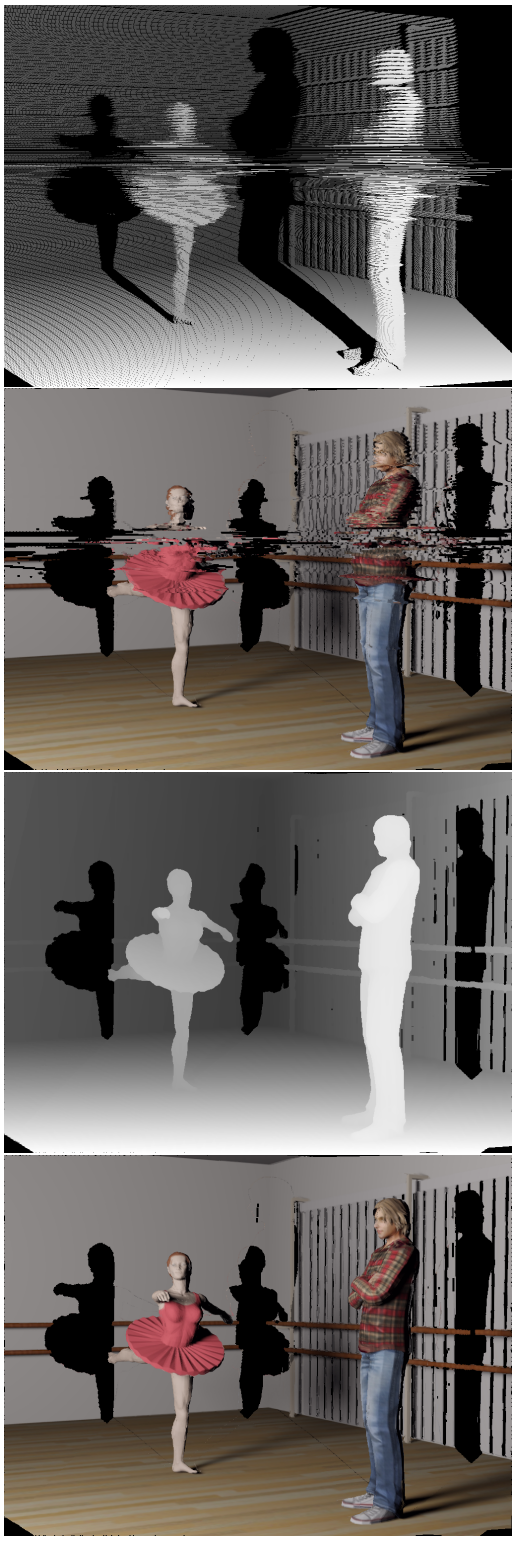

Fig. 12. Top to bottom: Unfiltered depth map estimation on the virtual camera obtained from a severely off-axis camera with Z quantization, final results of backwarp mapping for the same configuration, final depth map estimation using local reference depth maps and final results of backward mapping using local reference depth data.

These effects are not apparent when using pairs of cameras with a short baseline, even if they point at a moderate angle from the global Z axis. In such a case, quantization effects affect both cameras in a similar way and even if the geometric information they are encoding is wrong, it is consistently so. However, using closely placed cameras as sources severely hampers the freedom in "*Free* Viewpoint Video", whereas at a large scale the results from the camera configuration in our study are generally satisfactory.

Therefore, in order to test whether the failures exhibited by the extreme camera configuration are fully imputable to the effects just described or there is something intrinsically wrong about this configuration, we implemented a tool to export synthetic scenes from Blender as images plus depth, either relative to the world coordinate system or to each camera, together with the associated camera calibration parameters.

Figure 12 shows the results of reprojecting the depth of one reference camera onto a virtual camera. The global Z axis points up, and the quantization is coarser the farther from the floor. It clearly shows that, even for the finer steps of quantization, the background is visible through the main characters and, for the coarser levels of quantization, the shape that is being described is quite removed from what it should be.

