PanoSynthVR: Toward Light-weight 360-Degree View Synthesis from a Single Panoramic Input

John Waidhofer[*] California Polytechnic State University
Richa Gadgil[†] Carnegie Mellon University
Anthony Dickson[‡] University of Otago
Stefanie Zollmann[§] University of Otago
Jonathan Ventura[¶] California Polytechnic State University

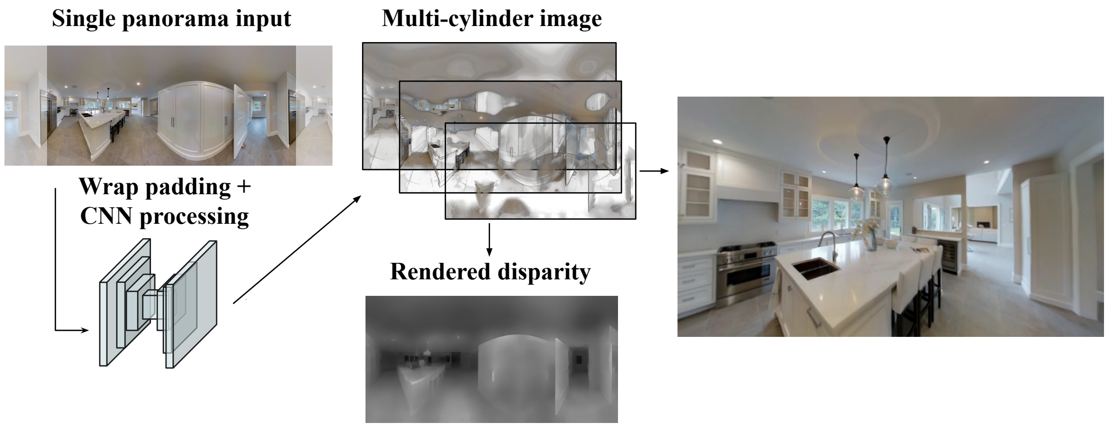


Figure 1: Our method provides free-viewpoint view synthesis from a single input panorama. The multi-cylinder image representation consists of semi-transparent cylindrical layers at varying depths. We produce a multi-cylinder image by processing a horizontally wrap-padded panorama in a convolutional neural network. Real-time view synthesis is achieved by projecting and compositing the textured cylinders with over blending. (Figure inspired by [28].)

Abstract

We investigate how real-time, 360° view synthesis can be achieved on current virtual reality hardware from a single panoramic image input. We introduce a light-weight method to automatically convert a single panoramic input into a multi-cylinder image representation that supports real-time, free-viewpoint view synthesis rendering for virtual reality. We apply an existing convolutional neural network trained on pinhole images to a cylindrical panorama with wrap padding to ensure agreement between the left and right edges. The network outputs a stack of semi-transparent panoramas at varying depths which can be easily rendered and composited with over blending. Quantitative experiments and a user study show that the method produces convincing parallax and fewer artifacts than a textured mesh representation.

Index Terms: Computing methodologies—Artificial intelligence—Computer vision; Human-centered computing—Human computer interaction (HCI); Computing methodologies—Computer graphics—Graphics systems and interfaces—Virtual reality;

1 Introduction

Depth cues such as binocular stereo and motion parallax are important aspects of immersion in virtual reality (VR). To achieve these effects when rendering captured imagery and video, a system must be able to provide free-viewpoint view synthesis in real time.

Traditionally, view synthesis techniques require specialized multi-camera hardware rigs to capture panoramic scenes [1, 4]. This requirement can make panoramic view synthesis inaccessible, since most consumers typically only have access to monocular smartphone cameras.

Recent work explores more practical approaches to view synthesis which only require a single consumer-grade camera for input [28, 29, 32]. For example, Tucker and Snavely created a multi-plane image (MPI) model that generates novel views from a single camera input, but is limited to a small field-of-view and single viewing direction [28]. While this is an effective solution for view synthesis in a single direction, it is not applicable to VR applications which often require view synthesis in all directions.

Our approach builds upon this work by requiring only a single panorama as input for novel panoramic view synthesis, which can be easily captured with a smartphone camera or 360 camera and rendered in a VR environment. Our model accepts cylindrical panoramas as input, which can be easily converted from equirectangular projections and cube maps. This approach is thus compatible with the vast amount of existing 360 imagery available on the Internet. In this work, we address the challenges inherent in applying a single-view MPI network to panoramic images, and present what is to our knowledge the first work capable of producing a multi-cylinder image (MCI) representation from a single panoramic input. We call our approach PanoSynthVR. Quantitative experiments and a user study show that our method produces convincing parallax and was preferred by study participants over a textured mesh representation using estimated depth maps.

Our specific contributions are as follows.

- We introduce a light-weight method to automatically and quickly (on average 383.5 ms) produce a panoramic view syn-

[*]e-mail: jwaidhof@calpoly.edu
[†]e-mail: rgadgil@cs.cmu.edu
[‡]e-mail: dican732@student.otago.ac.nz
[§]e-mail:stefanie.zollmann@otago.ac.nz
[¶]e-mail:jventu09@calpoly.edu

thesis representation from a single panoramic image, which can be easily captured with a smartphone or a consumer-grade 360 camera. Our method achieves real-time view synthesis rendering on consumer VR headsets. The method can be run on off-the-shelf computers/laptops.
- We introduce a pipeline for generating synthetic panoramas and perspective views and assemble a dataset for evaluating panoramic view synthesis methods.
- We introduce an improved evaluation methodology for panoramic view synthesis with an unknown scaling factor, which is more robust to outliers in the estimated disparity maps.
- We evaluated our method and compared to baseline methods in a user study to validate our approach.
- We share our code and data publicly[1], including an optimized WebXR renderer which is compatible with both desktop browsers and VR headsets.

2 Related Work

The goal of this work is to develop a practical technique to quickly capture and reconstruct a 360° scene with free-viewpoint rendering in a small region around the point of capture, using consumer hardware. A panoramic image displays visual data in all directions but does not allow for translational head movement. A simple extension to provide a convincing illusion of presence in a panoramic scene involves reconstructing or inferring a per-pixel panoramic depth map, so that the scene can be re-projected with parallax. The depth map can be estimated from multi-view stereo using a multi-camera rig, but this makes the system inaccessible to consumers. An alternative is to infer the depth map from a single panorama using a neural network [12,30,35,36].

