Multimedia Tools and Applications (2023) 82:3995–4010
https://doi.org/10.1007/s11042-022-13432-8

Omnidirectional stereo video using a hybrid representation

Xiaofei Ai[1] · Yigang Wang[2] · Xiaodiao Chen[1] · Hong Li[2]

Received: 14 February 2021 / Revised: 20 April 2022 / Accepted: 2 July 2022 /
Published online: 20 July 2022

Abstract

Compared with the traditional video, omnidirectional stereo video (ODSV) provides a larger field of view (FOV) with depth perception but makes the capturing, processing and displaying more complicated. Even though many attempts have been made to address these challenges, they leave one or more of the following problems: complicated camera rig, high latency and visible distortions. This paper presents a practical end-to-end solution based on a novel hybrid representation to solve these problems simultaneously. The proposed solution is directly from capturing to displaying, which removes the processing step, thus reducing the total time consumption and visible stitching distortions. This hybrid representation is piecewise linear about the horizontal viewing direction whose domain of definition is 0° to 360° with an assumption that the background is static, consisting of both static and moving regions. Using this representation, ODSV can be presented by omnidirectional stereo images and normal stereo pair of videos respectively. Moreover, a single panoramic camera strategy can be adopted to capture the omnidirectional stereo images in real environment and a normal binocular camera can be used to capture the stereo pair of videos. To display the ODSV, this paper presents a real-time tracking-based rendering algorithm for head mounted display (HMD). Experiments show that the proposed method is effective and cost-efficient. In contrast to state-of-the-art methods, the proposed method significantly reduces the complexity of camera rig and data amount, preserving a competitive stereo quality without visible distortions.

Keywords Omnidirectional stereo · Stereo video · Stereo panorama · Image based rendering · Head mounted display

✉ Xiaofei Ai
xiaofeiai@hdu.edu.cn

Yigang Wang
yigang.wang@hdu.edu.cn

[1] Hangzhou Dianzi University, School of Computer Science, Hangzhou, China

[2] Hangzhou Dianzi University, School of Media and Design, Hangzhou, China

1 Introduction

Recent developments in virtual reality (VR) and HMD [12] have led to an increased interest in the creation of high-quality VR content. Meanwhile, many attempts have been made to create high-quality content, which can be mainly summarized as two categories: model-based methods and image-based methods. The former is used for computer-generated objects, that is, rendering based on the knowledge of geometries, materials, lights. The latter is usually used for real-captured objects, that is, rendering novel view from a set of captured images on different viewpoints. Omnidirectional stereo [24] is one of the image-based methods, which can provide the realistic surface appearances and parallax information for high-quality VR content. However, studies over the past two decades mainly aim at high resolution and seamless stitching images in static scenes. To achieve omnidirectional stereo for dynamic scenes, much of approaches left one or more of the following problems:

- *Complicated Camera Rig* State-of-the-art ODSV capture requires multi-camera rig, which relies on the accurate and bulky designs. This is clearly beyond the reach of end users since it is expensive and inconvenient. Anderson et al. [1] use 16 GoPro cameras to construct a ring-shaped rig with a diameter of 28 cm to capture stereo panoramas for real environment. Ra et al. [23] capture panoramic stereo video from 5 GoPro cameras and a stereo camera. Xu et al. [35] generate live VR videos using 6 GoPro camera rig.
- *High Latency* Time is important for providing comfortable stereo video experience. The capturing and data processing are both time-consuming steps in the pipeline of ODSV creation. For example, existing approaches using a rotational camera usually result in a low frame rate. Approaches using multi-camera rig for video capture are limited by the stitching performance. A single panorama with the resolution of 8K takes more than 60 seconds using a dense optical flow blending algorithm [1]. 10 seconds of ODSV may occupy several gigabytes which is too large for real-time streaming.
- *Visible Distortions* Distortions are sensitive in VR since human perception of the surroundings mostly comes from vision [26]. Existing flow-based methods still have obvious stitching artifacts for thin or transparent objects. Moreover, limited by the size of physical camera, binocular parallax in an ODSV is gradually weakened from vertical direction [1], which has a significant impact on viewer's experience.

This paper presents a practical end-to-end solution to overcome the above problems simultaneously. Firstly, it explores the redundancy in traditional ODSV and simplifies its data model into a hybrid representation based on Ra's [23] idea which was successfully used for normal video creation with stereoscopy in a specific direction. An additional constraint is added to the existing representation to provide stereo perception in all directions. Secondly, a single camera and a binocular camera are used to capture the ODSV both in synthesized scenes and real environment. Finally, a real-time tracking-based rendering algorithm is proposed to display the ODSV in HMD, removing the time-consuming data processing step. The contributions of this paper can be summarized as follows:

- Removing the redundancy in traditional ODSV, the data amount is minimized.
- Using only two cameras to capture ODSV, the dependence on complicated camera rig is reduced.
- Real-time rendering with a tracking-based algorithm, ODSV is displayed without stitching distortions.

This paper has been organized in the following way. Section 2 gives a brief overview of related work, including research on omnidirectional images and videos. Section 3 describes the end-to-end solution from capturing to displaying. In detail, it elaborates the hybrid representation for ODSV followed with a capturing and displaying approach. Section 4 discusses the findings of experiments on a number of synthetic indoor scenes and real-captured scenes. Short conclusions, limitations and future work are given in Section 5.

2 Related work

This section presents a relevant overview of omnidirectional stereoscopy, including the creation of omnidirectional stereo images for stills and stereo videos. A more conceptual and theoretical overview on visual real world for VR can be found in Richardt's report [27].

2.1 Creation of omnidirectional stereo images

Omnidirectional stereo image is also called stereo panorama that can provide binocular parallax for human viewing in all directions. It is usually created from multi-view projection, which is different with the creation of omnidirectional image or monocular panorama. To achieve such a multi-view projection, many studies have focused on capturing from single rotational camera, multi-camera rig. Besides, some methods synthesize the parallax to achieve the multi-view projection and thus converting 2D to stereoscopy.

From single rotational camera To create stereo panoramas, Peleg et al. [20] presented a multi-view projection algorithm, *circular projection*. This study indicated that stereo panoramas can be created from a single rotational perspective camera or a slit camera. In the case of perspective camera, a conventional central projection was created by stitching several columns of pixels in the center of each image and stereo panoramas were extracted from small slices off the center columns. In the cases of slit camera, it just required to stitch the corresponding slits to generate a full panorama with large FOV. However, this method tended to stretch objects, resulting in distortions. Inspired by this work, Richardt et al. [25] proposed a method for high-quality stereo panoramas. It first used a compact representation to represent the input images and estimate the high-precision light space. Then, the effective input light was up-sampling based on the flow, which solved the known aliasing problems. However, it required a dense enough sampling to create a high-resolution stereo panorama.

