electronics

MDPI

Article

Free-Viewpoint Navigation of Indoor Scene with 360° Field of View

Hang Xu [1], Qiang Zhao [2], Yike Ma [2,*], Shuai Wang [1,3], Chenggang Yan [1,3] and Feng Dai [2]

[1] School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China; hxu@hdu.edu.cn (H.X.)
[2] The Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100086, China
[3] Lishui Institute of Hangzhou Dianzi University, Lishui 323000, China
* Correspondence: ykma@ict.ac.cn

Abstract: By providing a 360° field of view, spherical panoramas can convey vivid visual impressions. Thus, they are widely used in virtual reality systems and street view services. However, due to bandwidth or storage limitations, existing systems only provide sparsely captured panoramas and have limited interaction modes. Although there are methods that can synthesize novel views based on captured panoramas, the generated novel views all lie on the lines connecting existing views. Therefore these methods do not support free-viewpoint navigation. In this paper, we propose a new panoramic image-based rendering method for novel view generation. Our method represents each input panorama with a set of spherical superpixels and warps each superpixel individually so the method can deal with the occlusion and disocclusion problem. The warping is dominated by a two-term constraint, which can preserve the shape of the superpixel and ensure it is warped to the correct position determined by the 3D reconstruction of the scene. Our method can generate novel views that are far from input camera positions. Thus, it supports freely exploring the scene with a 360° field of view. We compare our method with three previous methods on datasets captured by ourselves and by others. Experiments show that our method can obtain better results.

Keywords: viewpoint navigation; 360° field of view; multi-view stereo; real-time rendering

Citation: Xu, H.; Zhao, Q.; Ma, Y.; Wang, S.; Yan, C.; Dai, F. Free-Viewpoint Navigation of Indoor Scene with 360° Field of View. *Electronics* **2023**, *12*, 1954. https://doi.org/10.3390/electronics12081954

Academic Editors: Agnieszka Pregowska and Klaudia Proniewska

Received: 4 March 2023
Revised: 16 April 2023
Accepted: 20 April 2023
Published: 21 April 2023

1. Introduction

Panoramic images which have a 360° field of view allow the viewer to interactively look around a scene and can provide vivid visual impressions. In recent years, industrial communities began to manufacture economic spherical cameras for panoramic image capture and virtual reality headsets for panoramic image display. Thus, these images are more and more easily obtained and are widely used in computer graphics [1], virtual navigation [2–4], cultural heritage and the entertainment industry. For example, mainstream web map service providers feature street view services which provide panoramic images from positions along streets in the world. The Google Arts & Culture project [5] can let the public access panoramic views of famous museums. Some video-sharing websites, such as Youtube and Facebook, have also begun to provide panoramic data streaming services.

However these systems usually only provide a collection of panoramic images captured at sparse viewpoints, thus they have limited interaction modes, and only support panning, tilting and zooming from one viewpoint. If the viewer wants to explore the scene from a different viewpoint, they must *hop* from the current one to the other one, which may bring unpleasant visual discontinuity. To smoothly transit between sparsely captured panoramas, some panoramic view interpolation methods are proposed [6–8]. However, these methods can only generate panoramic images at viewpoints that are located on the path connecting two adjacent views and do not support *free-viewpoint* navigation. Recently, some other algorithms [9,10] have ben proposed to enable walking through a scene. These methods all follow the triangulation and warping pipeline [8,11], which first triangulates the input panoramas based on the feature points and then warps the resulting triangles to

a novel viewpoint. However, these methods cannot deal with the occlusion problem and may induce ghosting or artefacts as they do not take the image content into consideration.

The preliminary content of this paper has partially appeared in VR 2019 [12]. The overall contributions of this extended journal version can be summarized as listed below.

We propose a panoramic image-based rendering algorithm for free-viewpoint navigation of indoor scenes. Unlike previous methods, our method does not assume that the novel view lies on the line connecting input views and thus can let the user freely explore an indoor scene. Our method adopts a spherical superpixel-based per-view representation and locally warps each superpixel individually for novel view synthesis, which can prevent occlusion problems and artefacts. To be specific, given input panoramic images, our method first estimates the camera parameters for each image using structure from motion and densely reconstructs the 3D scene using two different multi-view stereo algorithms. We then over-segment each input panorama into superpixels using a spherical superpixel segmentation algorithm [13]. As the superpixel boundaries always correspond to depth discontinuities and image edges, the usage of this representation in warping can produce much fewer artefacts than triangulation-based methods. We leverage the generated superpixels to identify the regions that lack depth samples and synthesize missing ones by combining the complementary depth information from the two multi-view stereo algorithms. To generate novel views, our method locally warps each superpixel to virtual viewpoints with reprojection and shape-preserving constraints. As the superpixels are warped individually, our method does not suffer from occlusion problems and can obtain much more plausible results.

The pipeline of our method in this paper is shown in Figure 1. **The highlights of this paper are as follows:**

- We propose a panoramic image-based rendering algorithm for free-viewpoint navigation of indoor scenes. Unlike previous methods, our method does not assume that a novel view lies on the line connecting input views and thus can let the user freely explore the indoor scene.
- We explore a spherical superpixel-based per-view representation and locally warp each superpixel individually for novel view synthesis, which can prevent us from occlusion problems and artefacts.
- We have tested our method on downloaded and self-captured panoramic datasets. Experimental results show that our method achieves better performance compared with baselines.

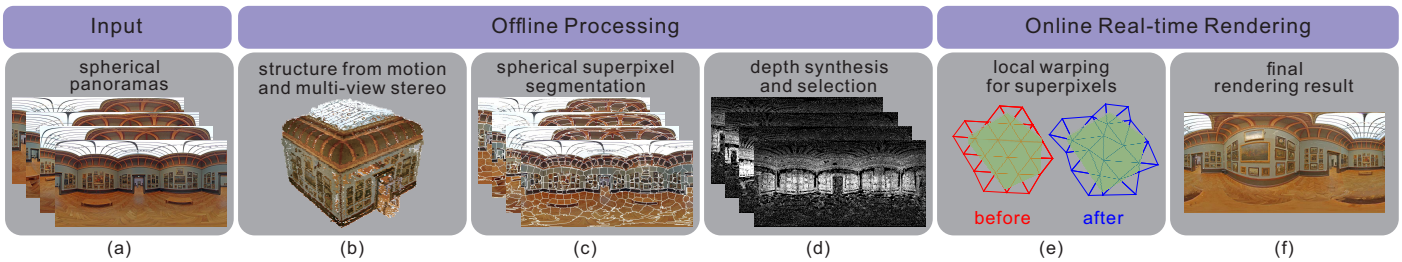

Figure 1. The pipeline of our method. Given a set of input panoramas (**a**), we first use structure from motion algorithm to estimate the camera poses and multi-view stereo for dense reconstruction (**b**). Then, we generate spherical superpixels for each image (**c**), based on which we synthesize missing depth samples and select some reliable ones for novel view synthesis (**d**). Finally, we warp each superpixel locally (**e**) and render the warped superpixels in real-time (**f**). See text for more details about each stage.

