INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG2018/m41990
January 2018, Gwangju, Korea

Source	**Poznań University of Technology**
Status	**Input**
Title	**Extended VSRS for 360 degree video**
Author	Marek Domański, Dominika Łosiewicz, Tomasz Grajek, Olgierd Stankiewicz, Krzysztof Wegner, Adrian Dziembowski, Dawid Mieloch

1. Introduction

Omnidirectional video formats natively provide capability of changing direction of viewing but not the position of viewing. Such capability, of changing also the position of viewing, is key for the prospective Omnidirectional 6DoF/3DoF+ video technology.

In one of the previous input documents [1] we have shown that it is possible to render additional omnidirectional view of the scene from monoscopic or stereoscopic omnidirectional video accompanied by omnidirectional depth. Up to now several custom applications for omnidirectional rendering have been demonstrated [5,6,7,8].

In this document we present extension of View Synthesis Reference Software towards omnidirectional virtual view rendering.

2. Omnidirectional virtual view synthesis

2.1. Algorithm

The goal of the omnidirectional virtual view synthesis is to create omnidirectional virtual viewpoint in any position in the space, different from position of input viewpoint, e.g. translated by vector T. The created virtual viewpoint needs to look like directly captured with 360 degree camera. View synthesis can use only as few as one input omnidirectional video accompanied with omnidirectional depth (OVD), or more. In the current version of the presented software, we provide support for up to two one input omnidirectional video accompanied with omnidirectional depth (OVD).

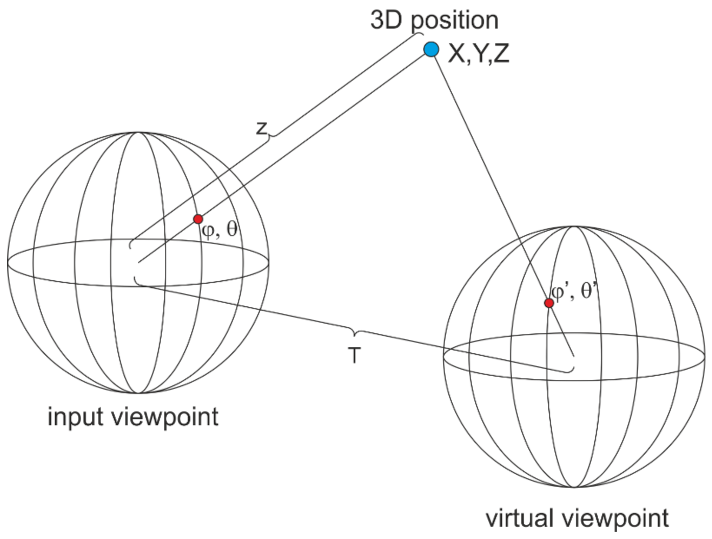

Figure 1. Omnidirectional virtual viewpoint rendering.

There are many ways to map omnidirectional video to rectangular video [2]. One of the most popular in the literature is equirectangular projection format (ERP). In this work, we assume that both input omnidirectional views and the requested virtual viewpoint use equirectangular projection.

In order to create virtual viewpoint, each pixel in each input omnidirectional view needs to be mapped to appropriate position in virtual viewpoint. At first, 3D location of each pixel has to be obtained.

For every pixel (m, n) in the rectangular input video of size (W, H) we obtain appropriate direction (φ, θ) in 3D space through (1).

$$\begin{gathered} \varphi = \left(\frac{m}{W} - 0.5\right) \cdot 2\pi \\ \theta = \left(0.5 - \frac{n}{H}\right) \cdot \pi \end{gathered} \tag{1}$$

From the depth map associated with video, we get depth value for the processed point (m, n). Given the depth map format used [4] (e.g. normalized disparity) we can easily calculate distance z of the given point from the center of the omnidirectional viewpoint.