Of course, when the contributions from the other camera are added into the mix, the holes between the slices are filled by whatever is visible through them, resulting in baffling results even though there is no noise, proving the point that using both global depth and depth quantization is not really a good idea. Moreover, as the angles between visual rays from the reference cameras and the global Z axis approach 90°, the codified surface becomes a complete disaster despite the fact that we had true depth data to start with, and just by using a wrong convention we managed to destroy it.

Whilst this example could seem far-fetched, if we wanted to design a system offering 360°navigation, some of the cameras would necessarily be off-axis in a similar way.

Figure 12 also shows a final render of the same scene, now using local depth maps. Even though they are still quantized to 8-bit values, they perform much better, achieving excellent reconstruction of the main characters. Therefore, our recommendation would be to always use local depth maps, which also prevents the problem that arises if the unprojection of a pixel in any image results in a ray perpendicular to the global Z direction, causing the corresponding 3D point to be undefined no matter the depth value.

4. CONCLUSIONS

We have explored a non-usual camera configuration for DIBR, looking for a way to reconstruct the main characters in a scene while allowing the user to choose a viewpoint with greater freedom. In the process, we have learned that backward mapping serves this purpose well, as long as high quality depth information is available. We have also shown that, even if this information is indeed available, it is necessary to relate it to coordinate systems local to each reference camera in order not to corrupt valuable data.

5. REFERENCES

[1] W. Matusik, C. Buehler, and L. McMillan, "Polyhedral visual hulls for real-time rendering," in *Proc. of the 12th Eurographics Workshop on Rendering Techniques*, London, UK, 2001, pp. 115–126, Springer-Verlag.

[2] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," *Computer Vision, Graphics, and Image Processing*, vol. 40, no. 1, pp. 1–29, Oct. 1987.

[3] J. Carranza, C. Theobalt, M. A. Magnor, and H. Seidel, "Free-viewpoint video of human actors," *ACM Trans. Graph.*, vol. 22, pp. 569–577, July 2003.

[4] L. McMillan, *An image-based approach to three-dimensional computer graphics*, Ph.D. thesis, University of North Carolina, 1997.

[5] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," *Proceedings of SPIE*, vol. 5291, pp. 93–104, 2004.

[6] W.R. Mark, *Post-rendering 3D image warping: Visibility, reconstruction, and performance for depth-image warping*, Ph.D. thesis, University of North Carolina, 1999.

[7] J. Shade, S. Gortler, L. He, and R. Szeliski, "Layered depth images," in *Proc. of the 25th annual conf. on Computer Graphics and interactive techniques*, New York, NY, USA, 1998, SIGGRAPH '98, pp. 231–242, ACM.

[8] Y. Mori, N. Fukushima, T. Fujii, and M. Tanimoto, "View Generation with 3D Warping Using Depth Information for FTV," May 2008.

[9] D. Tian, P. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," *Proc. of SPIE*, vol. 7443, no. 609, pp. 74430T–74430T–11, 2009.

[10] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand, "View Synthesis for Advanced 3D Video Systems," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–12, 2008.

[11] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM SIGGRAPH 2004 Papers on - SIGGRAPH '04*, vol. 1, no. 212, pp. 600, 2004.

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

[13] D. Min, D. Kim, S. Yun, and K. Sohn, "2D/3D freeview video generation for 3DTV system," *Signal Processing: Image Communication*, vol. 24, no. 1-2, pp. 31–48, Jan. 2009.

[14] P. Carballeira, F. Morán, and J. Cabrera, "View rectification for FTV Ref. SW depth estimation improvement," contribution M15281, ISO/IEC JTC1/SC29/WG11, Archamps, France, Apr. 2008.

[15] A. Hornung and L. Kobbelt, "Interactive pixel-accurate free viewpoint rendering from images with silhouette aware sampling," *Computer Graphics Forum*, vol. 28, no. 8, pp. 2090–2103, 2009.

[16] L. Yang, T. Yendo, M. P. Tehrani, T. Fujii, and M. Tanimoto, "Artifact reduction using reliability reasoning for image generation of FTV," *Journal of Visual Communication and Image Representation*, vol. 21, no. 5-6, pp. 542–560, July 2010.