The remainder of this paper is organized as follows. Section 2 reviews the most related work. We describe the overview of our method in Section 3. The 3D reconstruction of the scene from input panoramas and the generation of depth maps are introduced in Section 4. The spherical superpixel local warping and rendering are described in Section 5. After

discussing the experimental results in Section 6, we conclude the paper and give avenues for future work.

2. Related Work

In this section, we introduce related works, including 3D reconstruction from spherical panoramas, planar and panoramic image-based rendering and free-viewpoint navigation.

2.1. 3D Reconstruction from Panoramas

Free-viewpoint navigation as proposed in this paper relies on scene reconstruction. Zioulis et al. [14] presented a learning framework to estimate scene depth from a single 360° image in a completely supervised manner with ground truth depth. Later, they [15] also explored spherical view synthesis for learning monocular 360° depth in a self-supervised manner. Depth maps from omnidirectional stereo images were proposed in [16]. However, the above methods can only estimate depth from one image. When multiple spherical images are given, the camera parameters of each spherical image cannot be computed. Wang et al. [17] proposed a novel self-supervised learning approach for predicting the omnidirectional depth and camera motion from a 360° video. This method can only estimate the relative pose between two images. Kim and Hilton [18] presented a 3D environment modelling method using multiple pairs of spherical stereo images. For accurate surface reconstruction, they designed a PDE-based disparity estimation method which produces continuous depth fields with sharp depth discontinuities. The work in [19] proposed a novel method for estimating the 3D geometry of indoor scenes based on multiple spherical images. This method uses optical flow algorithms to obtain point correspondences for dense depth map estimation. EPAR [20] is an efficient and privacy-aware augmented reality framework for indoor location-based services, which can improve the quality of scene reconstruction. Our work reconstructs the scene from multiple spherical images by extending existing planar structures from motion and multi-view stereo algorithms to spherical ones.

2.2. Planar Image Based Rendering

Image-based rendering refers to the techniques that rely primarily on the source images to produce new virtual views. A good survey can be found in [21]. Here, we only introduce some works that are most related to ours. Chaurasia et al. [22] introduced a new image-based rendering algorithm that is robust to missing or unreliable geometry, providing plausible novel views even in regions quite far from the input camera positions. Their method synthesizes depth for poorly reconstructed regions and locally warps the input images to novel views. Our method adopts a similar pipeline to this approach, but takes panoramic images as the input. Hedman et al. [23] proposed an image-based rendering algorithm for an indoor scene. Their method combines a global mesh from indoor-friendly depth sensors and depth maps from multi-view stereo for improved reconstruction. Then a scalable rendering algorithm, which applies mesh simplification and tiling, is designed to accelerate the rendering speed. Hedman et al. [24] presented a deep blending approach for image-based rendering, in which held-out real image data are used to learn blending weights to combine input photo contributions. Additionally, they combined two complementary multi-view stereo reconstructions, which is also adopted in our work for 3D reconstruction based on panoramic images.

2.3. Panoramic Image Based Rendering

Compared with traditional images, panoramas provide a 360° field of view and are widely used in virtual navigation systems. Uyttendaele et al. [3] filmed a tour of an environment with panoramic videos and designed a system that let users move freely along a set of predefined tracks and choose between different directions of motion at decision points. Chen et al. [25] presented a system that integrates a map with a video automatically constructed from panoramic imagery captured at close intervals along the route. The speed

and field of view of the video are automatically varied to highlight turns and landmarks. Huang et al. [26] proposed an approach that synthesizes new views with a rotational and translational motion of the viewpoint, so that the input monoscopic panoramic video can be played in full stereo.

Taking sparsely captured panoramas as input, some other methods perform panoramic image-based rendering. Kolhatkar and Laganière [7] proposed a real-time cubic panorama generation algorithm, which first computes the optical flow field between the adjacent panoramas and then uses view morphing to generate virtual viewpoints between existing views. The Cube2Video [8] framework first detects and matches feature points between adjacent panoramas, then triangulates the matched points and finally warps corresponding triangles for novel view generation. These two methods can only generate novel views that are located on the line connecting two adjacent views and cannot support free-viewpoint navigation. Following the pipeline of Cube2Video, Kawai et al. [9] proposed a panorama interpolation technique that enables simplified walk-through in real space. Andersen and Popescu [10] presented a system that enables the user to acquire a collection of panoramic images sufficient for virtual navigation by image morphing. All these methods globally warp the input images and cannot deal with the occlusion problem.

2.4. Free-Viewpoint Navigation

Free viewpoint navigation can also be implemented by light-field rendering [27]. New views from arbitrary camera positions can be generated by simply combining and resampling the available images without knowing depth information or feature matching. Considering the large field of view, different strategies have been introduced to compute a panoramic light field. The method proposed in [28] transforms light fields with small fields of view into rays in 3D space, then extracts panoramic light fields by blending registered rays. Taguchi et al. [29] present a wide-angle light-field capturing device based on a spherical catadioptric imaging system. The light-field data are extracted through accurate geometric modelling of captured multi-perspective photos. Krolla et al. [30] captured several panoramic images by moving a spherical camera vertically and combining the captured images in a panoramic light field. Panoramic light fields can also be recovered from densely captured hand held video. Please note that although light-field rendering supports free-viewpoint navigation, the moving range of the virtual view is restricted because of the parameterization of the light field.

3. Overview

Our method shares the same spirit with that of Chaurasia et al. [22]. Their method first estimates the camera parameters using the structure from motion algorithm [31] and reconstructs a 3D point cloud using multi-view stereo [32]. The point cloud is projected into cameras to obtain depth maps with sparse depth samples. Then, each input image is over-segmented into superpixels, which are used to identify poorly reconstructed superpixels and synthesize missing depth values. Finally, a local shape-preserving warp is performed on each superpixel to allow plausible novel view generation. Our method follows a similar pipeline as shown in Figure 1, but has made some modifications and adaptions for panoramic input. Given a set of spherical panoramas taken from multiple viewpoints, we first estimate the camera parameters using the structure from motion algorithm by applying an imaging sphere model [33]. We then convert each spherical panorama into multiple perspective planar images, which are sent to a multi-view stereo algorithm for dense reconstruction. We over-segment the input panoramas using a spherical superpixel algorithm [13], which takes the geometry of panoramas into consideration. Finally, we synthesize depth values for spherical superpixels which do not contain any depth samples. In the online stage, given the position and orientation of the novel view, we use reliable depth samples in each spherical superpixel as constraints in the local warping and render the warped mesh overlaid on each superpixel for novel view generation.

4. Depth Map Generation

In this section, we introduce how to generate depth maps from the input. We first estimate the camera parameters for each input panorama and then generate depth maps with two complementary multi-view stereo algorithms. Finally, we combine the depth information of these two algorithms and synthesize some missing depth values, which is followed by reliable depth sample selection for local warping.