Depth maps are not ideal for view synthesis, because holes will appear in the depth map when projected to new viewpoints. One solution is to make a mesh out of the depth map. The mesh can be textured from input photographs using projective texture mapping. View-dependent texture mapping [6,7] enables higher-fidelity view synthesis by blending between source images depending on the viewing angle, to reproduce effects such as transparency and specular reflections. For example, several approaches [2,3,17] combine depth estimation or proxy geometry with flow-based blending to achieve view-dependent effects, thus requiring many input images. Similarly Huang et al. [10] estimate a sparse point cloud from a set of many input panorama images, and perform view synthesis using a novel warping technique. The SynSin method of Wiles et al. [31] uses a neural network to produce both a depth map and a feature map which can then be rendered to new viewpoints using a neural renderer.

Another improvement upon the per-pixel depth map is to use a layered depth approach, where each pixel is associated with multiple depth values [25]. Several recent methods [8,9,24] adopt layered representations for 360 view synthesis, where the scene consists of two or three layers and they use extrapolation and in-painting to fill holes. However only the method of Serrano et al. [24] is able to work with a single panoramic input image, the others require many input images captured at different positions. Later layered representations such as the multi-depth panorama [16] and layered mesh representation [4] produce more accurate view synthesis results but require a multi-camera rig.

Other previous methods explore sophisticated techniques to produce high-quality layers from a single input image [11,13,20,26] but do not support panoramic scenes. These methods all use a similar approach of expanding a measured or estimated depth map into a few soft mesh layers with in-painting. In contrast, our MCI representation has many soft cylindrical layers (32 in our work, although even more layers would be beneficial).

[1]https://github.com/jonathanventura/PanoSynthVR

The flexibility of layered depth representations makes them ideal for efficient storage and rendering, but also makes them difficult to use as a target representation in an end-to-end learning pipeline. An alternative is a multi-plane image (MPI) [34], where each layer is a flat plane placed at a fixed depth from the capture point. The regularity of the representation makes it suitable to be the output of a convolutional neural network. Tucker and Snavely [28] introduced a method to infer an MPI from a single perspective image. An important limitation of the MPI representation is that the view synthesis result degrades when the viewpoint moves too far from the origin. Mildenhall et al. [18] established a quantitative relationship between the translation magnitude and the required number of layers in the MPI to produce sufficient sampling. Extensions of the MPI address this issue by allowing for adaptive sampling of the layer depths [14, 15]. Attal et al. [1] trained a neural network to produce a multi-sphere image (MSI) from a stereoscopic panorama input. This work is likely the closest to ours, since we use a neural network to produce a multi-cylinder image (MCI) from an input panorama; however, we use a monoscopic panoramic input, rather than stereoscopic, thus requiring less specialized equipment.

Broxton et al. [4] find a bridge between both the layered depth approach and multi-layer image approaches. They use deep learning to produce a MSI from a multi-camera rig, and then compress the MSI into a layered mesh representation for more efficient storage and rendering. However, their approach is not directly transferable to our setting, since they require a multi-camera rig.

A neural radiance field (NeRF) [19] supports a broader range of viewpoints by representing the entire scene as a continuous color and density field. To synthesize a viewpoint, the NeRF is sampled at a dense set of points along each viewing ray, and the composite color of each pixel is determined by integrating the samples along the ray. While this approach enables high-fidelity view synthesis, it has several limitations: it requires a dense set of input images observing the scene from many angles and positions; both creating and rendering the NeRF are highly compute-intensive; and the learned representation is specific to the scene it was trained on, and does not generalize to new scenes. Yu et al. produce a NeRF from a single input image [32], but their test scenes are limited to small objects and their results do not match the high-quality of the original NeRF. Li et al. [14] produce a planar NeRF from a single input image, but don't support 360 scenes.

To the best of our knowledge, only the method of Serrano et al. [24] can produce view synthesis in a 360° scene with a single panorama as input. All other methods either are limited to a small field of view or require many images captured handheld or with a specialized camera rig. While Serrano et al. [24] use an in-painted layered depth representation, in this work, we explore how to adopt the advantages of the MPI representation to 360 scenes with a single panoramic image input. We considered comparing our approach to Serrano et al. [24], but decided against it as their approach is designed for dynamic 360 videos. Previous work by Bertel et al. [3] used a comparison for 360 static OmniPhotos against Serrano by creating a "static" 360 video from several static 360 images. However, they mentioned that "The resulting static scene does not play to their method's strength of propagating background information behind dynamic objects." Thus we decided against using this for comparison as our work focuses on single images and not videos.

To summarize, we find three dominant representations for view synthesis from a single image in the state-of-the-art: soft mesh layers [3, 11, 13, 20, 26, 31]; a multi-layer image [1, 4, 28, 34]; and a continuous volumetric representation [14, 19, 32]. However, few previous studies have considered how to adapt these representations to a single panoramic input. In this work, we focus on the multi-layer approach and explore the use of an MCI representation for 360° view synthesis from a single input panorama. While a complete comparison between the various representations discussed here would be interesting, we leave such a study for future work, since each competing method needs to be carefully adapted for panoramic input.

3 Method

The goal of our method is to produce free-viewpoint view synthesis from a single panoramic input that is suitable for rendering in VR headsets. Our method consists of two main steps. 1) We compute a multi-cylinder image (MCI) representation that consists of a number of semi-transparent cylindrical layers at varying depths (Figure 1). 2) We perform real-time view synthesis by projecting and compositing the cylindrical layers using over blending.

3.1 Multi-cylinder Image Representation

In the first step, our system converts a single panoramic image into a multi-cylinder image (MCI) representation. It builds upon the single-view multi-plane image (MPI) deep convolutional neural network [28], which produces layered images using a U-Net style architecture. The first seven convolutional blocks, which consist of two convolutional layers each, are concatenated element-wise with the following seven up-sampling blocks. Finally, two convolutional blocks produce the layered image output. The model uses a three part loss, accounting for similarity, smoothness and depth. The code for the original MPI network is publicly available online[2].

The output MPI consists of semi-transparent image layers at varying depths, which are easily rendered and composited with over blending to achieve view synthesis. Each colored pixel is weighted in importance by its alpha value, which leads to a soft appearance of the rendered scene. While the results of MPI are great for interactive desktop applications, they are limited by their directional field of view. However, VR experiences require an immersive 360º representation with an omnidirectional field of view. A MCI representation can provide such an omnidirectional view.

We originally tested applying the single-view MPI network to each face of a cube map separately. However, this resulted in disagreements between the cube map faces. To overcome this issue, we decided to adopt a cylindrical panorama representation. Although the cylindrical projection introduces distortions, locally it appears similar to a pinhole image. However, we found the vertical seam on the left and right edges of the panorama to still have inconsistent disparity estimations (Figure 2).

