Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Deep synthesis and exploration of omnidirectional stereoscopic environments from a single surround-view panoramic image

Giovanni Pintore[a,b,*], Alberto Jaspe-Villanueva[c], Markus Hadwiger[c], Jens Schneider[d], Marco Agus[d], Fabio Marton[a,b], Fabio Bettio[a], Enrico Gobbetti[a,b,*]

[a] *CRS4, Cagliari, Italy*
[b] *National Research Center in High Performance Computing, Big Data and Quantum Computing, Italy*
[c] *VCC, KAUST, Thuwal, Saudi Arabia*
[d] *College of Science and Engineering, Hamad Bin Khalifa University, Education City, Doha, Qatar*

ARTICLE INFO

Article history:
Received 18 December 2023
Received in final form 15 February 2024
Accepted 9 March 2024

Keywords: Indoor Environments, Omnidirectional Images, Data-driven Methods, Immersive Stereoscopic Exploration, WebXR

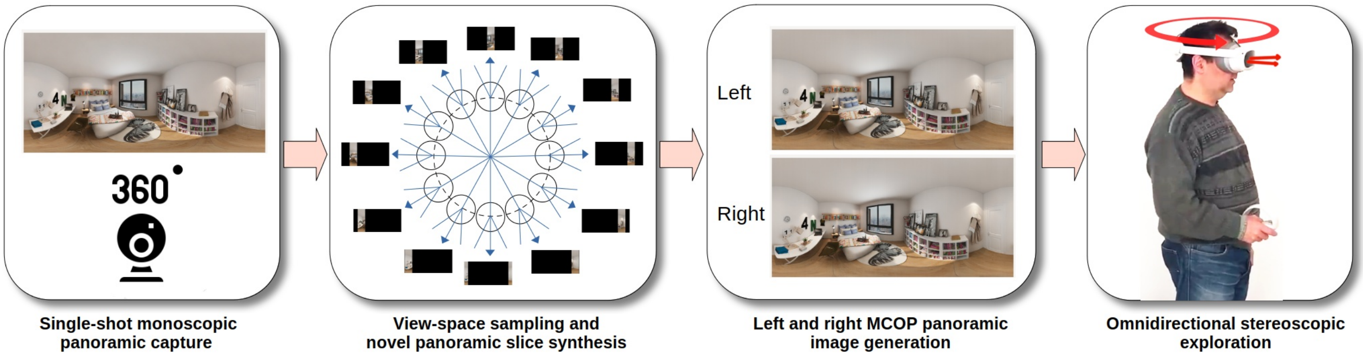
ABSTRACT

We introduce an innovative approach to automatically generate and explore immersive stereoscopic indoor environments derived from a single monoscopic panoramic image in an equirectangular format. Once per 360° shot, we estimate the per-pixel depth using a gated deep network architecture. Subsequently, we synthesize a collection of panoramic slices through reprojection and view-synthesis employing deep learning. These slices are distributed around the central viewpoint, with each slice's projection center placed on the circular path covered by the eyes during a head rotation. Furthermore, each slice encompasses an angular extent sufficient to accommodate the potential gaze directions of both the left and right eye and to provide context for reconstruction. For fast display, a stereoscopic multiple-center-of-projection stereo pair in equirectangular format is composed by suitably blending the precomputed slices. At run-time, the pair is loaded in a lightweight WebXR viewer that responds to head rotations, offering both motion and stereo cues. The approach combines and extends state-of-the-art data-driven techniques, incorporating several innovations. Notably, a gated architecture is introduced for panoramic monocular depth estimation. Leveraging the predicted depth, the same gated architecture is then applied to the re-projection of visible pixels, facilitating the inpainting of occluded and disoccluded regions by incorporating a mixed Generative Adversarial Network (GAN). The resulting system works on a variety of available VR headsets and can serve as a base component for immersive applications. We demonstrate our technology on several indoor scenes from publicly available data.

1. Introduction

Spherical cameras, also known as 360°, *panoramic*, or *omnidirectional*, or *surround-view* cameras, provide cost-effective and efficient solutions for rapidly capturing in a single shot the full context around the viewer of an entire environment [1]. A single panoramic image encompasses the complete scene visible from a specific viewpoint within a 360° field of view at a given instant.