4.1. Structure from Motion

Structure from motion is the process of estimating the camera parameters and reconstructing the sparse 3D structure from a collection of images taken from different viewpoints. It commonly starts with feature extraction and matching, followed by 3D structure and camera motion estimation, which is usually implemented by minimizing the Euclidean reprojection error through bundle adjustment. To extract camera parameters for the input panoramas, we use the Bundler software package [31]. As the original Bundler is designed for planar images, three modifications should be made to adapt it to deal with spherical panoramas.

First, the input spherical panoramas are represented in an equirectangular projection format, which has large image distortions, especially in polar regions. Thus, traditional feature detectors and descriptors cannot be used directly on the input images. In this paper, we convert each spherical panorama into a cube map and detect SIFT [34] features on each face of the cube map. Then, the features from all six faces are used as the features of the original panorama, and feature matching is performed to find correspondences between panoramas. Please note that although there are feature extractors designed for spherical panoramas, such as [35], we use the above strategy. This is because we also need to convert input panoramas into cube maps in the following multi-view stereo stage.

Second, given a world point X in 3D space, its projection into a spherical camera is represented by a unit vector, i.e.,

$$\mathrm{x} = \frac{\mathrm{RX} + \mathrm{t}}{\|\mathrm{RX} + \mathrm{t}\|}, \tag{1}$$

where R and t are the rotation matrix and the translation vector of the spherical camera, respectively. This is different from the imaging model for traditional planar images, which can be written as $\mathrm{x} = \mathrm{K}[\mathrm{R}\ \mathrm{t}]\mathrm{X}$, where K is the calibration matrix of the pinhole camera [36]. The spherical cameras are implicitly calibrated and we only need to estimate the external parameters, i.e., R and t, for each input panoramic image.

Third, due to the fact that each pixel of spherical panorama is actually a 3D point located on the unit sphere, the Euclidean distance minimized in the cost function of Bundler is replaced with the angle between the measured and reprojected pixel position, i.e.,

$$d(\mathrm{x}, \hat{\mathrm{x}}) = \arccos(\mathrm{x} \cdot \hat{\mathrm{x}}), \tag{2}$$

where x is the measured pixel position, $\hat{\mathrm{x}}$ is the reprojected pixel position derived from estimated 3D point $\widehat{\mathrm{X}}$ and camera motion $\widehat{\mathrm{R}}$ and $\widehat{\mathrm{t}}$ through Equation (1).

4.2. Multi-View Stereo

After the camera parameters for each input panorama are estimated, we perform multi-view stereo for dense reconstruction. As there are no algorithms designed for spherical panoramas, we also convert the panoramas into cube maps as shown in Figure 2. Given a panorama with camera parameters R and t, the camera matrix for face i of the cube map converted from this panorama is $\mathrm{P} = \mathrm{KR}_i[\mathrm{R}\ \mathrm{t}]$, where R_i is the rotation matrix that

transforms the point from the camera coordinate system of the panorama to that of face i. The following formulas show each R_i.

$$R_{+x} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad R_{-x} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{3}$$

$$R_{+y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \quad R_{-y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{4}$$

$$R_{+z} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad R_{-z} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

The matrix K is the calibration matrix of cube map faces and is expressed as

$$K = \begin{bmatrix} -s/2 & 0 & s/2 \\ 0 & s/2 & s/2 \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

where s is the size of cube map faces. With all the cube map faces converted from input panoramas and their camera matrices P, we send them to existing multi-view stereo algorithms for dense reconstruction.

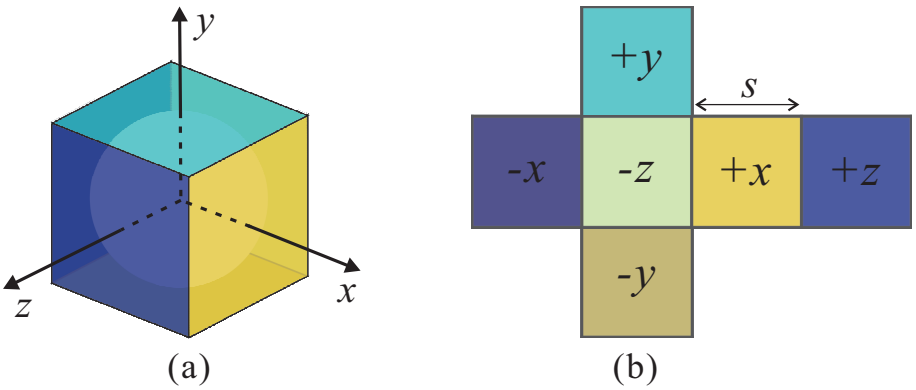

Figure 2. (**a**) Before performing multi−view stereo, we convert each spherical panorama (the sphere) to a cube map (the coloured cube). (**b**) The cube map uses the same camera coordinate system as spherical panorama and has six faces with size $s \times s$ which are perpendicular to $\pm x$, $\pm y$ and $\pm z$ axes of the camera coordinate system.

There are various multi-view stereo algorithms with different properties [37,38]. Some of them can give better detail accuracy, while some of them can give better global completeness. In this paper, we adopt two complementary methods as in [24]. The first one is CMPMVS [39], which generates a smooth global mesh estimation and can provide information in textureless regions. The other one is patch-based multi-view stereo PMVS [32], which can accurately capture small details but may break down for textureless regions. We send the faces of cube maps to each of these algorithms and generate dense reconstruction. Some reconstructed results are shown in Figure 3. We can see that CMPMVS can successfully reconstruct textureless regions such as walls, while PMVS can capture small details such as benches. Please note that in the local warping stage, we need the depth value of reconstructed pixels. As the output of CMPMVS is a triangular mesh, we use a ray tracing algorithm to generate depth maps from the mesh. The output of PMVS is a dense 3D point cloud for the scene and the information about images in which each reconstructed 3D point is visible. We also transform these data into depth maps. As we send the faces of cube maps

to multi-view stereo algorithms, we can only obtain the depth map for the faces of cube maps. We further transform the six depth maps for each cube map to a spherical depth map for each input panorama. Note that for planar images, the depth is computed as the distance between the reconstructed 3D point and the camera's centre in the direction of the principal ray, while for spherical panorama, the depth is computed as the distance between the 3D point and camera's centre, i.e., $\|RX + t\|$ in Equation (1). The spherical depth maps produced by CMPMVS and PMVS are shown in Figure 4b,c, respectively.

Figure 3. CMPMVS (**left**) can successfully reconstruct the textureless regions (the green square), while PMVS (**right**) can capture small details such as the benches (the blue square).