From multi-camera rig There has been an increased research on creating stereo panoramas from calibrated multi-camera rig in recent years. In 2016, Anderson et al. [1] used 16 GoPro cameras to construct a ring-shaped rig with a diameter of 28 cm to capture stereo panoramas for real environment. An optical flow algorithm was adopted to achieve seamless stitching for those 16 input images. While it created panoramas with up to 8K pixels, the computational cost of dense optical flow was relatively high. It took more than 60 seconds for a single 8K frame processed by a distributed computing platform. Moreover, stitching distortions are obvious in some cases. In order to reduce the number of used camera and preserve the quality, Matzen et al. [18] proposed a low-cost method for creating stereo panoramas from only two 360° cameras. This rig was cheaper and more convenient than conventional multi-camera rig. However, the parallax only existed in a specific horizontal direction. Meanwhile, the processing time was about 51-243 seconds for each 2K image, depending on the complexity of image features.

From 2D to stereoscopy conversion In addition, some researchers directly synthesize the left and right views from monocular panoramas. Learning-based methods [2, 6, 10, 34] used convolutional neural networks (CNN) to generate stereo views. For example, Xie et al. [34] used stereo pair of images extracted from a large collection of 3D movies to train CNN model. Meanwhile, Flynn et al. [6] used deep networks and applied regression analysis to directly predict the color of pixels without inferring their depth. In 2019, Lu et al. [16] used the input perspective stereo images to train and generate a network, synthesizing stereo panoramas from a single 2D panorama. This method first generated locally perspective views (LPVs) from a single panorama, and then used the generative adversarial networks (GAN) architecture to synthesize left and right image pairs for these LPVs. To enable the omnidirectional stereoscopy, it generated a series of stereo pair of images, and stitched them together to construct the concentric mosaic. Finally, these images were mapped to the output stereo panoramas. The availability of large amounts of training data further improved robustness and quality. Nevertheless, most techniques only deal with perspective images, and the results produced by these techniques are often loss of detail.

2.2 Creation of omnidirectional stereo videos

In contrast to conventional stereo videos, ODSV cannot be captured by a binocular camera [5, 19] directly. A simple way is to take the ODSV as a time sequence of stereo panoramas. Therefore, methods for the creation of stereo panoramas can be reused for ODSV. In fact, much of research has focused on stitching frame-by-frame and can be summarized as the following categories:

Resolution oriented Anderson et al. [1] used 16 cameras to record 16 video clips, generating high-resolution ODSVs through camera calibration, optical flow estimation, exposure correction and blending. The blending algorithm ran on a distributed computing platform which could process several hours of footage every day. Facebook [7, 17] used a similar multi-camera rig to capture a set of videos, combining the top and bottom camera views to generate ODSVs with a full $360° \times 180°$ FOV. Schroers et al. [29] proposed an ODSV system from capture to display using light field reconstruction. Zhang et al. [37] created the ODSV with one billion pixels (Gigapixel) from an unstructured cylindrically distributed camera array. Though this kind of methods can create high resolution videos, they all required expensive and bulky camera rig. Moreover, the huge data amount introduced by these methods requires much of time to process.

Time oriented A special branch of ODSV is live ODSV which requires data to be transmitted through a limited bandwidth in real time. In a given bandwidth, it requires a faster stitching algorithm and smaller data amount. Konrad et al. [11] used a rotational camera to capture live ODSV, which removed the time-consuming stitching process as well as the stitching artifacts. They implemented a prototype camera, *Voxel*, which can capture an ODSV with a resolution of 8192×4096 and a frame rate of 5 FPS. Huang et al. [9] used 4 fisheye cameras to capture four videos and performed features matching in the overlapping areas between adjacent frames based on previous work [13]. It improved 10-12 times in speed than the previous method, while the processing time for each 2K frame reached 50 seconds. However, the low frame rate caused by rotational camera still cannot satisfy the real-time performance. Xu et al. [35] used three pairs of stereo cameras to capture the stereo videos and tried to transmit live ODSV. This method dynamically switched seams to avoid stitching artifacts. Under the condition of 4Mbs bandwidth, the frame rate of videos with a resolution of 2048 × 2048 reached to 30 FPS.

Efficiency oriented Limonov et al. [15] captured videos with 17 fisheye lenses and used depth proxy stitching with a two-step calibration algorithm to estimate accurate depth. The stitching and estimation algorithm were optimized in parallel. Processing 4K videos on a workstation reached a frame rate of 30 FPS. Tang et al. [31] proposed a multi-scale optical flow algorithm that enabled ODSV to be displayed with low-latency and restored more accurate 3D information. This method only produced a delay of 2.2 seconds for real-time transmission of a 4K video. In addition, many researches made the high quality and real-time performance possible by simplifying data mode. Ra et al. [23] provided a new tool for the creation of ODSV. They believed that viewers only look at a specific part of the panoramic video at a given time. Therefore, only this area needs to be tracked and should be provided with parallax. Based on this idea, they proposed a method combining the stereo and no-stereo area to capture ODSVs efficiently. However, the stereoscopy produced by this method is only in front of the viewers.

Overall, a large number of techniques have been developed for omnidirectional stereo images and videos. On the one hand, some methods use complicated camera rig for high resolution, resulting in a huge data amount and increased processing time. On the other hand, by optimizing the creation pipeline and stitching algorithm, some methods still result in a low frame rate and huge data amount. Existing methods still leave one or more problems of complicated camera rig, high latency and visible distortions.

3 Our proposed method

To overcome these problems, we first find the redundancy in conventional ODSV and simplify its data model into a hybrid representation. This hybrid representation allows that the ODSV can be created from stereo panoramas and videos. Then, a single panoramic camera strategy is adopted to capture the stereo panoramas and a normal binocular camera is used to capture the stereo pair of videos in real environment. The captured data is fed into our hybrid representation. Finally, we present a tracking-based algorithm for real-time displaying the ODSV in HMD. Different from existing methods, the displaying of ODSV is directly from the captured data and the data processing step is removed, as shown in Fig. 1.