To address this, we added horizontal wrap padding to the input image. We copy the left half of the image and concatenate it to the right side and visa versa. After processing the padded image with the MCI network, we crop the padding from each layer. With the addition of wrap padding, we no longer observe disagreement across the seam, and the MPI network is able to produce a consistent representation around the entire cylindrical panorama (Figure 2).

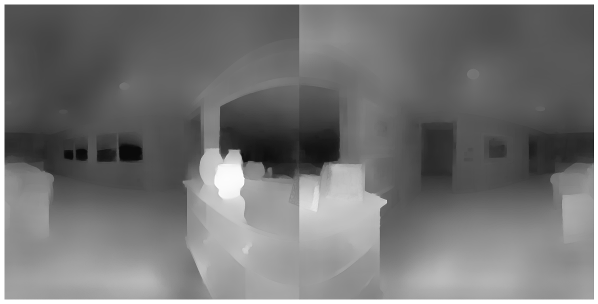
3.2 View Synthesis

Once the MCI representation is computed, we use view synthesis to create novel views and render them within VR headsets such as the Oculus Quest.

For the view synthesis, we create a 3D scene consisting of a series of concentric cylindrical meshes. The radius of the cylinders equals the corresponding depth of the layer and the height is set to be twice the disparity value. We map each layer texture to the inside of a cylinder and activate alpha blending to visually combine the layers. By placing the virtual camera inside the innermost cylinder, our renderer produces a realistic soft layer effect for an entire panoramic scene. Stereo rendering computes the respective image for each eye.

[2]`https://github.com/google-research/google-research/tree/master/single_view_mpi`

(a) Input panorama


(b) Predicted disparity without image padding

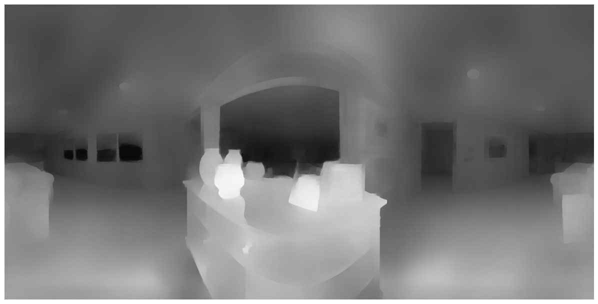

(c) Predicted disparity with 50% wrap padding

Figure 2: Adding horizontal wrap padding to the panorama eliminates discontinuities along the left and right edges of the image. Here the disparity map has been shifted so that the original edges are moved to the middle, to highlight the discontinuity when wrap padding is not applied. Image from Matterport 3D dataset [5].

3.3 Implementation Details

PanoSynthVR adopts the neural network model and pre-trained weights provided by Tucker and Snavely [28]. To prepare a panorama for input, we first apply horizontal wrap padding. Feeding this input through the model produces a set of 32 RGBα images representing the layers of the MCI, from which the excess padding is cropped. The depths of the layers are sampled uniformly in inverse depth from a depth of 1 to 100.

A disparity map can be extracted from the MCI as a per-pixel weighted average of the individual layer disparities. The weights are computed by accumulating the alpha values in the layers.

We computed the average inference time over 100 trials be 383.5 ms. We define inference as the time to produce the 32 MCI layers and the corresponding disparity map. The test was conducted using an NVIDIA V100 GPU. It is also possible to run the full pipeline on a standard laptop without GPU but processing time will increase (e.g on average 24.1 seconds on MacBook Pro (15-inch, 2016)).

We implemented a renderer in WebXR to support interactive rendering on both desktop, mobile phones, and VR headsets. We used

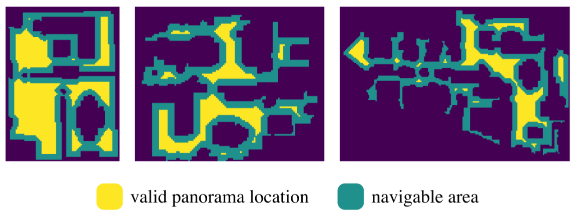


Figure 3: Examples of scene floor plans.

panoramas of size 2048×1024 in the WebXR viewer, as we found this to provide the best balance between resolution and rendering speed. The web-based renderer allows users to interact with scenes in real time. An example is provided with the source code in the supplementary material. We found that the renderer consistently produces 30 fps while the user is in motion on an Oculus Quest 2. Tethered headsets yield higher performance (60 fps via Oculus link).

4 Qualitative and Quantitative Results

We implemented a textured mesh renderer as a baseline for comparison. To create the textured mesh, we extract the estimated disparity map from our results and use it as a displacement map on a cylinder geometry. For fair comparison and to ensure real-time rendering, we restrict each representation to have the same number of vertices. While we could have used a different and possibly more accurate method to infer the depth map [30], this would have introduced a confound in our experiments, since we would be comparing the outputs of two different systems. We wanted to isolate the comparison between potential of the textured mesh and MCI representations for view synthesis. Using the disparity map produced by our method eliminated any differences due to the underlying scene representation, and focused the experiments on the differences between the textured mesh and layered image rendering. (Existing 360 depth map methods are also incompatible with our setup, since they require spherical panoramas and we use cylindrical panoramas.)

We provide, a qualitative comparison between the MCI and mesh renderers in Figure 4. In this comparison we test several indoor panoramas from the Matterport3D dataset [5] and an outdoor panorama from the web. The textured mesh result has noticeable artifacts such as texture "stretching" at the edges of objects. However, the multi-cylinder image (MCI) is more blurry than the textured mesh in some areas, which is an acknowledged issue in the single-view multi-plane image (MPI) network output [28].

4.1 Synthetic Data Generation

To evaluate the perceptual and quantitative efficacy of PanoSynthVR, we compare the similarity of multiple views from our multi-cylinder image renderer and the benchmark mesh renderer against the ground truth and the views captured from a mesh render at the same position. We sample the ground truth snapshot from the same virtual 3D scenes used to generate the panoramas. The mesh renderer maps the original panorama texture onto a depth-based mesh synthesized from the disparity map. We compare each of the three snapshots using the SSIM, PSNR, LPIPS and MIFD metrics.

We use Habitat Sim [27] to load and extract images from synthetic 3D environments. Then, we generate batches of high-quality examples by extracting panoramas and snapshots from the scene. The synthetic panoramas had a resolution of 2048×512.