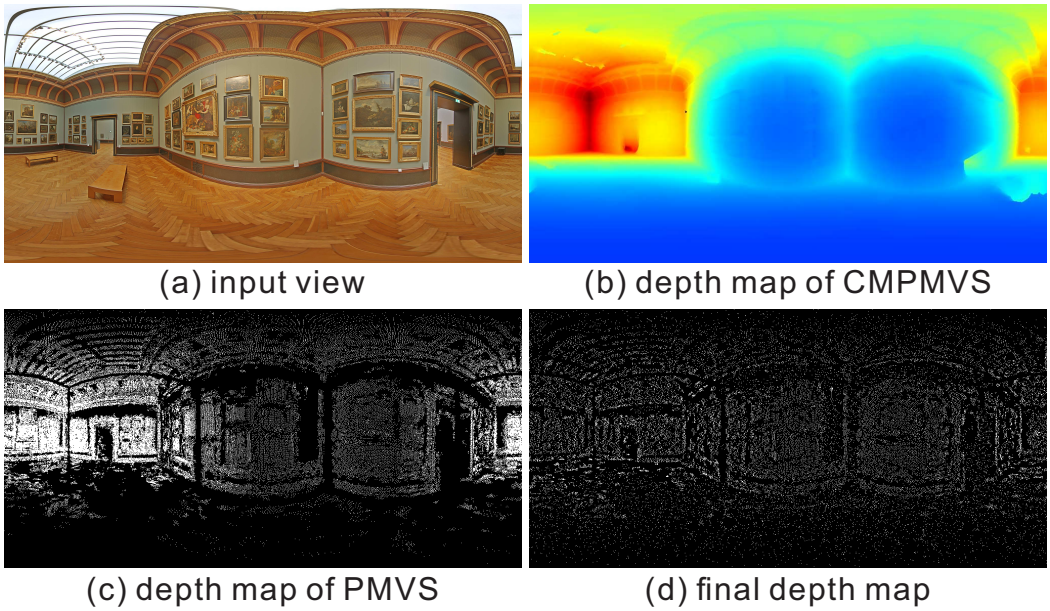

Figure 4. The input view (a) and depth maps (b,c,d). As the output of CMPMVS is a triangular mesh, the depth map of CMPMVS is much denser than that of PMVS. (b) In the depth map of CMPMVS, a cold colour means a small distance. (c) In the depth map of PMVS, each white pixel corresponds to a depth sample. (d) After depth synthesis and selection, the textureless regions will have depth samples, while some redundant depth samples in texture regions would be removed. Please zoom in and see subfigures (c,d) for comparison.

4.3. Depth Synthesis and Selection

As the two used multi-view stereo algorithms have complementary accuracy and completeness tradeoffs, we combine the depth information from these two methods for better depth map generation. To be specific, as our local warping strategy warps each superpixel individually, we first over-segment each input panorama into spherical superpixels. We identify target superpixels, which do not contain any depth samples reconstructed by

PMVS. Then we use that from CMPMVS to refine the depth maps or synthesize missing depth samples by borrowing from source superpixels which are visually similar and close to the target superpixels. Finally, for each spherical superpixel, we select reliable depth samples for local warping.

Superpixel Segmentation. Superpixels group similar pixels into perceptually meaningful atomic regions, which always conform well to the local image structures. These local image structures always correspond to object boundaries or occlusions. Thus, superpixels preserve depth discontinuities well, which is helpful in image-based rendering applications [22]. Most existing superpixel segmentation algorithms are designed for planar images. They do not deal with the image distortions of panoramas and cannot preserve coherence across panoramic image boundaries. In this paper, we adopt the SphSLIC method [13] for spherical superpixel segmentation. This method resembles the fundamental idea of the mature SLIC algorithm [40] and uses clustering to generate superpixels by explicitly considering the geometry for spherical images. It first initializes the superpixel centres with Hammersley points sampled on the sphere. Then it iterates between the assignment step, which associates each pixel to its nearest cluster centre, and the update step, which adjusts the cluster centres. Both the assignment step and update step take cosine dissimilarity as the spatial distance measure. The superpixels generated by SphSLIC reserve the coherence across panoramic image boundaries and have regular size and shape when mapped to the sphere.

Depth Synthesis. As PMVS is friable to textureless regions, there are some target superpixels that do not contain any depth samples. As we warp each superpixel individually, if a superpixel does not contain any depth samples, it would be warped incorrectly. We deal with this problem via a two-step strategy. In the first step, we check whether CMPMVS has reliable depth samples at target superpixels, where a reliable depth sample means it has good photometric consistency and would not induce visibility conflict as in [32]. If CMPMVS has reliable depth samples, we assign the depth value of these samples to that of PMVS. Otherwise, we use the method in [22] to synthesize depth samples for the target superpixels. Specifically, for each target superpixel, we first identify a set of visually similar superpixels, where the similarity is measured by the χ^2 distance between LAB histograms of two superpixels. Among these similar superpixels, we select the three closest source superpixels for depth synthesis. This is accomplished using a graph traversal algorithm, where the nodes of the graph are the spherical superpixels and the edge weights are χ^2 distance between LAB histograms of the adjacent superpixels. Note that the superpixels near the left boundary of the panoramic image are also treated as to be adjacent to those near the right boundary of the panoramic image. Benefiting from spherical superpixel segmentation, we do not need to consider this explicitly. The distance between one target superpixel and its visually similar superpixel is defined as the shortest path between the two nodes of the graph corresponding to two superpixels. Once the three source superpixels are determined, we can synthesize depth samples for the target superpixel by interpolating depth samples from the source superpixels with interpolation weights as the probability density function of depth samples.

Depth Selection. Please note that although there are target superpixels that do not contain any depth samples, there are also superpixels that contain redundant depth samples. As the depth samples for each superpixel have many similar values, the use of redundant depth samples in local warping could not obviously improve the rendering quality but could increase the computational complexity of the optimization problem in Section 5.1. Therefore we select a subset of depth samples that are distributed uniformly within each superpixel and use the selected depth samples in local warping. The depth map after depth synthesis and selection is shown in Figure 4d.

5. Local Warping and Rendering

In this section, we introduce how to warp the spherical superpixels for real-time novel viewpoint rendering.

5.1. Superpixel Local Warping

After we have estimated the camera parameters, computed the accurate depth map and generated compact spherical superpixels for each input panoramic image, we can synthesize the view for novel viewpoints by locally warping the spherical superpixels as shown in Figure 5. Unlike the work in [26], our method does not warp each input panoramic image globally, therefore it can synthesize parallax when the novel viewpoint is far from the input ones. Unlike the work in [22], our work warps superpixels on the surface of the unit sphere and is specifically designed for spherical panorama input.

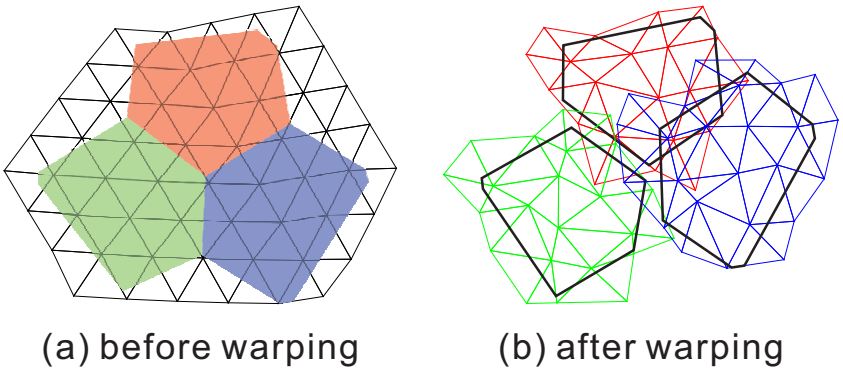

Figure 5. The illustration for local warping. In subfigure (**a**), there are three superpixels indicated by colour patches. We locally warp each superpixel individually. After warping, the blue superpixel slides under the red one, while the green superpixel moves away from the red and the blue one. This makes local warping can handle occlusions and disocclusions. In subfigure (**b**), the colour curves represent the warping mesh for each superpixel, and the black curves represent the boundaries of superpixels.