*Corresponding authors
e-mail: giovanni.pintore@crs4.it (G. Pintore), alberto.jaspe@kaust.edu.sa (A. Jaspe), markus.hadwiger@kaust.edu.sa (M. Hadwiger), jeschneider@hbku.edu.qa (J. Schneider), MAgus@hbku.edu.qa> (M. Agus), fabio.marton@crs4.it (F. Marton), fabio.bettio@crs4.it (F. Bettio), enrico.gobbetti@crs4.it (E. Gobbetti)



Single-shot monoscopic panoramic capture

View-space sampling and novel panoramic slice synthesis

Left and right MCOP panoramic image generation

Omnidirectional stereoscopic exploration

Fig. 1: **Overview**. Taking as input a single panoramic image, a data-driven architecture synthesizes a comprehensive coverage of the scene's portion visible to both eyes during head rotation, encoding the views in the form of panoramic slices. The slices are then combined into an omnidirectional stereo representation composed of two multiple-center-of-projection (MCOP) images, tuned for the left and right eye. A lightweight WebXR viewer presents the suitable portions of these images on an HMD, responding to rotational head motions and delivering both stereo and motion parallax.

When experienced through a Head-Mounted Display (HMD), users dynamically explore this image by directing their attention to the desired content through head movements, leading to Virtual/Augmented/Extended Reality (VR/AR/XR) experiences with a natural interface and good degree of immersion [2].

For these reasons, omnidirectional imagery is increasingly recognized as a foundational element for generating immersive content from real-world scenes and for supporting a variety of VR/AR/XR applications [3]. Notably, 360° virtual tours have gained widespread popularity in the real estate sector [4]. Furthermore, omnidirectional images are easily shareable across various devices and platforms, making them highly versatile and accessible. Since they can be seamlessly integrated into websites, VR/AR/XR applications, or mobile platforms, they enable a broad audience to engage with indoor environments irrespective of their location or their available equipment [1]. Serving as representations of the user's surroundings, panoramic images also promise to be one of the essential building blocks for the construction of the shared physical and digital realities envisioned by the Metaverse concept [5].

Even though capturing a single shot panorama is a very appealing way to create a virtual clone of a real environment, the limitation of presented content to what was visible around the fixed location from which the panorama was taken leads to the loss of binocular stereo, which is very important to provide a sense of presence [6]. The fact that panoramas appear flat is a particularly strong limitation in indoor environments, given the relatively short distance from the viewer to the architectural surfaces and the objects. To provide stereo cues for full 360-degree rotations, views from a continuous set of shifted viewpoints must be available to the renderer. Omnidirectional stereo techniques [7, 8] are employed for that purpose but require the creation of stereo panoramas using cameras moving on a circular path [9, 8, 7] or multiple synchronized 360 cameras [3]. These acquisition approaches, however, reduce the possibility of quickly capturing, experiencing, and sharing a 360° scene using consumer hardware. In particular, while several low-cost cameras are widely available for monocular 360° capture (e.g., GoPro, Ricoh Theta, LadyBug, or Insta360), also due to the booming ”action-camera” market, stereo 360° solutions (e.g., Vuze+) are more costly and limited, and also typically offer only a low number (i.e., six to eight) of different point of views, leading to stereo and stitching artifacts. Moreover, while rotating camera solutions provide more viewpoints, they do not share the same simplicity and flexibility of single-shot instantaneous capture. For this reason, research has concentrated on view synthesis methods that generate stereo contents from a single 360° panorama. However, current methods either require complicated representations or are too heavy to run directly on HMDs and interactive rates (Sec. 2).

To overcome these limitations, we propose in this paper a novel approach for quickly and automatically generating and experiencing an omnidirectional stereo representation of an indoor environment starting from a single monoscopic panoramic image in an equirectangular format. In our approach, summarized in Fig. 1 and Sec. 3, we start by estimating full-frame per-pixel depth using a gated deep network designed to exploit interior environment constraints and trained on large sets of synthetic examples (Sec. 4). Then, we synthesize panoramic slices through reprojection and view-synthesis using a deep network that shares the same design features and training set of the depth estimation one (Sec. 5). These slices are placed around the central viewpoint, on the circle formed by the two eyes during head rotations, and cover an angular portion sufficient to accommodate the potential gaze directions of both the left and right eye. A stereoscopic multiple-center-of-projection stereo pair in equirectangular format is then composed by suitably blending the precomputed slices. The resulting pair is loaded into a WebXR viewer for a lightweight, responsive experience with both motion and stereo cues during runtime (Sec. 6). In this approach, based on approximating a full stereo experience through an omnidirectional stereo pair (see Sec. 2), the run-time costs are minimized, both in terms of storage and bandwidth and in terms of rendering performance, at the cost of a slight degradation of stereo reconstruction in the peripheral vision (see Sec. 6).