We evaluate the model on 700 synthetically generated snapshots from 26 scenes in the Replica, Habitat-Matterport 3D, and Gibson datasets. These datasets are primarily composed of home interior scenes. We grid sample viable positions from a top-down floorplan of navigable areas to determine the optimal panorama placement. The mask undergoes four iterations of binary erosion before extraction in order to distance all sampled positions from any obstacle (Figure 3). At each position, the virtual camera captures a cylindrical panorama and disparity map by taking an image for each column of vertical pixels in the panorama map, averaging the center column of pixels, and stitching together these columns into a complete image. We pass each panorama through the PanoSynthVR layer generation model to generate the cylindrical layers for the MCI renderer.

Because the MPI network was trained on structure-from-motion (SfM) data, its output is only determined up to scale. Therefore, we need to determine the unknown scaling factor σ between the MCI output and the ground truth 3D scene before synthesizing views.

Let $\mathbf{P}_s$ be a set of known 3D points (x,y,d) in the scene, where x,y is the 2D location in the image and d is the depth of the point. Let $\hat{\mathbf{D}}_s$ be the disparity map extracted from the MCI layers. Tucker and Snavely [28] propose to find the scaling factor that minimizes the mean log-squared error between the predicted disparity $\hat{\mathbf{D}}_s$ and the point set $\mathbf{P}_s$:

$$\sigma_{\text{mean}} = \exp\left[\frac{1}{|\mathbf{P}_s|}\sum_{(x,y,d)\in\mathbf{P}_s}(\ln\hat{\mathbf{D}}_s(x,y) - \ln(d^{-1}))\right]. \quad (1)$$

However, the mean is not robust to outliers, and thus this estimate for the scaling factor is highly sensitive to errors and artifacts in the disparity maps. We filter out images from the mean and median datasets where the mean scaling estimate renders the scene too close to the camera or pushes the scene past the camera.

As a more robust alternative, we propose to use the median instead of the mean when estimating the scale factor:

$$\sigma_{\text{median}} = \exp\left[\text{median}_{(x,y,d)\in\mathbf{P}_s}(\ln\hat{\mathbf{D}}_s(x,y) - \ln(d^{-1}))\right]. \quad (2)$$

We perform the scale factor calculation with mean and median over all scenes in our dataset for comparison purposes. Then, we generate a series of offset poses by sampling a random uniform offset from the origin of each panorama. Two batches are created for each scene, one with snapshots at a fixed distance of 10cm from the panorama location, and the other at a fixed distance of 50cm. At each offset location, we generate a perspective image (a snapshot) in Habitat Sim to serve as the sources of truth for the render comparison. The MCI and mesh renderers use the corresponding pose vectors to get the predicted snapshots at each pose. Finally, we evaluate the predicted offset snapshots by comparing the quality of the outputs over several metrics.

4.2 Image Metrics

We compare the generated snapshots from the MCI model and mesh based model (Mesh) against the ground truth snapshots with four quantitative image similarity metrics: SSIM, PSNR, LPIPS, and MIFD. We present the results of these comparisons in Table 1 for σ_{mean} and Table 2 for σ_{median}.

SSIM and PSNR are metrics that are widely used for comparing images and compare statistics calculated directly from the pixel values. Learned Perceptual Image Patch Similarity (LPIPS) [33] is a metric that has gained popularity recently. It is a distance metric that works by comparing the feature maps extracted with a deep convolutional neural network from a pair of images.

Mean Image Feature Distance (MIFD) is a geometric metric similar to reprojection errors commonly used in image reconstruction. It works by measuring the average Euclidean distance between SIFT features that have been matched between a pair of images, with candidate matches being filtered with Lowe's ratio test. We only use measurements where there are at least ten matches to ensure robustness in the results. One caveat with this metric is that it does

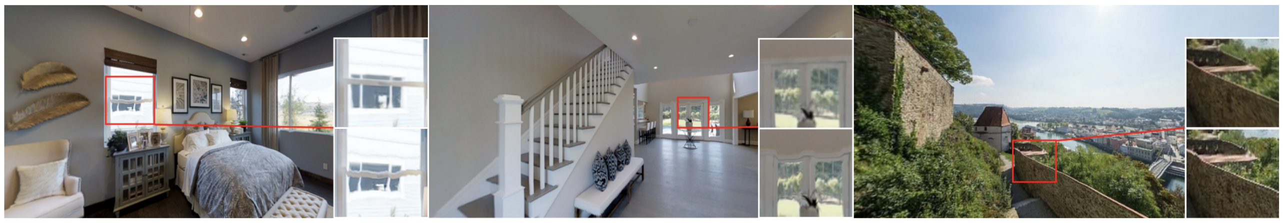


Figure 4: Comparison of a baseline mesh model (inset bottom) and our proposed MCI representation (inset top). First two images are from Matterport3D dataset [5]; Right image by Jürgen Matern (Creative Commons license).

Table 1: Image metrics by distance and method using σ_{mean}. Best results for a given distance are *italicized* and the best overall results are in **bold**.

		Higher is Better ↑		Lower is Better ↓	
Distance	Method	SSIM	PSNR	LPIPS	MIFD
10 cm	MCI	*0.725*	*22.4*	0.330	18.3
	Mesh	0.700	21.9	*0.281*	*11.9*
50 cm	MCI	*0.644*	*18.4*	0.477	58.9
	Mesh	0.594	17.1	*0.470*	*42.0*
All	MCI	**0.685**	**20.4**	0.404	38.6
	Mesh	0.647	19.5	**0.375**	**27.0**

Table 2: Image metrics by distance and method using σ_{median}.

		Higher is Better ↑		Lower is Better ↓	
Distance	Method	SSIM	PSNR	LPIPS	MIFD
10 cm	MCI	*0.804*	*25.2*	0.258	13.8
	Mesh	0.795	25.0	*0.192*	*8.8*
50 cm	MCI	*0.708*	*20.6*	0.402	46.5
	Mesh	0.688	19.9	*0.362*	*30.3*
All	MCI	**0.756**	**22.9**	0.330	30.1
	Mesh	0.742	22.5	**0.277**	**19.6**

not produce a score if there are zero matching features (due to either the two input images being too different or producing an insufficient number of SIFT features).

4.3 Discussion

Our model outperforms mesh renders for the SSIM and PSNR metrics, likely due to the blurring effect of the semitransparent layers in comparison to the sharp tesselations of the mesh. While the mesh renderer is highly sensitive to error in the disparity predictions between layers, the blending effects of the MCI provide a more robust representation that averages individual pixel values to a plausible output.

However, the mesh output outperforms the MCI output for both LPIPS and MIFD. In the case of LPIPS, the clarity of the mesh output likely produces a higher correlation between the actual and predicted image patches. The convolutions of the LPIPS model can find more similarities in the sharper images than the soft images. This is indicative of the limitations of the multi-cylinder image (MCI) approach, in which views get progressively blurrier as the virtual camera approaches the wall of the cylinder.