The warping is controlled by a triangular mesh overlaid on the unit sphere. To construct a nearly regular grid, we begin with an icosahedron inscribed inside a unit sphere and subdivide its faces into finer resolutions. For each spherical superpixel of the reference input panorama, we first determine a sub-mesh that covers this superpixel. We then warp the sub-mesh by disturbing its vertices v. Denoting the camera matrix for input viewpoint i and novel viewpoint n as C_i and C_n, respectively, our goal is to find the disturbed vertices v' that correspond to initial vertices v by minimizing an energy function involving two terms, which is widely used in video stabilization and image-based rendering applications [22,26,41]. The first term is the reprojection term or data term, which encourages the warped feature point to be close to its reprojected position; the second term is the shape-preserving term or regularization term, which tries to preserve the shape of the superpixel during warping.

For each spherical superpixel s, we use the set $\mathcal{D}_s$ of pixels that have depth samples after depth selection. For each pixel $\mathrm{x} \in \mathcal{D}_s$, we find the triangle $\triangle \mathrm{v}_1\mathrm{v}_2\mathrm{v}_3$ that pixel x falls in, then the reprojection energy for x is computed as

$$E_P(\mathrm{x}) = \|\alpha \mathrm{v}'_1 + \beta \mathrm{v}'_2 + \gamma \mathrm{v}'_3 - \mathrm{C}_n(\mathrm{C}_i^{-1}(\mathrm{x}))\|^2, \tag{7}$$

where α, β, γ are the barycentric coordinates of x in $\triangle \mathrm{v}_1\mathrm{v}_2\mathrm{v}_3$, i.e., $\mathrm{x} = \alpha \mathrm{v}_1 + \beta \mathrm{v}_2 + \gamma \mathrm{v}_3$. The symbols $\mathrm{v}'_1, \mathrm{v}'_2, \mathrm{v}'_3$ are the vertex positions after warping that we want to optimize. The function $\mathrm{C}_i^{-1}(\mathrm{x})$ takes the position and depth of pixel x as input, and outputs the 3D point in the scene projected to x based on the camera matrix C_i. Function $\mathrm{C}_n(\cdot)$ denotes the reprojection process of Equation (1). The shape-preserving energy is defined for each triangle t of the sub-mesh that covers superpixel s. Denote the vertices of t as $\mathrm{v}_{t1}, \mathrm{v}_{t2}$ and v_{t3}, the energy is defined as

$$E_S(t) = \|\mathrm{v}'_{t1} - (\mathrm{v}'_{t2} + u(\mathrm{v}'_{t3} - \mathrm{v}'_{t2}) + v\mathrm{R}^t_{90}(\mathrm{v}'_{t3} - \mathrm{v}'_{t2}))\|^2, \tag{8}$$

where (u, v) are the coordinates of v_{t1} represented in the local coordinate system formed by the vector $\mathrm{v}_{t3} - \mathrm{v}_{t2}$ and the vector perpendicular to $\mathrm{v}_{t3} - \mathrm{v}_{t2}$. R_{90}^{t} is the 90° rotation matrix defined by triangle t. $\mathrm{v}'_{t1}, \mathrm{v}'_{t2}, \mathrm{v}'_{t3}$ are the disturbed vertex positions to be optimized. The overall optimization problem for superpixel s is then

$$\arg\min_{\mathrm{v}'} \sum_{\mathrm{x}\in\mathcal{D}_s} E_P(\mathrm{x}) + \sum_{t\in\mathcal{T}_s} E_S(t), \text{ s.t. } \forall \mathrm{v}', \|\mathrm{v}'\|_2 = 1, \tag{9}$$

where $\mathcal{T}_s$ is the set of triangles contained in the sub-mesh that covers superpixel s. The constraints $\|\mathrm{v}'\|_2 = 1$ are added to ensure that the disturbed vertex positions v' are located on the surface of the unit sphere.

After the above problem is solved, we can warp the superpixel s in the reference panorama into novel views. During warping, we render a mesh with v' as the vertex position and v as the texture coordinates.

5.2. Real-Time Rendering

Given the camera pose of the novel view, we select four input panoramic images whose cameras are closest to the novel camera position. Please note that because panoramas have a 360° field of view, we do not consider the orientation of input cameras when selecting the input ones. We then warp the superpixels of each selected image as described previously and render the warped superpixels of each input image in a separate render target. When rendering the superpixels, we set the depth of each warped superpixel as the reprojected median of all depth samples contained within this superpixel and enable the depth test. This can let us deal with the occlusion and disocclusion problems. For example, some background superpixels would be occluded by foreground superpixels due to local warping. After the selected input panoramic images are warped and rendered, we blend the four rendering results together using the weights defined in the unstructured lumigraph rendering algorithm [42], which contain the angle term and distance term. Finally, inpainting is carried out as holes may appear when the novel view is significantly away from input cameras.

6. Experimental Results

In this section, we introduce the experimental results, including implementation details, the used dataset and the comparisons with previous algorithms.

6.1. Implementation Details

There are some issues we should pay attention to when solving the optimization problem in Section 5.1. First, as pixel x is located on the surface of the unit sphere, the barycentric coordinates of x do not satisfy $\alpha + \beta + \gamma = 1$. Second, unlike in the planar superpixel warping, the rotation matrices R_{90}^{t} are not identical for different triangles t. This is because the triangles are not on the same 3D plane. Third, instead of imposing the constraints $\|\mathrm{v}'\|_2 = 1$ explicitly to the minimization problem, we first solve the unconstrained problem and then normalize the resulting vertex vectors to unit vectors. We minimize the energy function for each superpixel by building a sparse linear system and solving it using CHOLMOD [43].

In the implementation, we generate the controlling mesh in advance and pre-compute the barycentric coordinates for each pixel and the rotation matrix for each triangle of the controlling mesh. The input spherical panoramas are converted into a cube map format as the graphics library natively supports cube map texturing. The spherical superpixel segmentation result is also represented in a cube map format, which is used to define an alpha matte when warping each superpixel. To assign a depth to one superpixel, we cannot set the depth of rendered fragment corresponding to the superpixel directly in the shader. This is because the warped superpixel is located on the unit sphere instead of a plane. In this paper, we scale the vertices of the controlling mesh of each superpixel by the depth and send the scaled vertices to the rendering pipeline for superpixel local warping.