For direction (φ, θ) and the distance z, exact 3D position the point can be calculated (2):

$$\begin{aligned} X &= z \cdot cos(\theta) \cdot cos(\varphi) \\ Y &= z \cdot sin(\theta) \\ Z &= -z \cdot cos(\theta) \cdot sin(\varphi). \end{aligned} \tag{2}$$

Next, 3D point is projected onto a virtual viewpoint sphere to create a view for requested viewpoint. Each 3D point is shifted by translation of virtual view point form input viewpoint $T = [T_x \quad T_y \quad T_z]$ as expressed in following equation (3):

$$\begin{aligned} X' &= X + T_x \\ Y' &= Y + T_y. \\ Z' &= Z + T_z \end{aligned} \tag{3}$$

For the obtained position (X', Y', Z') in virtual viewpoint coordinate system, we can calculate direction (φ', θ') relative to the virtual viewpoint center (4):

$$\varphi' = tan^{-1}\left(-\frac{Z}{X}\right)$$
$$\theta' = sin^{-1}\left(\frac{Y}{\sqrt{X^2 + Y^2 + Z^2}}\right). \qquad (4)$$

In the end, appropriate pixel position (m', n) in the output rectangular video (representing the virtual viewpoint) can be obtained from direction (φ, θ) in 3D space through equation (5):

$$m' = \left(\frac{\varphi'}{2\pi} + 0.5\right) \cdot W$$
$$n' = \left(0.5 - \frac{\theta'}{\pi}\right) \cdot H. \qquad (5)$$

2.2 Implementation in VSRS

Based on a one or two input omnidirectional video and omnidirectional depth in equirectangular projection format (ERP) new omnidirectional viewpoint is rendered. We have follow general VSRS design (Fig. 1).

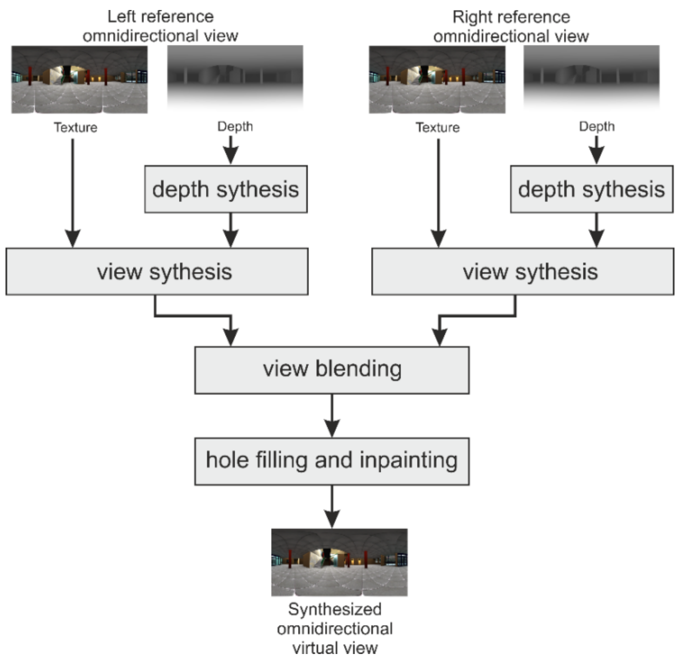
Left reference omnidirectional view
Right reference omnidirectional view
Texture
Depth
Texture
Depth
depth sythesis
depth sythesis
view sythesis
view sythesis
view blending
hole filling and inpainting
Synthesized omnidirectional virtual view

Figure 1. General overview of the algorithm.

Synthesis takes place in two separate processing paths - one per each input view. In each path, at first the depth map of a rendered omnidirectional virtual view is created in depth synthesis block. This is performed independently based of both input depth maps. Forward depth warping of each pixel from input view is used. Next, virtual depth maps are filtered by the same set of processing filters already present in VSRS. After that, in each path, texture of virtual view is created by backward warping in view synthesis block. Based on rendered omnidirectional virtual depth map a texture of a processed input view is being sampled to create a variants of virtual view texture. The process is ended by application of set of filters already present in VSRS.

Finally, the two variants of texture of virtual views are merged together in view blending stage. All remaining holes and regions that could not be rendered due to occlusion within the scene are inpainted in hole filling and inpainting stage.

Only depth synthesis and view synthesis steps have been modified in order to support omnidirectional view rendering. Rest of the processing pipeline remains unaltered. This means that all modes of view blending, and hole filling works correctly, and can be used.

2.3. Camera parameters

The camera parameters file in supported in normal format used by VSRS for single view rendering, for compatibility. Because in the proposed enhanced VSRS 360 video is generated, most of parameters related to view frustum are not required. Therefore, only camera positions are taken into consideration. In particular, the difference in camera position between real input view and virtual view is used.