The texture captured at a single viewpoint cannot supply enough visual information to produce a convincing image at large offsets. When the user approaches the walls of the inner cylinder, the angle of incidence with each MCI cylinder is too large which causes spatially distinct features to blur together. This occurs in part because our datasets push the MCI representation past its recommended limits. For example, our 10cm dataset has a maximum disparity of 75 pixels, while the recommended disparity limit is 32 [18]. Additionally, the spaces between each cylinder become visible at the top and bottom of the viewport, leading to a banding effect. The experimental results reflect these issues, with the 50cm offset images having inferior scores to the test images in the 10cm offset dataset. Increasing the number of cylinders in the MCI would likely ameliorate these aberrations.

The accuracy improvements of using median over mean for calculating sigma scale differences between disparity maps is worth noting. The outliers in disparity differences may have a large effect on the effective representation of the scene. The use of median reduces this effect and produces more convincing scene imagery. Predicting the scale of a scene in an unsupervised environment poses a challenge that would likely improve with more accurate depth estimation trained specifically on cylindrical panoramas.

5 User Study

As we found conflicting results between SSIM/PSNR and LPIPS/MIFD, we were interested in measuring what these contrasting measurements mean for actual users. Thus we performed a user study to gain more insights. In particular, we were interested in how users perceive the difference between the MCI and the Depth-based renderings as the image metrics can only measure differences in the rendered images but not how this affects the users within an VR environment. Preliminary feedback from test users showed that users experienced less distortions in the MCI results compared to the textured mesh (Mesh): e.g. mentioning "less wrinkles in the window areas." For the majority of scenes, users rated MCI as their preferred option when comparing to the textured mesh. However, they also mentioned the reduced resolution of the MCI compared to the textured mesh.

5.1 Study Design

Based on the initial feedback, we designed a within-subject study to investigate if there are any effects on presence and perceived quality when using PanoSynthVR (MCI). Similar to the work by Serrano et al., we used a plain 360 panorama (Plain) as the reference condition [24]. In addition, we used a textured mesh with an estimated depth model (Depth) to investigate the difference in image metrics more in detail from a user perspective.

For this purpose, we postulated four hypotheses.

- H1: MCI creates a higher sense of presence compared to using a plain panorama or textured mesh (depth map).
- H2: MCI improves depth perception compared to using a plain 360 panorama.
- H3: MCI has a similar visual quality as a plain panorama (Plain) and shows less artifacts than using a textured mesh with the estimated depth map (Depth).

- H4: In a direct comparison, participants would prefer MCI over the textured mesh (Depth).

In the first part of the study, we use the IPQ, a questionnaire for analyzing presence [21], and additional questions about depth perception and quality. The dependent variables are the IPQ score, the perceived depth perception, the perceived visual quality and the perceived amount of artifacts. The independent variable is the rendering method with three conditions: Plain, Depth and MCI. In the second part of the study, we captured the preference of participants when directly comparing the Depth and the MCI condition using a two choice question “which one of the rendering options do you prefer?”

5.2 Apparatus

We use an Oculus Quest 2 that is connected to a desktop computer via Oculus Link. The rendering is done on the desktop computer in Chrome using a WebXR implementation. We decided to use PC-powered renderings via Oculus Link to maintain high rendering frame rates while being able to supervise the experiment, logging information (such as conditions and scenes shown) and receive feedback on what the participants are witnessing. The shown conditions and scenes are controlled by the study operator using the Oculus controllers.

5.3 Procedure

Participants were initially presented with a basic questionnaire for demographic purposes. These include non-identifiable properties such as gender, age, ethnicity, and whether they have prior VR experience. After ensuring the participants were comfortable in a VR headset, they started the first part of the experiment, where they then experienced a rendering from each of the rendering conditions (Plain, Depth, MCI) in randomized order (Latin square) for three different randomized scenes per condition. In total, participants experienced nine different VR scenes with the randomized conditions and were asked questions based on the IPQ presence questionnaire as well as the depth perception and visual quality of the scenes. Presence scores were obtained using the seven-point Likert scale for 14 items of the IPQ, and Quality scores were obtained using 3 questions about sense of depth, visual quality and the presence of visual artifacts using a 7 Likert Scale (Low to High).

In the second part of the experiment, participants were asked to observe the same (randomized) scene using two conditions: Depth and MCI and to perform a preference ranking. Participants were given as much time as they needed and were able to switch between the two conditions as often as they needed before making a decision.

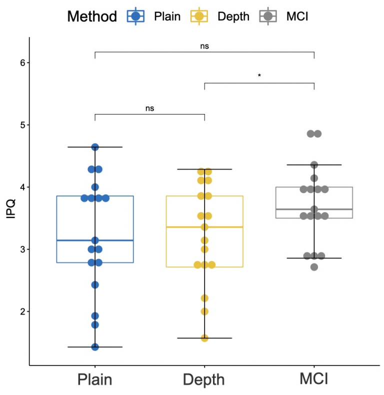


Figure 5: Overall Presence as captured by the IPQ questionnaire with significant difference between ”Depth” and ”MCI”.

The study was approved by the University of Otago Human Ethics Committee (reference number D21/103). Participants were assured that if they decide not to take part in the project that this would result into no disadvantage to themselves. They were also instructed that they could stop at any time if they felt uncomfortable or experienced motion sickness. No personally identifiable data is collected beyond those included in the demographic questionnaire, and every effort was be made to ensure that no data can be linked to any individual participant. We also ensured health and safety standards by cleaning and disinfecting the headset after each user and using disposable hygiene covers[3] in light of COVID-19.

5.4 Participants

We recruited participants via announcements in university lectures, flyers and word-to-mouth. We invited 18 participants (9 male, 9 female, mean age = 26.78 ± 7.65 years ranging from 20-45) to take part in our study. Each study took approximately half an hour. Participants received either a coffee voucher or a sweet or healthy snack as acknowledgment for their time. We had to exclude the results of one participant due to a technical issue that showed one of the three conditions twice and another condition not at all in the first part of the study. The issue was discovered when checking the log files after the experiment was finished.

5.5 Results

We analyzed the results for the IPQ and the ratings with regards to depth perception, visual quality and the presence of visual artifacts using R.

5.5.1 IPQ

We calculated the overall IPQ presence score [21–23] (Figure 5) as well as the IPQ subscale scores. Since the IPQ scores are captured on ordinal scales, we used non-parametric tests for statistical analysis (Friedman and Wilcoxon (holm) as posthoc test). According to the IPQ data analysis instructions[4] we converted the IPQ scales to a range from 0 - 6.