3.1 Representation

The conventional ODSV can be regarded as a time sequence of omnidirectional stereo images or stereo panoramas. Each stereo panorama records the light rays started from all directions to the multi-view projection center at the time t_i. Obviously, there is redundancy

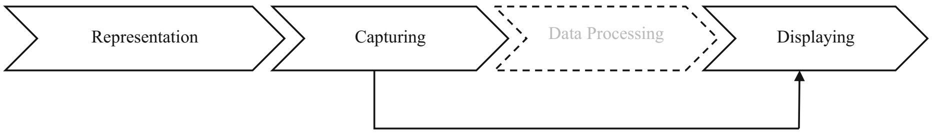


Fig. 1 Our pipeline from representation to displaying the ODSV, which removes the time-consuming data processing step

inherently in conventional ODSV because it is unnecessary to record the entire light information for static surroundings in each frame of video. In fact, to simplify conventional ODSV, we could make two assumptions (Asm.) as follows:

Asm. 1 Moving objects in an ODSV are all located in a small FOV.
Asm. 2 The appearance of static objects does not change over time.

For the first assumption, it is reasonable to accept that in most of case, moving objects are located on the regions of interest (ROI) which requires only a small FOV. For the second assumption, it allows the static objects to be regarded as the background. Based on these two assumptions, an ODSV can be segmented into two independent parts: the ROI and background. Therefore, the conventional ODSV is simplified into a hybrid representation. In this paper, the combination of stereo panoramas and normal stereo videos is used to achieve this hybrid representation.

Formally, a conventional ODSV is noted as a 4-dimensional (4D) function $\Gamma(\alpha, \beta, \gamma, \tau)$, in that α is the viewpoint in a circular trajectory, β, γ is the light direction located on a spherical surface and τ indicates the time of video. In details, the circular trajectory is defined by $\zeta : x^2 + y^2 = r^2$, where r is usually the half of human interpupillary distance (IPD). The spherical surface is defined by $\Omega : X^2 + Y^2 + Z^2 = R^2$, where R is the radius of image projection. In particular, the traditional omnidirectional video using this definition can be written as $\Gamma_c(\beta, \gamma, \tau)$, in that α is a single projection center instead of a circular trajectory.

While the 4D function $\Gamma(\alpha, \beta, \gamma, \tau)$ represents the light rays started from all directions Ω to all the viewpoints on ζ, the produced huge data amount is unbearable for modern hardware. Fortunately, such data is redundant based on the above assumptions. Considering that the static and moving objects are in two independent regions: Ω_s and $\tilde{\Omega}_s$ corresponded the viewpoints ζ_s and $\tilde{\zeta}_s$, an ODSV can be simplified into a piecewise function written as (1), in that K is the number of viewpoints, L is the video length, c_1 and c_2 are constants, M and N is the subdivisions in β and γ dimensions respectively.

$$\begin{aligned}\Gamma(\alpha, \beta, \gamma, \tau) &= \sum_{i=1}^{K}\sum_{j=1}^{M}\sum_{p=1}^{N}\sum_{q=1}^{L} x_{(i,j,p,q)}(\alpha, \beta, \gamma, \tau) \\ &= \begin{cases} \Gamma(\alpha, \beta, \gamma, c_1) & \alpha \in \zeta_s, (\beta, \gamma) \in \Omega_s, \tau = c_1 \\ \Gamma(c_2, \beta, \gamma, \tau) & \alpha = c_2, (\beta, \gamma) \in \tilde{\Omega}_s \end{cases}\end{aligned} \tag{1}$$

In the case of $\alpha \in \zeta_s$, $(\beta, \gamma) \in \Omega_s$,

$$\Gamma(\alpha, \beta, \gamma, \tau) = \sum_{i=1}^{K_s}\sum_{j=1}^{M_s}\sum_{p=1}^{N_s} x_{(i,j,p)}(\alpha, \beta, \gamma, c_1) \tag{2}$$

In the case of $(\beta, \gamma) \in \tilde{\Omega}_s$,

$$\Gamma(\alpha, \beta, \gamma, \tau) = \sum_{j=1}^{\tilde{M}_s}\sum_{p=1}^{\tilde{N}_s}\sum_{q=1}^{L} x_{(j,p,q)}(c_2, \beta, \gamma, \tau) \tag{3}$$

Since $\Omega_s \gg \tilde{\Omega}_s$ and $\zeta_s \gg \tilde{\zeta}_s$ under the **Asm.1** and **Asm.2**, the 4D function is simplified as a sum of two 3D functions. This representation allows us to create ODSV from sparse data, which significantly reduces the produced data amount. In particular, we recommend an empirical value of $K_s = 4$ and $\tilde{K}_s = 2$ which is suitable for the most of cases. In this configuration, the ratio of data amount between the conventional ODSV and the proposed

ODSV may reach 83.4:1 according to (8) under the conditions of $p_i = p_j = 1$ byte, $t_i = 1$ second, $f_i = 60$ FPS, $w_i = 2048$ pixels, $h_i = 1024$ pixels, $w_j = 512$ pixels, $h_j = 256$ pixels.

3.2 Capturing

The data capture step is divided into two independent parts according to our hybrid representation. One is the capture for omnidirectional images and the other is the capture for normal stereo videos. Both the captures can be implemented in synthesized scenes and in real environments.

3.2.1 Panoramas from a single camera

Equation (2) indicates the light rays from each direction Ω_s to the view space ζ_s. It represents the background in our ODSV. We use four panoramas formatted in Equirectangular Projection (ERP) to reconstruct (2). By using a single omnidirectional camera to capture the static background from four locations, it allows that the ODSV can be captured from a single camera, reducing the dependence on complicated camera rig. To get a uniform sampling in synthesized scenes, the image space (u, v) for each ERP is changed to spherical coordinate (θ, φ, R) and finally converted to Euler space (x, y, z). This can be written as (4).

$$F_i''(u, v) = F_i'(\theta, \varphi, R) = F_i(x, y, z), \tag{4}$$

where the first transformation is achieved by

$$\begin{bmatrix} \theta \\ \varphi \end{bmatrix} = \begin{bmatrix} 2\pi/w & 0 \\ 0 & \pi/h \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{5}$$

in that w, h is the width and height of the ERP respectively. And the second transformation is achieved by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \\ \cos\varphi \end{bmatrix} \tag{6}$$

Therefore, the color value for each pixel (u, v) represents the light from (x, y, z) to the viewpoints v_i in ζ_s.