2.4. Configuration parameters

The following additional configuration parameters have been added for omnidirectional rendering.

```
SynthesisMode             2         # 0...General, 1...1D parallel, 2...360 equirectangular
```

SynthesisMode

Unsigned int (0, 1 or 2), default: 0

Specifies the mode of view synthesis. 0 means the method has no restriction, 1 means the method is only works on 1D parallel multi-view sequences, 2 means the new proposed 360 degree rendering.

2.5. Exemplary configuration file

```
#================ Input Parameters ================
DepthType                 1      #0...Depth from camera, 1...Depth from the origin of 3D space
SourceWidth               2048   # Input frame width
SourceHeight              1024   # Input frame height
StartFrame                0      # Starting frame #
TotalNumberOfFrames       1      # Total number of input frames
LeftNearestDepthValue     23.344623  # Nearest depth value of left image from camera
LeftFarthestDepthValue    54.470788  # Farthest depth value of left image from camera
RightNearestDepthValue    23.175928  # Nearest depth value of right image from camera
RightFarthestDepthValue   54.077165  # Farthest depth value of right image from camera
CameraParameterFile       cam_param.txt  # Name of text file which with camera parameters
LeftCameraName            param_cam001  # Name of real left camera
VirtualCameraName         param_cma002  # Name of virtual camera
RightCameraName           param_cma001  # Name of real right camera
LeftViewImageName         PoznanHall360_Texture001.yuv  # Name of left input video
RightViewImageName        PoznanHall360_Texture001.yuv  # Name of right input video
LeftDepthMapName          PoznanHall360_Depth001.yuv    # Name of left depth map video
RightDepthMapName         PoznanHall360_Depth001.yuv    # Name of right depth map video
OutputVirtualViewImageName  PoznanHall360_Texture002.yuv  # Name of output virtual view video

ColorSpace                0             # 0...YUV, 1...RGB
Precision                 2             # 1...Integer-pel, 2...Half-pel, 4...Quater-pel
Filter                    1             # 0...(Bi)-linear, 1...(Bi)-Cubic, 2...MPEG-4 AVC

BoundaryNoiseRemoval      0             # Boundary Noise Removal: Updated By GIST

SynthesisMode             2             # 0...General, 1...1D parallel, 2...360 equirectangular

#---- General mode ------
ViewBlending              0             # 0...Blend left and right images, 1...Not Blend
```

3. Conclusions

We have implemented omnidirectional rendering as an extension of View Synthesis Reference Software.

4. Acknowledgement

This work has been supported by the public funds as a DS research project for years 2017/2018.

5. References

[1] K. Wegner, O. Stankiewicz, A. Dziembowski, D. Mieloch, M. Domański, "Omnidirectional 6-DoF/3-DoF+ rendering", ISO/IEC JTC1/SC29/WG11, MPEG2017/m40806, , Torino, Italy, July 2017.

[2] Y. Ye, E. Alshina, J. Boyce, Algorithm descriptions of projection format conversion and video quality metrics in 360Lib, Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 6th Meeting: doc. JVET F-1003, Hobart, AU, 31 March – 7 April 2017.

[3] G. Bang, G. S. Lee, N. Ho H., “Test materials for 360 3D video application discussion”, ISO/IEC JTC1/SC29/WG11 MPEG2016/M37810 February 2016, San Diego, USA.

[4] T. Senoh, K. Yamamoto, R. Oi, T. Mishina, M. Okui: Consideration of depth format. ISO/IEC JTC1/SC29/WG11 MPEG m15047, Antalya, Turkey, Jan. 2007.

[5] B. Kroon, “Adding depth maps to fulfill MPEG-I Phase 1b parallax requirement”, ISO/IEC JTC1/SC29/WG11 m41824, Macau, China, October 2017.

[6] B. Wang, Y. Sun, L. Yu, “[MPEG-I-Visual] 360 video sequence with depth map”, JTC1/SC29/WG11 m41863, Macau, China, October 2017.

[7] R. Dore, M.-L. Champel, “3DoF+ proof of concept“, JTC1/SC29/WG11 m40862, Torino, Italy, July 2017.

[8] R. Doré, I. Orlac, G. Briand, M.-L. Champel, “User tests on interactive parallax and stereoscopy” ISO/IEC JTC1/SC29/WG11 MPEGM40863, Torino, Italy, July 2017.