Descriptive statistics show that the Plain method (mean = 3.23, std = 0.95, median = 3.14) and Depth method (mean = 3.24, std = 0.82, median = 3.36) have similar medians for the overall presence score, while the median for MCI is larger (mean = 3.72, std = 0.64, median = 3.64). Using the Friedman test, we measured a statistically significant difference in the IPQ scores depending on method, $\chi^2(2) = 6.969$, p = 0.031. The result is significant at $p < .05$. We then used a Wilcoxon (with holm correction) for post-hoc analysis that indicated a significant difference between Depth and MCI (p = 0.044). This indicates that presence score is significantly higher for MCI compared to Depth. We did not measure a significant difference between Plain and MCI (p = 0.118) nor between Plain and Depth (p = 0.451). The Wilcoxon effect size r indicates small to large effect sizes (Plain-Depth r = 0.179 (small), Plain-MCI r = 0.471 (moderate) and Depth-MCI r= 0.597 (large)).

We also analyzed the IPQ subscales (General presence (G), Spatial presence (SP), Involvement (INV) and Realism (REAL)) for a more detailed analysis (Figure 6). We did not find a significant effect of rendering condition on G ($\chi^2(2) = 1.1915$, p-value = 0.551), nor on SP ($\chi^2(2) = 3.966$, p-value = 0.1376), nor on INV ($\chi^2(2) = 1.548$ p-value = 0.461), nor on REAL ($\chi^2(2) = 4.557$, p-value = 0.102).

5.5.2 Depth perception and Quality

For statistical analysis of depth perception and quality scores, we once again used non-parametric tests (Friedman and Wilcoxon (holm) as post hoc test) to work with Likert scales. While the

[3]`https://vrcover.com/about/`
[4]`http://www.igroup.org/pq/ipq/data.php`

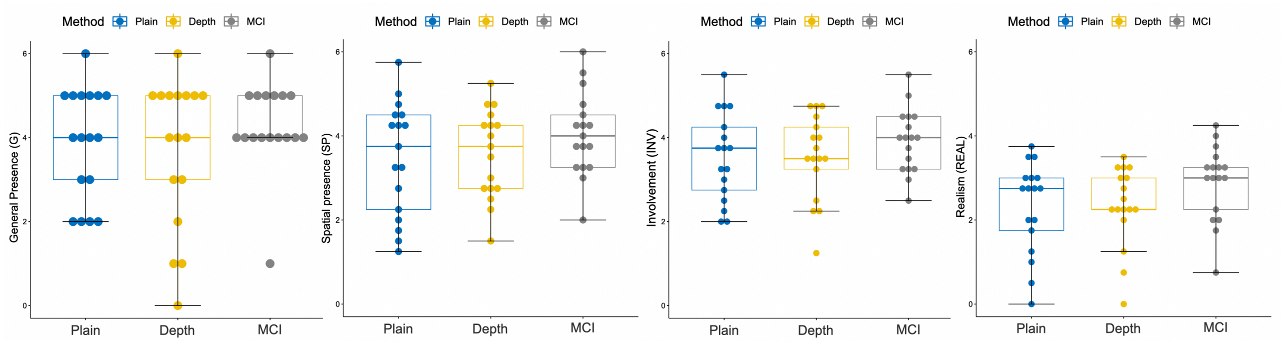


Figure 6: Presence subscales as captured by the IPQ questionnaire. General presence (G), Spatial presence (SP), Involvement (INV) and Realism (REAL)). We did not find any significant effects of rendering condition on any of the subscales.

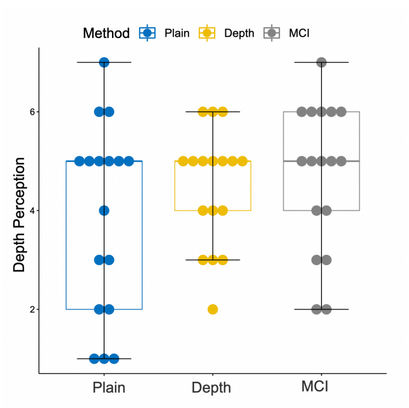


Figure 7: Rating of depth perception using a Likert scale (low (1) to high (7)).

ratings for the depth perception (Figure 7) were in average higher for the Depth and the MCI condition (Plain: mean = 3.88, std = 1.93, median = 5, Depth: mean = 4.47 std = 1.18, median = 5, MCI: mean = 4.71 std = 1.49, median = 5), we did not measure a significant effect of rendering mode on depth perception (Friedman chi-squared = 2.4151, df = 2, p-value = 0.299). Analysis of the Wilcoxon effect size r indicates a small effect size between Depth and MCI (r = 0.205) and moderate effect sizes between Plain and Depth (r = 0.312) and between Plain and MCI (r = 0.381).

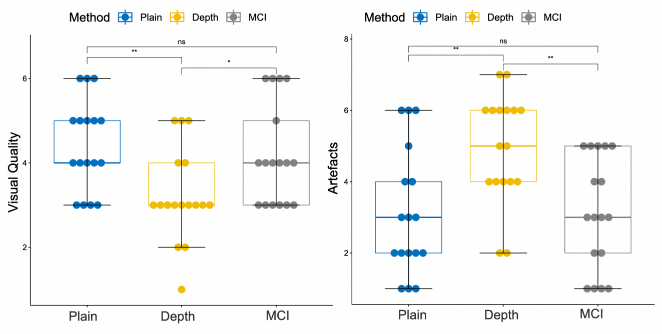


Figure 8: Likert scale results (low (1) to high (7)) for visual quality and amount of visual artifacts. Left) Visual quality. Right) Rating on amount of visual artifacts. We found significant effects of rendering method on visual quality and and visual artifacts.

For the visual quality (Figure 8, Left), we found a significant effect of rendering mode on visual quality (chi-squared = 13.451, df = 2, p-value = 0.001). Post hoc analysis showed a significant difference (p=0.006) between Plain (mean = 4.41 std = 1.06) and Depth (mean = 3.24, std = 1.09) with a decrease in visual quality for Depth. We could not find a significant difference between Plain and MCI (mean = 4.18, std = 1.19) (p = 0.4) nor between Depth and MCI (p = 0.077). The Wilcoxon effect size r indicates small to large effect sizes (Plain-Depth: r = 0.812 (large), Plain-MCI: r = 0.187 (small) and Depth-MCI: r = 0.555 (large)).