Without loss of generality, the view space ζ_s is subdivided into four parts, $O_i, i \in \{0, 1, 2, 3\}$, corresponding to four main directions as shown in Fig. 2(a). Unlike traditional omnidirectional stereo images, four panoramas instead of two are used to record the binocular parallax, which indicates more clues can be used to reconstruct parallax in all directions. The camera positions of O_i are first computed using the definition of ζ. Then we capture a monocular panorama for each O_i using (4). This can be achieved by placing the camera to each position in turn or rotating a single camera around an axis behind the optical center. Both of these two cases, the distance $\|O_0O_2\|$ and $\|O_1O_3\|$ is equal to the IPD. That is, the distance between the camera center and axis is half of the IPD. Under this configuration, we can capture four monocular panoramas to construct the background of ODSV. One of benefits is that the binocular parallax can be preserved well, which ensures the correct stereo perception in later displaying.

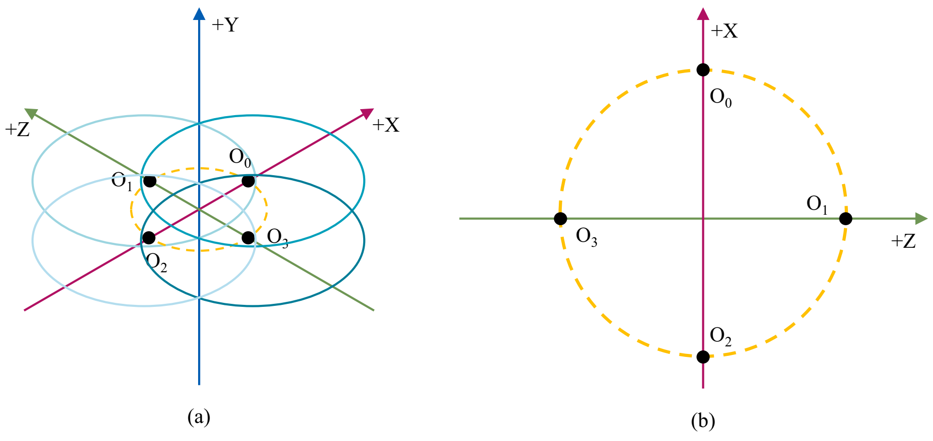


Fig. 2 (a) Four panoramas are captured in turn by placing a 360° camera on $O_i, i \in \{0, 1, 2, 3\}$ and (b) the top view of our capture geometry

3.2.2 Stereo videos using green screen

Equation (3) indicates the light rays from each direction $\tilde{\Omega}_s$ to the view space $\tilde{\zeta}_s$. In practice, we use normal stereo videos to reconstruct this function. Existing methods have utilized a binocular camera to capture the standard stereo videos. However, our captured videos have to be blended seamlessly with the background of the ODSV. It requires that the videos used to represent the ROI can be segmented. For simplicity, stereo videos with a blend channel are used to reconstruct (3). More specifically, stereo videos are captured by placing a green screen behind the moving objects. One of benefits is that we can reuse various green screen matting algorithms for later displaying. To relax the restriction on green screen, more general matting algorithms could be adopted, such as Sengupta's method [30]. However, these algorithms usually take more time to segment the videos.

3.3 Displaying

After the capture step, we get enough data for the hybrid representation. The final step of our end-to-end solution is to display the ODSV. We display the stereo views in HMD using a tracking-based rendering algorithm. The displaying is directly from the captured data, including three steps: geometry estimation, tracking and blending. The algorithm is implemented in a native browser powered by the webGL technique as well as webXR standard [32]. To viewing our ODSV, widely off-the-shelf HMDs such as *Google Cardboard* [8], *Oculus Quest* [4] and *Pico* [22] are available choices.

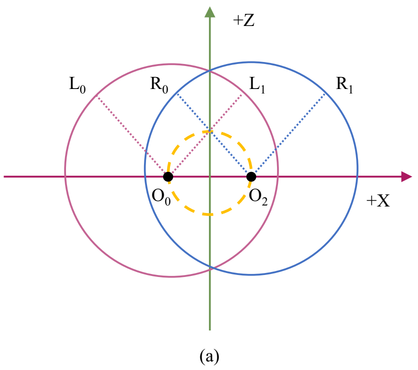
Step 1 Geometry estimation from the input panoramas and videos. We use a spherical geometry to approximate the background and planar geometries to approximate the ROI. The radius of the spherical geometry and the size of planar geometries are specified by manually according to the relative size of the background and foreground of the captured scenes. By this step, it guarantees of spatial consistency in the capturing and rendering so that the parallax can be reconstructed correctly.

Step 2 Tracking the viewing direction using the Inertial Measurement Unit (IMU). It requires a conversion from Euler angles to the local rendering space. We assume that viewer's left and right eyes keep aligned in horizontal, the rotation around the z-axis in Euler angles can be negligible. All we need to do is to convert the x and y elements of Euler angles to the space of $\theta \in [0, 2\pi)$. This can be achieved by the following piecewise linear function in (7). Through this conversion, it ensures the continuity when viewer looks around. We track the viewing direction all the time for novel stereo views rendering.

$$\theta = \begin{cases} \pi - y & \cos x \geq 0 \\ y & \cos x < 0, \sin y \geq 0 \\ 2\pi + y & \cos x < 0, \sin y < 0 \end{cases} \tag{7}$$

Step 3 Blending the input panoramas with videos. We first look up the corresponded panoramas and videos according the viewer's direction. Then, panoramas are directly projected on the spherical geometry using texture mapping. Meanwhile, green screen matting is computed in shader and videos are mapped to the corresponded planar geometry. The matting algorithm is implemented in shader program which can provide real-time performance.