For input spherical panoramas with 2048 × 1024 resolution, our current implementation achieves 25.24 FPS on a computer installed with Intel(R) Core(TM) i5-7500 CPU@3.40GHz and NVIDIA GeForce GTX 780 GPU.

6.2. Dataset

There are three sparsely captured panoramic image datasets used in this paper. The first two are downloaded from Google Arts & Culture [5], an online platform through which the public can access high-resolution images of artworks. The images of these two datasets are captured in Staatliches Museum Schwerin (termed SCHWERIN) and the CaRezzonico Museum of 18th Century Venice (termed CAREZZONICO), respectively. The last dataset is captured by ourselves and is termed HISTORY. Compared with the first two datasets, this dataset is more densely captured. That is to say, the distance between two adjacent views in this dataset is smaller. Another fact is that the HISTORY dataset is captured along a path, while the input views of the first two datasets are sparsely distributed in the scene. In Table 1, we give the number of input views and the number of reconstructed points for each scene of the three datasets. One reference input view and the 3D reconstruction results of PMVS for each scene are shown in Figure 6.

Table 1. The. number of input views and reconstructed points for each scene of the datasets.

Scene Name	♯ Input Views	♯ Reconstructed Points
SCHWERIN	9	176,999
CAREZZONICO	11	152,429
HISTORY	97	1,106,558

Figure 6. One input view and corresponding reconstruction result for each scene of the datasets.

6.3. Discussion

In this section, we compare our method with three existing panoramic image-based rendering methods. The first method is an optical-flow-based algorithm [7], which first computes the optical flow between the corresponding faces of two adjacent cubic panoramas and then morphs the views based on the interpolated optical flows for novel view generation. The second method is Cube2Video [8]. It first triangulates the matched feature

points between adjacent panoramas, then warps corresponding triangles through triangle-to-triangle homography determined by the relative pose between two input panoramas. The third method [26] has a similar pipeline to ours, but does not over-segment each input panorama into superpixels and warps the input views globally. As the input of the first two methods are cubic panoramas, we convert the spherical images in each dataset to cube maps for these two methods. Another fact is that the first two methods are designed for view interpolation, while the third method theoretically supports free-viewpoint navigation.

View Interpolation. We first compare different methods in the view interpolation task, where the novel viewpoints are located on the path connecting two adjacent input views. In this paper, we set the position and orientation of the novel views using the method given in Cube2Video [8]. To be specific, we denote the position of two adjacent views as 3D vector C_1 and C_2, respectively, and the orientation of these two views as a 3×3 rotation matrix R_1 and R_2, respectively. The position C_n of novel view is linearly interpolated as

$$C_n = (1-t)C_1 + tC_2, \tag{10}$$

where $t \in [0,1]$ is the interpolation weight. The orientation R_n of the novel view is computed by interpolating the matrix R_1 and R_2 using spherical linear interpolation of a quaternion (Slerp), i.e.,

$$q_n = \text{Slerp}(q_1, q_2; t) = \frac{\sin[(1-t)\theta]}{\sin\theta} q_1 + \frac{\sin(t\theta)}{\sin\theta} q_2, \tag{11}$$

where q_1, q_2 and q_n is quaternion corresponding to rotation matrix R_1, R_2 and R_n, respectively. $\theta = \arccos(q_1 \cdot q_2)$ is the angle between q_1 and q_2. For the optical-flow-based method, we scale the flow field from view at C_1 to view at C_2 with weight t and the flow field from view at C_2 to view at C_1 with weight $1-t$. Then the two scaled flow fields are used for novel view interpolation.

We first compare view interpolation results on the HISTORY dataset. The generated results of different methods and corresponding zoomed-in images are shown in the first and the second row of Figure 7, respectively. We can see that all four methods can obtain plausible interpolation results. This is because the HISTORY dataset is captured densely along a path, which is suitable for the view interpolation task. However, if we look at the zoomed-in images, we can find that global warping and our method can obtain the best interpolation results. Since the computation of optical flow cannot make the best tradeoff between the accuracy and the smoothness of the flow field, the results of the optical-flow-based method [7] contain obvious noise and artefacts. As the Cube2Video algorithm [8] is based on the triangulation of matched feature points, there would be ghosting artefacts or distortions if incorrectly matched feature points exist. Compared with these two baseline methods, global warping [26] and our method can generate results that are free from noise and ghosting artefacts.

We next compare view interpolation results of different methods on the SCHWERIN and CAREZZONICO datasets. As the input views of these two datasets are sparsely distributed in the scene, we manually specify a path along which the input views are interpolated. The interpolation results of different methods on these two datasets are given in the third and fifth row of Figure 7, respectively. The zoomed-in images of the interpolation results are given in the fourth and sixth rows. The optical-flow-based method [7] assumes that the image contents of the corresponding faces of two input cubic panoramas are similar and computes the flow between the corresponding faces. However, for some adjacent views in these two datasets, the camera movement is relatively large and contains orientation rotations. Thus, the assumption is not satisfied and the interpolation results are incorrect. Furthermore, as the flows are computed independently for each face, the consistency between flows from different faces is not guaranteed. Therefore, the interpolation result may contain seams as shown in the first zoomed-in image of the result of the CAREZZONICO dataset. Cube2Video algorithm [8] assumes that the region of each triangulated triangle

falls on a plane, which is not always true. Its interpolation results have obvious ghosting artefacts. For the global warping [26], it cannot deal with the occlusion problem well. It may generate unreasonable results such as the second zoon-in image on the CAREZZONICO dataset. In contrast, our method can still obtain satisfactory results. Please note that, even though we give the interpolation results of Cube2Video [8] here, it cannot successfully interpolate between every two adjacent views of these two datasets.