Questions regarding the amount of artifacts (Figure 8, Right) indicated a significant effect of rendering mode (Friedman chi-squared = 10.561, df = 2, p-value = 0.005). Post hoc testing with Wilcoxon (holm correction) indicated significant difference (p = 0.013) between Plain (mean = 3.12, std = 1.76) and Depth (mean = 4.94, std = 1.52), as well as a significant difference (p = 0.0089) between Depth and MCI (mean = 3.12, std = 1.58) with Depth showing a significant higher amount of visible artifacts. We could not find a significant difference between Plain and MPI (p = 0.94). The Wilcoxon effect size r indicates small to large effect sizes (Plain-Depth r = 0.702 (large), Plain-MCI r=0.00585 (small) and Depth-MCI r = 0.723 (large)).

5.5.3 Preference

The results for the preference ranking show that the majority of participants prefer the MCI option (88% vs 12%) in a direct comparison. For analyzing the results, we use a binomial test. The results of the binomial test indicate the preference for MCI is statistically significant ($p < 0.001$).

5.6 Discussion

Within our study we were able to show that our method provides a significantly higher overall presence compared to a textured mesh (Depth) partly confirming our hypothesis **H1**. It is interesting to note that the standard deviation of the MCI condition was smaller compared to the other conditions indicating that participants have less variation in their opinions on the presence questionnaires. While we did not find a significant difference between Plain and MCI, the moderate effect size between both indicates that there might an effect that was not detected with our sample size and requires further studies.

While the ratings for the depth perception were in average highest for the MCI condition, we did not measure a significant effect of rendering method on depth perception. Thus we need to reject **H2**. However, we recommend further study on this aspect as moderate effect sizes as computed by Wilcoxon r indicate that there might be an effect, but our sample size was too small.

Our results showed that there was a significant difference in visual quality between Plain and Depth, but we could not find a signifi-

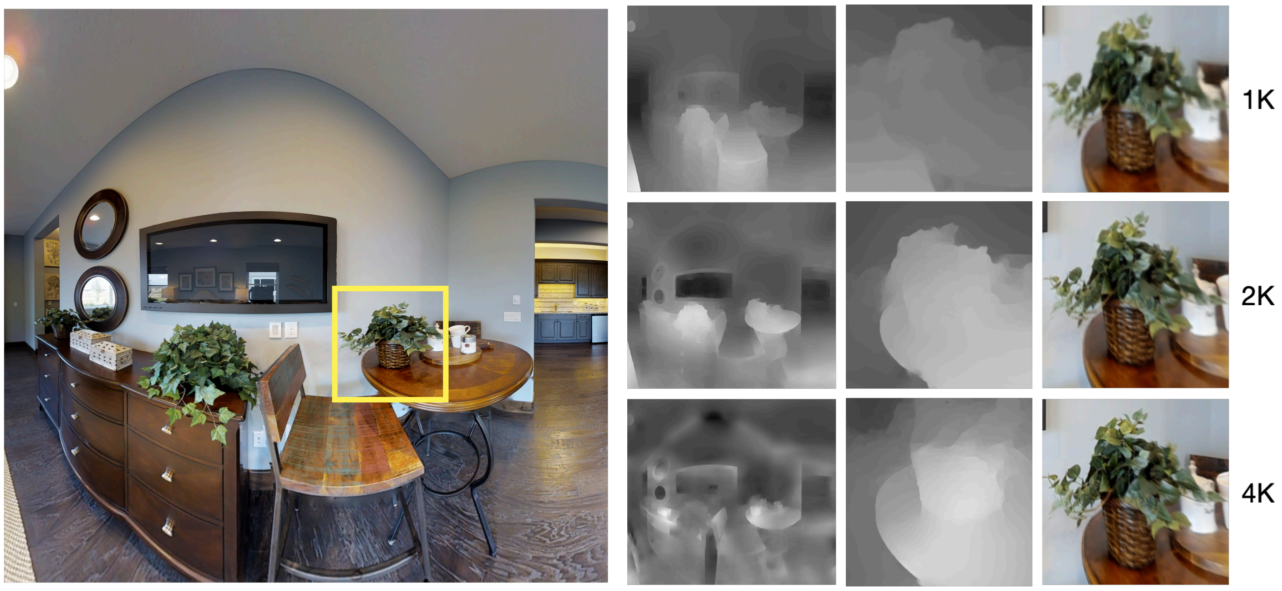


Figure 9: Comparison of generated depth maps and view synthesis at various resolutions. At lower resolutions, the depth map lacks detail, but at higher resolutions, the depth map lacks global coherence and artifacts are introduced. *Left*: Input image (cropped to fit page) with detail region outlined in yellow. *Right*: Generated depth maps, depth map detail, and view synthesis detail, at 512 × 1024 (*top*), 1024 × 2048 (*middle*), and 2048 × 4096 (*bottom*) input image resolution. Image from Matterport3D dataset [5].

cant difference between Plain and MCI. There was also just a small effect size between Plain and MCI. These results suggest that we can achieve a similar visual quality as the plain panorama while still providing a depth effect. Participants also found significantly fewer artifacts in MCI and in Plain than in the mesh renderer (Depth) confirming our hypothesis **H3**. The soft MCI layer approach maintains the structure of complex objects in the image, while mesh rendering struggles with curved and multifaceted surfaces. The MCI system scores comparably with the plain texture renderer, showing that the MCI model effectively maintains the quality of the image around the panoramic origin point.

Test subjects also reported shortcomings of the MCI model, primarily that the environment appeared to have lower resolution than that of the mesh and plain renders. When a user approaches the innermost cylinder, the layered representation lacks sufficient data to provide an accurate representation of the scene. This reduced the overall realism and was likely a factor in our model's score distribution for that category. Higher resolution textures or additional intermediary layers may allow the MCI model to handle large offsets gracefully.

The second part of our study showed that the participants preferred the MCI condition when directly comparing it with the Depth condition for the same scene. This confirms our fourth hypothesis **H4** and shows that participants see an advantage of using MCI.

6 Conclusions and Future Work

In this work, we introduced the MCI representation for real-time, immersive free-viewpoint view synthesis rendering, and demonstrated an initial pipeline to extract an MCI from a single input panorama. We developed an evaluation pipeline which produced synthetically generated panoramas as input to our MCI model and mesh renderer. The results provided the materials for our quantitative analysis, in which our MCI renderer outperformed a textured mesh in terms of the standard PSNR and SSIM metrics. However, our method was less effective in terms of perceptual metrics, likely due to occasional blurriness and ghosting in the renderings. Users of our system were more distracted by the visual artifacts in the mesh renderings than blurriness in the MCI renderings. This could indicate that the perceptual image metrics are not able to capture these differences effectively and it would be interesting to investigate these aspects more in detail in future work. Finally, we conducted a user study that showed that our proposed technique achieves higher levels of presence and visual quality compared to a textured mesh. As Serrano et al. highlighted in their work [24], there are many studies that assess presence when exploring synthetic virtual reality content, but there is only limited research on the experience of 360 footage in VR headset. We hope that we contribute to this research direction with our study and advance the field with our findings.