Using this tracking-based algorithm, the rendered stereo views are free from the stitching artifacts. This is because our rendering is directly from the captured data and removes the stitching processing. The schematic diagram of our rendering algorithm can be seen in Fig. 3. The dash inner circle indicates where the viewpoint is located. The solid outer circles are projection surface for the left and right eye respectively. In detail, Fig. 3(a) is the rendering for the background, that is the stereo panorama when the viewer is looking forward to $y+$ axis. In this case, the left view is from O_0 and the right view is from O_2. When viewers change their viewing direction, the views are rendered by dynamic sampling the captured panoramas according to (7). Concretely, when the viewer's head is rotated to an angle of 0°, 90°, 180° or 270° that the selected panoramas will give exactly correct results as the eye positions correspond to those of the captured panoramas. As the head rotation angle moves towards 45°, 135°, 225° or 315° that there will be an increasing difference between the rendered view and the correct view, with the error depending on the difference between the assumed scene depth and the actual depth. As the viewer's head direction increases beyond one of these figures, then different panoramas will be selected with the potential for this



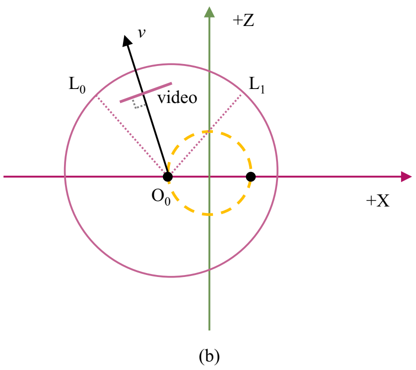


Fig. 3 The schematic diagram of our rendering. (a) Rendering the background when viewer is looking forward to $y+$ axis. (b) Rendering the left-eye view of ROI when the viewing direction is v

switch to be visible. When the moving region enter into the viewer's eyes, the video is rendered on the estimated geometry, *video plane*, using video mapping technique as shown in Fig. 3(b). In this figure, we only describe the rendering of video for left eye in the direction of v. In this case, the *video plane* is perpendicular to the viewer's head direction v, so that videos are free of distortions. Similarly, the right view is rendered from the right channel of stereo video.

4 Experiments and results

4.1 Capturing

We first test our capture method in several virtual scenes, including the *Bedroom*, *Bathroom*, *Living room*, *woman*, *pikachu*, *man*. A virtual 360° camera is placed according to Fig. 2(b) to render panoramas using physically based rendering toolkit (PBRT) [14, 21, 28, 36] and a virtual binocular camera is used to record stereo videos in *3ds Max*. Parts of our synthesized data are shown in Fig. 4. The resolution is 2048 × 1024 for both panoramas and videos. The lengths of videos are about 3–10 seconds. The length of virtual camera's baseline is set to the human IPD, 64 mm, so as to preserve the binocular parallax for human viewing. These stereo videos are captured in real time because the use of a simple shading model. On the contrary, panoramas are rendered using the PBRT which is an off-line rendering algorithm. On Intel Xeon(R) CPU E5-1650 v4 @ 3.60 GHz, the time costs for capturing a panorama of the *Bedroom*, *Living room*, *Bathroom* are 17747, 12633, 36579 seconds respectively. Generally, the capture speed can be improved by a better CPU or a parallel implementation on GPU.

Then we use Insta 360 EVO to test our capture method in real environment. Based on our capturing method, the camera is placed on the four positions to capture four panoramas and placed in front of a green screen to capture stereo videos by turning on the EVO's stereo mode. The resolution is 6080 × 3040 for panoramas and 5760 × 2880 for stereo videos. The length of each video is one minute and four panoramas can be captured within 30 seconds. Parts of the captured data (*Exhibition*, *Exhibition hall*, *Laboratory* and *Man 2*) are shown in Fig. 4 with the tag of (Real).

4.2 Displaying

Next, we test our displaying method using both synthesized data and real-captured data. All of panoramas are formatted in ERP and videos are encoded in H.264 using FFmpeg [3].

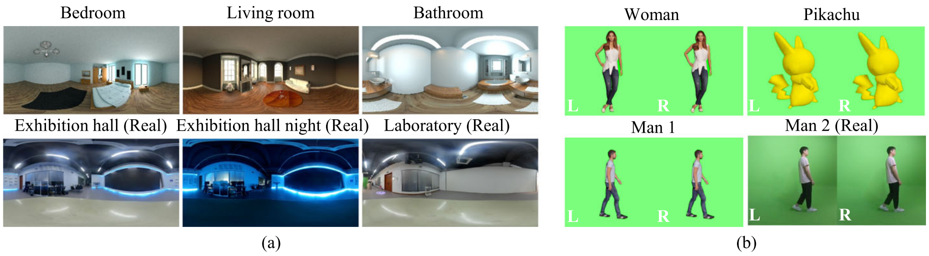


Fig. 4 Parts of the generated (a) panoramas and (b) stereo video frames in synthesized scenes and real environment

The rendering algorithm is implemented by JavaScript and OpenGL Shading Language. In this paper, the rendering program runs on the native browser in *Oculus Quest 2*. Since the stereo view is rendered on the fly, the parallax in all directions cannot be shown in a single image. Considering that four panoramas are used to represent the omni-directional stereo, the horizontal viewing direction $(0°, 360°)$ can be divided into four sub-regions (i.e. $(0°, 90°)$, $(90°, 180°)$, $(180°, 270°)$ and $(270°, 360°)$). Without losing generality, one of these regions (i.e. $(0°, 90°)$) is selected for stereoscopic perception evaluation, and the stereo quality of other regions is the same as that of the selected region. We take small perspective views from three of the playing ODSVs (*Bedroom*, *Living room* and *Laboratory*) in the selected region at five different horizontal viewing directions (i.e. 0°, 30°, 45°, 60° and 90°) and use the red-cyan style [33] for stereoscopy visualization, as shown in Fig. 5. The viewing directions are presented by an order of pitch, yaw and roll (e.g. the yaw angle refers to the horizontal viewing direction). The upward (i.e. $[90°, 0°, 0°]$) views are also evaluated because a full FOV is crucial for exploring panoramic scenes.

As can be seen in Fig. 5, our method can provide good stereoscopic perception in the viewing directions of $[0°, 0°, 0°]$, $[0°, 90°, 0°]$, proper stereoscopic perception in the viewing directions of $[0°, 30°, 0°]$, $[0°, 60°, 0°]$ and poor stereoscopic perception in the viewing directions of $[0°, 45°, 0°]$. On the one hand, it is easy to perceive the stereoscopy in the directions of $[0°, 0°, 0°]$, $[0°, 90°, 0°]$ because the binocular parallax in these two directions has been sampled before. The stereoscopy is preserved for the content on the background of our ODSVs as well as the moving content, such as the *window* in *Bedroom* and *woman* in *Living room*. Moreover, the upward views can also provide good stereoscopic perception, as shown in the rightmost column of Fig. 5. On the other hand, we have sampled the binocular parallax only in four directions and try to reproduce it in all directions using the nearest rendering algorithm, indicating that inevitable parallax errors would be produced. With the increase of the deviation between the viewing direction and the sampled directions, the parallax error gradually increases, which will lead to the wrong stereoscopic perception. However, these errors are small enough to present stereoscopic perception in the views of Fig. 5. Our method could partially render stereoscopic perception for those unsampled