Our main contributions are the following:

- we introduce a novel end-to-end deep network architecture that generates shifted views of an indoor panoramic image in equirectangular format; a first network module estimates a depth map from a single panoramic input; then, these views are reprojected to the desired position, and a full image is synthesized through a second network capable to generate plausible content in disoccluded areas. Unlike other state-of-the-art approaches in the literature [10, 11], the network is based on a lightweight gated architecture and a dilated bottleneck; as a result, we ensure scalability to larger image sizes and/or embedded hardware, while maintaining maximum visual detail when re-projecting onto new views;

- we introduce a unified network architecture with custom training strategies for both depth estimation and view synthesis. The same lightweight network is exploited for both tasks, just adapting the final activation function and changing the training mode. To this end, we introduce a specific photometric loss for novel view synthesis, combined with a GAN approach. As a result, photorealistic novel views are generated with a low computational cost. We moreover use super-resolution GAN-based architectures to increase further the resolution between the stereo images [12].

- we exploit our depth estimation, reprojection, and synthesis approach to generate a set of panoramic slices and use them to compute an omnidirectional stereo image pair that can be directly experienced on WebXR viewers that sample them to generate stereo couples that respond to head motion with low-latency and high frequency. The limitation to panoramic slices greatly simplifies off-line computational costs in comparison with previous solutions [13], and the direct exploitation of standard omnidirectional stereo formats fosters the applicability of the method to a variety of hardware and software platforms.

This article is an invited extended version of our ACM Web3D 2023 contribution [13]. In addition to providing a much more thorough exposition, we introduce very significant new material, in particular concerning a full redesign of the view sampling aspects, the exploitation of panoramic slices for the construction of a much more effective view synthesis network, and the off-line computation of omnidirectional stereo panorama in place of the run-time blending of few stereo couples.

Our evaluation (Sec. 7) illustrates how depth inference and inpainting networks achieve state-of-the-art performance and how they can be exploited to produce seamless omnidirectional stereo images at a high angular sampling rate. Since the proposed framework is easy to integrate into current panoramic viewers, just replacing the current monoscopic renderers, it promises to be a practical building block for delivering engaging and realistic experiences that captivate audiences and enable them to virtually explore and interact with indoor spaces in current and future Metaverse applications.

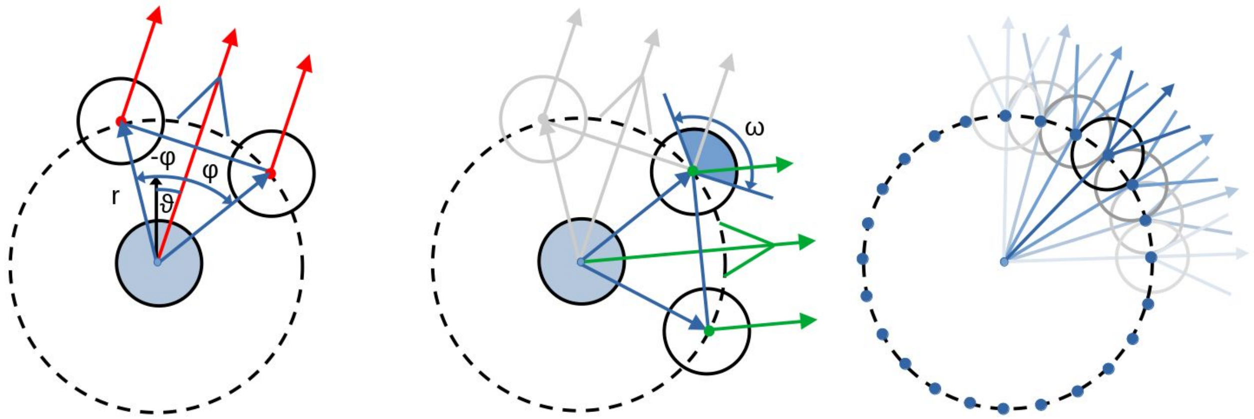
2. Related work

Our research focuses on creating immersive content using as sole input a single monoscopic panoramic image captured within an interior setting. The presentation of an image with stereo-parallax effects requires synthesizing different views for the left and the right eye. These views should respond to user motion by taking into account that the visibility of scene elements may change even for small shifts of the eye position. This requires not only implicit or explicit geometry estimation to take into account depth-dependent stereo parallax but also the handling of occlusions and disocclusions. In the subsequent discussion, we provide a concise overview of only the most closely-related works. We direct the reader to recent surveys on indoor reconstruction [14], scene understanding from panoramic imaging [15], as well as extraction of 3D geometry from 360° imagery [16] for a more comprehensive coverage of the subject matter.