Because of the limited number of layers in the MCI, our current model works best when the user is less than 50cm from the center of the scene. Thus our current model is best-suited for a standing or sitting user who is looking around a scene but not walking around in it. While this lowers the barrier to entry for creating an immersive scenes, users are still limited to a small radius of movement. Future versions of our technique with more MCI layers and higher-resolution imagery would allow for a larger radius of travel. Adding more layers does come at the cost of performance, the rendered image is a composite of all layers in the MCI.

While we applied an existing network trained on perspective images, in future work we plan to explore training or fine-tuning the network on cylindrical images, to achieve more accurate and higher-resolution output. As can be seen in Figure 9, our current neural network works best at a small image resolution, similar to the images it was trained on, but higher resolutions are necessary to support comfortable viewing in a VR headset.

Acknowledgments

This work was partially supported by National Science Foundation Awards 1924008 and 2144822 and the New Zealand Marsden Council through Grant UOO1724.

References

[1] B. Attal, S. Ling, A. Gokaslan, C. Richardt, and J. Tompkin. MatryODShka: Real-time 6DoF video view synthesis using multi-sphere images. In *European Conference on Computer Vision (ECCV)*, Aug. 2020.

[2] T. Bertel, N. D. Campbell, and C. Richardt. Megaparallax: Casual 360 panoramas with motion parallax. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):1828–1835, 2019.

[3] T. Bertel, M. Yuan, R. Lindroos, and C. Richardt. Omniphotos: Casual 360° VR photography. *ACM Transactions on Graphics (TOG)*, 39(6):1–12, 2020.

[4] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. DuVall, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 39(4):86:1–86:15, 2020.

[5] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[6] P. Debevec, Y. Yu, and G. Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Workshop on Rendering Techniques*, pp. 105–116. Springer, 1998.

[7] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 11–20, 1996.

[8] P. Hedman, S. Alsisan, R. Szeliski, and J. Kopf. Casual 3D photography. *ACM Transactions on Graphics (TOG)*, 36(6):1–15, 2017.

[9] P. Hedman and J. Kopf. Instant 3d photography. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[10] J. Huang, Z. Chen, D. Ceylan, and H. Jin. 6-DOF VR videos with a single 360-camera. In *2017 IEEE Virtual Reality (VR)*, pp. 37–44. IEEE, 2017.

[11] V. Jampani, H. Chang, K. Sargent, A. Kar, R. Tucker, M. Krainin, D. Kaeser, W. T. Freeman, D. Salesin, B. Curless, et al. Slide: Single image 3d photography with soft layering and depth-aware inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12518–12527, 2021.

[12] L. Jin, Y. Xu, J. Zheng, J. Zhang, R. Tang, S. Xu, J. Yu, and S. Gao. Geometric structure based and regularized depth estimation from 360 indoor imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 889–898, 2020.

[13] J. Kopf, K. Matzen, S. Alsisan, O. Quigley, F. Ge, Y. Chong, J. Patterson, J.-M. Frahm, S. Wu, M. Yu, et al. One shot 3D photography. *ACM Transactions on Graphics (TOG)*, 39(4):76–1, 2020.

[14] J. Li, Z. Feng, Q. She, H. Ding, C. Wang, and G. H. Lee. MINE: Towards continuous depth MPI with NeRF for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12578–12588, October 2021.

[15] Q. Li and N. K. Kalantari. Synthesizing light field from a single image with variable mpi and two network fusion. *ACM Trans. Graph.*, 39(6):229–1, 2020.

[16] K.-E. Lin, Z. Xu, B. Mildenhall, P. P. Srinivasan, Y. Hold-Geoffroy, S. DiVerdi, Q. Sun, K. Sunkavalli, and R. Ramamoorthi. Deep multi depth panoramas for view synthesis. In *European Conference on Computer Vision*, pp. 328–344. Springer, 2020.

[17] B. Luo, F. Xu, C. Richardt, and J.-H. Yong. Parallax360: Stereoscopic 360 scene representation for head-motion parallax. *IEEE transactions on Visualization and Computer Graphics*, 24(4):1545–1553, 2018.

[18] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[19] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer vision*, pp. 405–421. Springer, 2020.

[20] S. Niklaus, L. Mai, J. Yang, and F. Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (ToG)*, 38(6):1–15, 2019.

[21] H. Regenbrecht and T. Schubert. Real and illusory interactions enhance presence in virtual environments. *Presence*, 11(4):425–434, 2002.

[22] T. Schubert, F. Friedmann, and H. Regenbrecht. The experience of presence: Factor analytic insights. *Presence: Teleoperators & Virtual Environments*, 10(3):266–281, 2001.

[23] V. Schwind, P. Knierim, N. Haas, and N. Henze. Using presence questionnaires in virtual reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–12. Association for Computing Machinery, New York, NY, USA, 2019.

[24] A. Serrano, I. Kim, Z. Chen, S. DiVerdi, D. Gutierrez, A. Hertzmann, and B. Masia. Motion parallax for 360 RGBD video. *IEEE Transactions on Visualization and Computer Graphics*, 25(5):1817–1827, 2019.

[25] J. Shade, S. Gortler, L.-w. He, and R. Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 231–242, 1998.

[26] M.-L. Shih, S.-Y. Su, J. Kopf, and J.-B. Huang. 3D photography using context-aware layered depth inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8028–8038, 2020.

[27] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34, 2021.

[28] R. Tucker and N. Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 551–560, 2020.

[29] S. Tulsiani, R. Tucker, and N. Snavely. Layer-structured 3d scene inference via view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[30] F.-E. Wang, Y.-H. Yeh, M. Sun, W.-C. Chiu, and Y.-H. Tsai. Bifuse: Monocular 360 depth estimation via bi-projection fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 462–471, 2020.

[31] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7467–7477, 2020.

[32] A. Yu, V. Ye, M. Tancik, and A. Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4578–4587, June 2021.

[33] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

[34] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, (65):1 – 12, 2018.

[35] N. Zioulis, A. Karakottas, D. Zarpalas, F. Alvarez, and P. Daras. Spherical view synthesis for self-supervised 360 depth estimation. In *2019 International Conference on 3D Vision (3DV)*, pp. 690–699. IEEE, 2019.

[36] N. Zioulis, A. Karakottas, D. Zarpalas, and P. Daras. Omnidepth: Dense depth estimation for indoors spherical panoramas. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 448–465, 2018.