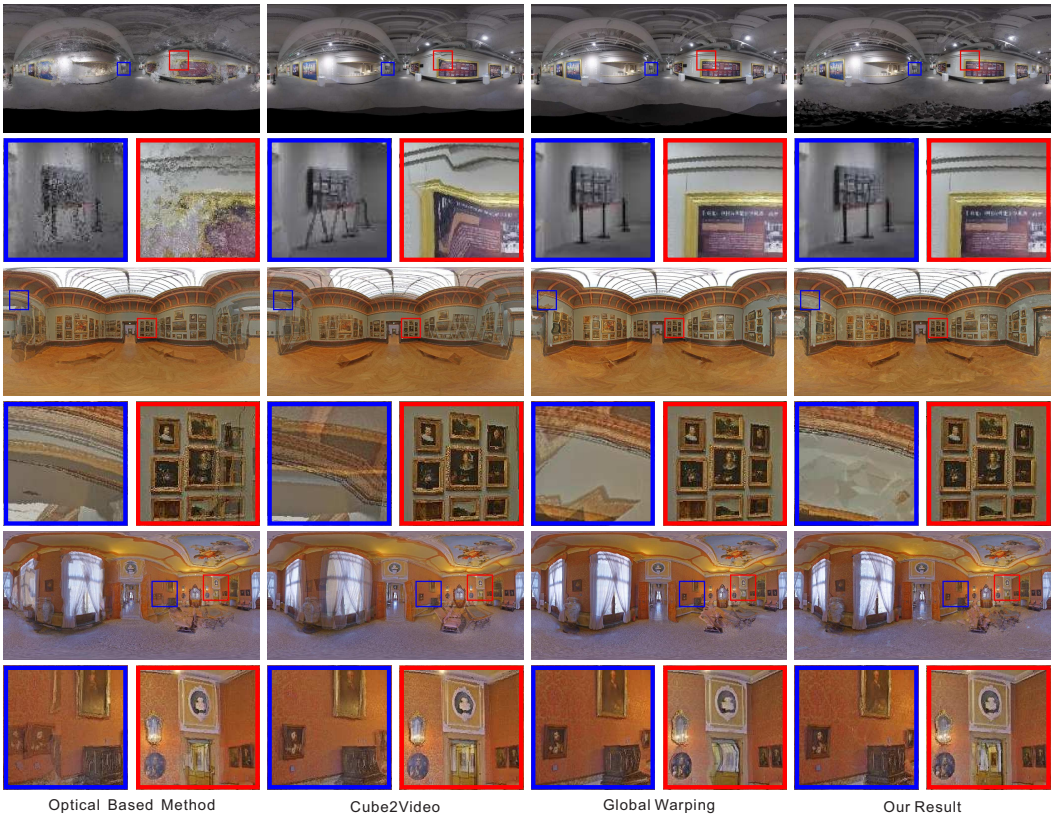

Figure 7. The view interpolation results of different methods on the HISTORY, SCHWERIN and CAREZZONICO datasets. Here we also give the zoomed-in images for more detailed comparisons.

Free-viewpoint Navigation. We next evaluate our method with a free-viewpoint navigation task on the SCHWERIN and CAREZZONICO datasets. As the optical-flow-based method [7] and Cube2Video [8] is designed for view interpolation task, we only compare our method with global warping [26]. For comparison purposes, we pre-define the free viewpoint paths and camera poses. In our setting, the cameras move along a circle. The generated results at novel viewpoints are shown in Figure 8. For global warping [26], as all the triangles of controlling mesh are warped globally, some of them may be stretched if the depth samples are not distributed evenly. At the same time, global warping cannot deal with the occlusion and disocclusion problems. Thus, it may produce results with distortions. In contrast, our method can obtain more reasonable results. However, our result contains visible seams due to different exposures, which is a limitation that will be discussed in the next section.

6.4. Limitation

Although our method can obtain reasonable results for both view interpolation and free-viewpoint navigation tasks, there are still some limitations. First, our method heavily replies on the quality of multi-view stereo reconstruction and depth synthesis. It is challenging to obtain high-quality reconstructions for indoor scenes which may contain very close objects. It is also necessary to fully reconstruct the scene, as our method generates panoramic novel views. Otherwise, the unreconstructed regions will appear as holes, which

would be overly smoothed by inpainting. Second, when capturing each dataset, different views may have different exposures. This may induce obvious seams between warped superpixels on the generated novel view. Simply using the definition of blending weights in [42] cannot deal with the problem well. Furthermore, if we move from one novel view to another, the lighting condition would also be changed suddenly. Third, we solve the optimization problem in Equation (9) on CPU; this may limit its parallelization.

Figure 8. The free-viewpoint navigation results of global warping and our method on the SCHWERIN and CAREZZONICO datasets.

7. Conclusions

In this paper, we propose a panoramic image-based rendering method, which supports free-viewpoint navigation. Our method represents each input panorama with a set of spherical superpixels and warps each superpixel individually. The warping is controlled by energy, which can preserve the shape of the superpixel and ensure the superpixel is warped to the correct position. Thus, our method can generate plausible novel views. The experiments on the downloaded and self-captured datasets show that our method can obtain better results.

In the future, firstly, we would like to adopt deep-learning-based methods in each step of the pipeline of our method. To be specific, we would use deep-learning-based bundle adjustment [44] and multi-view stereo [45] to improve the estimation of camera parameters

and refine the dense reconstruction of the scene. Deep blending algorithms [24] may be used for blending weights computation. Furthermore, deep-learning-based single image novel view synthesis [46] can also be incorporated into the pipeline of our method. Finally, we will introduce sensors to improve the performance of the reconstruction, so that our method is more adaptable to the environment.

Author Contributions: Methodology, H.X. and F.D.; Software, Y.M. and S.W.; Validation, H.X.; Writing—original draft, H.X.; Writing—review & editing, F.D. and Q.Z.; Supervision, S.W. and Y.M.; Project administration, C.Y.; Funding acquisition, F.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Key R&D Program of China under Grant (2022YFD2001601), the National Nature Science Foundation of China(62072438, U21B2024, 61931008, 62071415), Strategic Priority Research Program of Chinese Academy of Sciences (XDA28040000, XDA28120000), Natural Science Foundation of Shandong Province (ZR2021MF094), Key R&D Plan of Shandong Province (2020CXGC010804), Central Leading Local Science and Technology Development Special Fund Project (YDZX2021122) and Science & Technology Specific Projects in Agricultural High-tech Industrial Demonstration Area of the Yellow River Delta (2022SZX11).