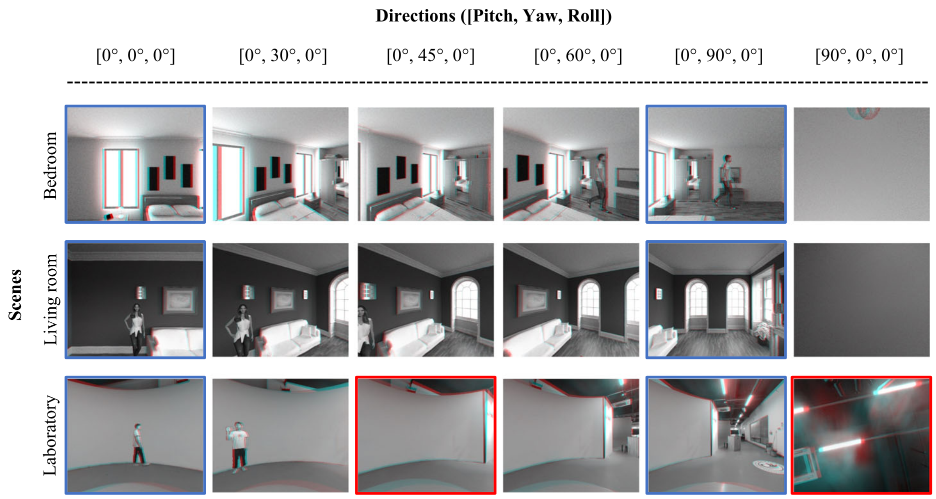


Fig. 5 Small views taken from our playing ODSVs. Good and poor stereoscopic views are marked by blue and red border respectively. Red-cyan glasses are recommended to view the stereoscopy correctly

directions (i.e. [0°, 30°, 0°], [0°, 60°, 0°] and [0°, 45°, 0°]) where they lie between captured directions, such as [0°, 30°, 0°] in *Bedroom*. But for the third column of *Laboratory*, it shows poor stereoscopic perception. This is because the stereoscopic perception is not only affected by the binocular parallax, but also related to the depth of scene objects. It is for this reason that only one view (picked by red borderlines) is with poor stereoscopic perception in the direction of [0°, 45°, 0°] among the three scenes.

Moreover, for the upward view in *Laboratory*, the provided stereoscopy is relatively poor. This is due to the influence of the lamp on the lens, which can be avoided by improving the capture conditions. In the synthesized scenes, the upward view can provide good stereoscopy, such as the lamp in *bedroom*. In fact, there is usually no specific content to care about in the upward view, such as the upward view in *Living room*, indicating a complete FOV is enough for the displaying in HMDs.

4.3 Comparative evaluation

Finally, we compare our method to the most closely-related methods [1, 18, 23, 35]. The number of cameras used for capturing, camera cost, stereo FOV in horizontal (H-SFOV) and vertical direction (V-SFOV) in each method are reported in Table 1. As can be seen from Table 1, our method only requires a single 360° camera and a binocular camera during the capturing, which is less than other methods, indicating that our method is low-cost and independent from complicated camera rig. The 360° camera used in our experiments is Insta 360 EVO which also provides the ability like a binocular camera. Although the camera cost may be a little higher than 540.61$ while using an additional binocular camera, our method is cheapest and can be affordable for common users. Moreover, our method provides the stereo perception in a full FOV (360° × 180°) which is largest among these methods.

We evaluate the rendering quality in the spirit of virtual re-photography on our synthesized scene (i.e. *Living room*) according to the camera configurations of each method (JUMP, DHPS, GVLV and LCSP). Because there are barely open-source codes for these methods and we do not have access to their software, we use the classical circular projection [20] instead of their whole pipeline to synthesize each frame of ODSV. The comparisons are all under the same conditions where frame resolution is 2048 × 1024 and FOV is 360°, and we compare small rendered views at the same camera location (viewer's position) (0, 1.6, 0) in four viewing directions (i.e. [0°, 0°, 0°], [0°, 30°, 0°], [0°, 60°, 0°] and [0°, 90°, 0°]). More specifically, we render left-eye views with a resolution of 512 × 512 for each viewing direction, compare the generated views to the ground truth rendered by PBRT using peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM), and report the PSNR and SSIM in Fig. 6(a) and (b) respectively.

Table 1 Basic configurations in the closely-related methods (top) and our method (bottom)

Methods	Cameras	Cost($)	H-SFOV	V-SFOV
JUMP [1]	120° × 16	3999.20	360°	120°
DHPS [23]	160° × 7	2204.93	60°	160°
GVLV [35]	120° × 6	999.80	360°	120°
LCSP [18]	360° × 2	578.00	180°	180°
Our method	360° × 1 120° × 2	540.61	360°	180°

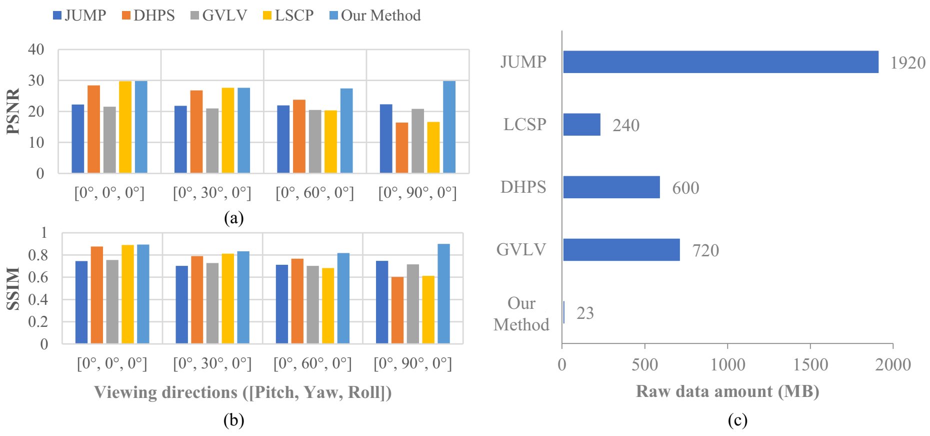


Fig. 6 Comparisons of (a) PSNR, (b) SSIM of rendered views and (c) raw data amount of different camera configurations in each method