Data Availability Statement: The data used to support the findings of this study could be found freely here: https://artsandculture.google.com (accessed on 21 April 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Greene, N. Environment Mapping and Other Applications of World Projections. *IEEE Comput. Graph. Appl.* **1986**, *6*, 21–29.
2. Chen, S.E. QuickTime VR: An Image-based Approach to Virtual Environment Navigation. In Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 6–11 August 1995; pp. 29–38.
3. Uyttendaele, M.; Criminisi, A.; Kang, S.B.; Winder, S.; Szeliski, R.; Hartley, R. Image-based interactive exploration of real-world environments. *IEEE Comput. Graph. Appl.* **2004**, *24*, 52–63.
4. Anguelov, D.; Dulong, C.; Filip, D.; Frueh, C.; Lafon, S.; Lyon, R.; Ogale, A.; Vincent, L.; Weaver, J. Google Street View: Capturing the World at Street Level. *Computer* **2010**, *43*, 32–38.
5. Google Arts and Culture. Available online: https://artsandculture.google.com (accessed on 21 April 2023).
6. Chan, Y.-F.; Fok, M.-H.; Fu, C.-W.; Heng, P.-A.; Wong, T.-T. A panoramic-based walkthrough system using real photos. In Proceedings of the Pacific Conference Computer Graphics and Application, Seoul, Republic of Korea, 5–7 October 1999; pp. 231–240.
7. Kolhatkar, S.; Laganière, R. Real-Time Virtual Viewpoint Generation on the GPU for Scene Navigation. In Proceedings of the Canadian Conference Computer and Robot Vision, Ottawa, ON, Canada, 31 May–2 June 2010; pp. 55–62.
8. Zhao, Q.; Wan, L.; Feng, W.; Zhang, J.; Wong, T. Cube2Video: Navigate Between Cubic Panoramas in Real-Time. *IEEE Trans. Multimed.* **2013**, *15*, 1745–1754.
9. Kawai, N.; Audras, C.; Tabata, S.; Matsubara, T. Panorama Image Interpolation for Real-time Walkthrough. In Proceedings of the ACM SIGGRAPH Posters, Anaheim, CA, USA, 24–28 July 2016; pp. 33:1–33:2.
10. Andersen, D.; Popescu, V. HMD-Guided Image-Based Modeling and Rendering of Indoor Scenes. In Proceedings of the Virtual Reality and Augmented Reality, London, UK, 22–23 October 2018; pp. 73–93.
11. Siu, A.M.K.; Lau, R.W.H. Image registration for image-based rendering. *IEEE Trans. Image Process.* **2005**, *14*, 241–252.
12. Dai, F.; Zhu, C.; Ma, Y.; Cao, J.; Zhao, Q.; Zhang, Y. Freely Explore the Scene with 360° Field of View. In Proceedings of the 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 23–27 March 2019.
13. Zhao, Q.; Dai, F.; Ma, Y.; Wan, L.; Zhang, J.; Zhang, Y. Spherical Superpixel Segmentation. *IEEE Trans. Multimed.* **2018**, *20*, 1406–1417.
14. Zioulis, N.; Karakottas, A.; Zarpalas, D.; Daras, P. OmniDepth: Dense Depth Estimation for Indoors Spherical Panoramas. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 453–471.
15. Zioulis, N.; Karakottas, A.; Zarpalas, D.; Alvarez, F.; Daras, P. Spherical View Synthesis for Self-Supervised 360 Depth Estimation. *arXiv* **2019**, arXiv:1909.08112v1.
16. Lai, P.K.; Xie, S.; Lang, J.; Laganière, R. Real-Time Panoramic Depth Maps from Omni-directional Stereo Images for 6 DoF Videos in Virtual Reality. In Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 23–27 March 2019; pp. 405–412.
17. Wang, F.E.; Hu, H.N.; Cheng, H.T.; Lin, J.T.; Yang, S.T.; Shih, M.L.; Chu, H.K.; Sun, M. Self-supervised Learning of Depth and Camera Motion from 360 Videos. In Proceedings of the Asia Conference on Computer Vision, Perth, Australia, 2–6 December 2018; pp. 53–68.
18. Kim, H.; Hilton, A. 3D Scene Reconstruction from Multiple Spherical Stereo Pairs. *Int. J. Comput. Vis.* **2013**, *104*, 94–116.

19. da Silveira, T.L.T.; Jung, C.R. Dense 3D Scene Reconstruction from Multiple Spherical Images for 3-DoF+ VR Applications. In Proceedings of the IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 23–27 March 2019; pp. 9–18.
20. Peng, Z.; Hou, S.; Yuan, Y. EPAR: An Efficient and Privacy-Aware Augmented Reality Framework for Indoor Location-Based Services. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 22–25 May 2022; pp. 8948–8955.
21. Shum, H.Y.; Chan, S.C.; Kang, S.B. *Image-Based Rendering*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008.
22. Chaurasia, G.; Duchene, S.; Sorkine-Hornung, O.; Drettakis, G. Depth Synthesis and Local Warps for Plausible Image-based Navigation. *ACM Trans. Graph.* **2013**, *32*, 30:1–30:12.
23. Hedman, P.; Ritschel, T.; Drettakis, G.; Brostow, G. Scalable Inside-Out Image-Based Rendering. *ACM Trans. Graph.* **2016**, *35*, 231:1–231:11.
24. Hedman, P.; Philip, J.; Price, T.; Frahm, J.M.; Drettakis, G.; Brostow, G. Deep Blending for Free-viewpoint Image-based Rendering. *ACM Trans. Graph.* **2018**, *37*, 257:1–257:15.
25. Chen, B.; Neubert, B.; Ofek, E.; Deussen, O.; Cohen, M.F. Integrated Videos and Maps for Driving Directions. In Proceedings of the ACM Symposium on User Interface Software and Technology, Victoria, BC, Canada, 4–7 October 2009; pp. 223–232.
26. Huang, J.; Chen, Z.; Ceylan, D.; Jin, H. 6-DOF VR videos with a single 360-camera. In Proceedings of the IEEE Virtual Reality (VR), Los Angeles, CA, USA, 18–22 March 2017; pp. 37–44.
27. Levoy, M.; Hanrahan, P. Light Field Rendering. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, New Orleans, LA, USA, 4–9 August 1996; pp. 31–42.
28. Birklbauer, C.; Bimber, O. Panorama Light-Field Imaging. *Comput. Graph. Forum* **2014**, *33*, 43–52.
29. Taguchi, Y.; Agrawal, A.; Veeraraghavan, A.; Ramalingam, S.; Raskar, R. Axial-cones: Modeling Spherical Catadioptric Cameras for Wide-angle Light Field Rendering. *ACM Trans. Graph.* **2010**, *29*, 172:1–172:8.
30. Krolla, B.; Diebold, M.; Goldlücke, B.; Stricker, D. Spherical Light Fields. In Proceedings of the British Machine Vision Conference, Nottingham, UK, 1–5 September 2014.
31. Snavely, N.; Seitz, S.M.; Szeliski, R. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. Graph.* **2006**, *25*, 835–846.
32. Furukawa, Y.; Ponce, J. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1362–1376.
33. Gluckman, J.; Nayar, S.K. Ego-motion and omnidirectional cameras. In Proceedings of the International Conference on Computer Vision, Bombay, India, 4–7 January 1998; pp. 999–1005.
34. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110.
35. Guan, H.; Smith, W.A.P. BRISKS: Binary Features for Spherical Images on a Geodesic Grid. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4886–4894.
36. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2003.
37. Scharstein, D.; Szeliski, R. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *Int. J. Comput. Vis.* **2002**, *47*, 7–42.
38. Multi-View Stereo: A Tutorial. *Found. Trends Comput. Graph. Vis.* **2015**, *9*, 1–148.
39. Jancosek, M.; Pajdla, T. Multi-view reconstruction preserving weakly-supported surfaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 20–25 June 2011; pp. 3121–3128.
40. Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Süsstrunk, S. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2274–2282.
41. Liu, F.; Gleicher, M.; Jin, H.; Agarwala, A. Content-preserving Warps for 3D Video Stabilization. *ACM Trans. Graph.* **2009**, *28*, 44:1–44:9.
42. Buehler, C.; Bosse, M.; McMillan, L.; Gortler, S.; Cohen, M. Unstructured Lumigraph Rendering. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 12–17 August 2001; pp. 425–432.
43. Chen, Y.; Davis, T.A.; Hager, W.W.; Rajamanickam, S. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. Math. Softw.* **2008**, *35*, 22.
44. Tang, C.; Tan, P. BA-Net: Dense Bundle Adjustment Networks. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
45. Huang, P.; Matzen, K.; Kopf, J.; Ahuja, N.; Huang, J. DeepMVS: Learning Multi-view Stereopsis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2821–2830.
46. Liu, M.; He, X.; Salzmann, M. Geometry-Aware Deep Network for Single-Image Novel View Synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4616–4624.