As can be seen from Fig. 6(a), the left-eye views rendered by our method gets higher PSNR, especially when the viewing direction is [0°, 90°, 0°], the PSNR is much higher than that of other methods. In the viewing direction [0°, 0°, 0°], our method is basically similar to LSCP's camera configuration with circular projection, because we both use binocular camera to record the parallax in this direction and the noise produced by PBRT makes the PSNR slightly different from each other. With the increase of viewing direction, the difference between the left-eye view generated by LSCP's camera configuration with circular projection and the ground truth becomes larger and larger. This is because LSCP tries to use two panoramas to reproduce the omni-directional stereo vision, which brings large parallax error. In contrast, our method gets the best PSNR in the direction of [0°, 0°, 0°] and [0°, 90°, 0°] which are both the sampling direction. The score of our method in the direction of [0°, 30°, 0°] and [0°, 60°, 0°] has decreased, but it is still higher than that of other methods. This is because we use the IPD to record the binocular parallax, while we use circular projection for other methods to reproduce the parallax. Although the circular projection can provide stereo vision, it is quite different from the ground truth of views. In addition, the PSNR generated by all methods are below 30. This is because of the noise (salt and pepper noise) in the PBRT rendered images, which makes the overall image quality not higher than 30. The above results should also be confirmed by the SSIM score in Fig. 6(b). Because SSIM is usually used as an objective evaluation index closer to human vision, it indicates that our method can provide high-quality stereo vision in these evaluated directions. Because we divide the whole horizontal viewing domain into four regions and select one region for this evaluation, the parallax distribution of the other three regions is the same as that of the evaluated region, i.e. the result can be extended to the whole horizontal viewing domain.

To further illustrate the data amount reduced by our method, we count the uncompressed data amount for each method according to (8).

$$\mathrm{D_{ODSV}} = \mathrm{D_{Videos}} + \mathrm{D_{Panoramas}} = \sum_{i=1}^{m} p_i \times w_i \times h_i \times t_i \times f_i + \sum_{j=1}^{n} p_j \times w_j \times h_j, \quad (8)$$

in that p_i and p_j represent the data amount for a single pixel, w_i, w_j, h_i, h_j refer to video or image's width and height, t is video length, f is video's frame rate, m, n refer to the number of videos and panoramas respectively. Our evaluation is performed under the same conditions including $p_i = p_j = 1$ byte, $t_i = 1$ second, $f_i = 60$ FPS, $w_i = 2048$ pixels, $h_i = 1024$ pixels, $w_j = 512$ pixels, $h_j = 256$ pixels.

As can be seen from Fig. 6(c), our method takes about 23MB for one second uncompressed ODSV with a resolution of 2048×1024, which is far less than the closely-related methods. This is because the static regions are presented by four panoramas using the hybrid representation, only those moving regions need to be recorded frame by frame.

4.4 Limitations

While our method provides high-quality stereoscopy in four main directions, it introduces the jittering artifact that appears as a minor regular movement when the viewing direction crosses $\pm 45°$, $\pm 135°$ where there is a switch between the panoramas used for rendering. This phenomenon is negligible because it only appears in the background where the objects are relatively far away from the viewer. It can be avoided by increasing the sampling rate of viewpoints. This trade-off between quality and quantity is left to future work.

Moreover, the reliance on two assumptions in our ODSV representation may cause failures when one of them is untenable such as the dramatic changes in lighting. Although stereo videos with alpha channel can be generated by various matting algorithms, our system requires a green screen for capturing in real environment. It would be a fruitful area for future work to relax these constraints and preserve the competitive quality.

5 Conclusions

To overcome the complicated camera rig, high latency and visible distortions in traditional ODSV, this paper has presented a practical end-to-end solution based on a hybrid representation consisting of both static and moving regions. A 360° camera and a binocular camera have been used for data capture and a tracking-based algorithm has been proposed to render stereo views directly from the captured data, removing the time-consuming processing step. The proposed method has been evaluated on both synthesized scenes and real environment. Experiments show that our method is effective and cost-efficient. The complicated camera rig can be simplified to a panoramic camera and a binocular camera. The data amount using our hybrid representation can be reduced significantly. Our ODSVs can be viewed in a wide of HMDs and the novel stereo views are well reconstructed, covering a full FOV. Moreover, the total time for data capture in real environment is reduced. An implication of this is the possibility of real-time ODSV transmission. Our method should be applicable to many situations such as virtual education and live-VR. As higher-quality cameras and HMDs become available, we expect to see an increased desire for high-quality VR content generation. Our method would be a useful solution for capturing and displaying stereo VR content in the future.

Declarations

Conflict of Interests All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Anderson R, Gallup D, Barron JT, Kontkanen J, Snavely N, Hernandez C, Agarwal S, Seitz SM (2016) Jump: virtual reality video. ACM Trans Graph 198:1–13. https://doi.org/10.1145/2980179.2980257
2. Appia K, Batur U (2014) Fully automatic 2D to 3D conversion with aid of High-Level image features. In: Proceedings of SPIE 9011, stereoscopic displays and applications XXV, 90110W. https://doi.org/10.1117/12.2040907
3. FFmpeg Developers (2021) ffmpeg tool (version be1d324)[software]. http://ffmpeg.org/. Accessed 18 May 2021
4. Facebook (2021) Introducing oculus quest 2, the next generation of all-in-one VR. https://www.oculus.com/blog/introducing-oculus-quest-2-the-next-generation-of-all-in-one-vr-gaming/. Accessed 18 May 2021
5. Fan CL, Lo WC, Pai YT, Hsu CH (2019) A survey on 360° video streaming: acquisition, transmission, and display. ACM Comput Surv 52(4):1–36. https://doi.org/10.1145/3329119
6. Flynn J, Neulander I, Philbin J, Snavely N (2016) Deep stereo: learning to predict new views from the world's imagery. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), vol 595, pp 5515–5524. https://doi.org/10.1109/CVPR.2016.595
7. Forrest B (2021) Surround360 is now open source. https://engineering.fb.com/2016/07/26/video-engineering/surround-360-is-now-open-source/. Accessed 18 May 2021
8. Google VR (2021) Experience virtual reality in a simple, fun, and affordable way. https://arvr.google.com/cardboard/. Accessed 18 May 2021
9. Huang SK, Lin HS, Ouhyoung M (2017) Effective omnistereo panorama video generation by deformable spheres. ACM SIGGRAPH 2017 Posters (SIGGRAPH '17) 22:1–2. https://doi.org/10.1145/3102163.3102199
10. Konrad J, Wang M, Ishwar P, Wu C, Mukherjee D (2013) Learning-based, automatic 2D-to-3D image and video conversion. IEEE Trans Image Process 22(9):3485–3496. https://doi.org/10.1109/TIP.2013.2270375
11. Konrad R, Dansereau DG, Masood A, Wetzstein G (2017) SpinVR: towards live-streaming 3D virtual reality video. ACM Trans Graph 36(6):1–12. https://doi.org/10.1145/3130800.3130836
12. Koulieris GA, Akşit K, Stengel M, Mantiuk RK, Mania K, Richardt C (2019) Near-Eye display and tracking technologies for virtual and augmented reality. Comput Graph Forum 38:493–519. https://doi.org/10.1111/cgf.13654
13. Lee J, Kim B, Kim K, Kim Y, Noh J (2016) Rich360: optimized spherical representation from structured panoramic camera arrays. Int Conf Comput Graph Interact Tech 35(4):1–11. https://doi.org/10.1145/2897824.2925983
14. Lessig C, Desbrun M, Fiume E (2014) A constructive theory of sampling for image synthesis using reproducing kernel bases. ACM Trans Graph 33(7):1–14. https://doi.org/10.1145/2601097.2601149
15. Limonov A, Yu X, Juan L, Lei C, Jian Y (2018) Stereoscopic realtime 360-degree video stitching. In: 2018 IEEE international conference on consumer electronics (ICCE), pp 1–6. https://doi.org/10.1109/ICCE.2018.8326105
16. Lu J, Yang Y, Liu RY, Kang SB, Yu JY (2019) 2D-to-stereo panorama conversion using gan and concentric mosaics. IEEE Access 7:23187–23196. https://doi.org/10.1109/ACCESS.2019.2899221
17. Mashraki A (2021) Surround360. https://github.com/facebook/Surround360. Accessed 18 May 2021
18. Matzen K, Cohen MF, Evans B, Kopf J, Szeliski R (2017) Low-Cost 360 Stereo photography and video capture. ACM Trans Graph 36(4):1–12. https://doi.org/10.1145/3072959.3073645
19. Ochi D, Kunita Y, Kameda A, Kojima A, Iwaki S (2015) Live streaming system for omnidirectional video. In: 2015 IEEE virtual reality (VR), pp 349–350. https://doi.org/10.1109/VR.2015.7223439
20. Peleg S, Ben-ezra M (1999) Stereo panorama with a single camera. Proceedings 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149) 1:395–401. https://doi.org/10.1109/CVPR.1999.786969
21. Pharr M, Jakob W, Humphreys G (2016) Physically based rendering: from theory to implementation, 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco
22. Pico (2021) Pico Enterprise solutions. https://www.pico-interactive.com/us/. Accessed 18 May 2021
23. Ra KK, Clark JJ (2019) Decoupled hybrid 360° panoramic stereo video. In: 2019 International conference on 3D vision (3DV), pp 386–394. https://doi.org/10.1109/3DV.2019.00050
24. Richardt C (2020) Omnidirectional stereo. In: Ikeuchi K (ed) Computer vision. Springer, Cham. https://doi.org/10.1007/978-3-030-03243-2_808-1

25. Richardt C, Pritch Y, Zimmer H, Sorkine-Hornung A (2013) Megastereo: constructing high-resolution stereo panoramas. 2013 IEEE Conference on Computer Vision and Pattern Recognition 1:1256–1263. https://doi.org/10.1109/CVPR.2013.166
26. Richardt C, Hedman P, Overbeck RS, Cabral B, Konrad B, Sullivan S (2019) Capture4VR: from VR photography to VR video. In: SIGGRAPH courses, pp 1–319. https://doi.org/10.1145/3305366.3328028
27. Richardt C, Tompkin J, Wetzstein G (2020) Capture, reconstruction, and representation of the visual real world for virtual reality. In: Magnor M, Sorkine-Hornung A (eds) Real VR – immersive digital reality. Lecture notes in computer science, vol 11900. Springer, Cham, pp 3–32. https://doi.org/10.1007/978-3-030-41816-8_1
28. Rousselle F, Jarosz W, Novak J (2016) Image-space control variates for rendering. ACM Trans Graph 35(6):1–12. https://doi.org/10.1145/2980179.2982443
29. Schroers C, Bazin JC, Sorkine-Hornung A (2018) An omnistereoscopic video pipeline for capture and display of real-world VR. ACM Trans Graph 37(3):1–13. https://doi.org/10.1145/3225150
30. Sengupta S, Jayaram V, Curless B, Seitz S, Kemelmacher-Shlizerman I (2020) Background matting: the world is your green screen. In: 2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 2291–2300
31. Tang M, Wen J, Zhang Y, Gu J, Junker P, Guo B, Jhao G, Zhu Z, Han Y (2019) A universal optical flow based real-time low-latency omnidirectional stereo video system. IEEE Trans Multimedia 21(4):957–972. https://doi.org/10.1109/TMM.2018.2867266
32. W3C Immersive web working and community groups (2021) WebXR. https://immersiveweb.dev/. Accessed 18 May 2021
33. Wikipedia contributors (2021) Anaglyph 3D. https://en.wikipedia.org/wiki/Anaglyph_3D. Accessed 18 May 2021
34. Xie J, Girshick R, Farhadi A (2016) Deep3D: fully automatic 2D-to-3D video conversion with deep convolutional neural networks. In: Leibe B, Matas J, Sebe N, Welling M (eds) Computer vision – ECCV 2016. ECCV 2016. Lecture notes in computer science, vol 9908. Springer, Cham. https://doi.org/10.1007/978-3-319-46493-0_51
35. Xu F, Zhao T, Luo B, Dai Q (2018) Generating VR live videos with tripod panoramic rig. In: 2018 IEEE conference on virtual reality and 3D user interfaces (VR), pp 446–449. https://doi.org/10.1109/VR.2018.8448283
36. Zhang E, Cohen MF, Curless B (2016) Emptying, refurnishing, and relighting indoor spaces. ACM Trans Graph 35(6):1–14. https://doi.org/10.1145/2980179.2982432
37. Zhang J, Zhu T, Zhang A, Yuan X, Wang Z, Beetschen S, Xu L, Lin X, Dai Q, Fang L (2020) Multiscale-VR: multiscale gigapixel 3D panoramic videography for virtual reality. In: 2020 IEEE international conference on computational photography (ICCP), pp 1–12. https://doi.org/10.1109/ICCP48838.2020.9105244