Depth estimation from a single panorama. State-of-the-art monocular depth estimation solutions involve the adoption of data-driven approaches that extrapolate implicit relationships from extensive labeled datasets, incorporating priors tailored to specific applications, particularly within interior environments [14]. Prior studies have demonstrated that the direct application of perspective methods to 360° depth estimation in indoor settings yields suboptimal outcomes [17]. For this reason, ongoing research directly exploits the wide geometric context inherent in omnidirectional images while addressing wraparounds and distortions characteristic of equirectangular projections [18, 19, 20, 17, 21, 11, 22, 23]. Following this trend, our work introduces a streamlined and lightweight pipeline directly working on an equirectangular image, introducing an architecture that we also exploit for the view synthesis network.

Novel view synthesis. A panoramic image with an accompanying depth map can be utilized for view synthesis using diverse approaches, such as directly rendering point clouds [24], generating and rendering view-independent meshes from depth maps [25, 26], or integrating and blending depth maps or generated meshes with multiple images or signals [9, 27]. Recently, end-to-end view synthesis networks have been proposed to generate shifted panoramic views at run time [28, 29]. While these networks excel at inferring immersive views within a limited volume around the viewer (e.g., 50cm), their computational demands preclude direct execution on embedded platforms. Consequently, Head-Mounted Displays (HMDs) are exclusively supported via remote rendering [29]. For stereo generation, Pintore et al. [13], proposed, instead, to generate a set of stereo pairs off-line and to perform rendering on the HMD starting from these inferred views through a simple interpolation method. Since per-frame generation is confined to stereo pairs, the complexity of view synthesis networks is significantly reduced compared to more general previous solutions for free-viewpoint synthesis [28, 29]. In this work, we further streamline the method by generating off-line a set of panoramic slices optimized for subsequent blending into an omnidirectional stereo panorama. As a result, we further reduce both the off-line computation and the on-line rendering costs.

View interpolation. The generation of novel views by interpolating images taken at nearby viewpoints has been widely researched, with effective solutions being proposed, even in the



(a)

(b)

(c)

Fig. 2: **Viewing geometry.** (a): we consider that the two eyes are positioned along a circle whose center is at the center of rotation of the head and their central gaze direction is aligned with the head rotation; their position is uniquely determined by the head radius r, the inter-pupillary distance IPD, and by the angular position of the head θ. (b): during head rotation, at a given position, there is an angular slice of size ω that contains both the right and the left gaze direction; the slice's angular size can be solely determined by r and the IPD. (c): to cover all potential gaze positions and directions, we compute all angular slices placed at closely spaced positions on the circle.

absence of a prior depth estimation step [30, 31]. However, end-to-end networks tackling this task face similar computational constraints as depth estimation, limiting their applicability to interactive-rate frame generation on Head-Mounted Displays (HMDs). An emerging approach for rapid novel viewpoint synthesis involves employing layered depth representations, associating each pixel with multiple depth values [32]. This methodology has been effectively expanded to operate with single panoramic images [33, 34], as well as to create light field videos through layered mesh representations [35]. For perspective views, multi-plane panoramas (MPI) have also been proposed as an output representation produced with convolutional neural networks [36, 37]. However, MPIs are limited to viewpoints that are close to the origin and degrade when the viewpoint moves further. To address this limitation, adaptive sampling schemes have been proposed [38]. The concept of capturing the scene at multiple fixed depths has been extended for panoramic imaging by considering different capturing proxies like multi-spherical images (MSI) [39] or multi-cylinder images (MCI) [6]. In contrast, our proposed framework synthesizes a discrete set of panoramic slices that cover the circular trajectory made by both eyes during head rotations and are oriented towards the main view directions. These images are subsequently blended to form an omnidirectional stereo pair comprised of two multiple-center-of-projection (MCOP) equirectangular images. Compared to the current state-of-the-art, our approach offers the advantage of being lightweight, both in terms of cost of inferring novel views, since they are constrained to small angular portions of the sphere, and for immersive exploration through WebXR viewers, since rendering has about the same cost of monoscopic viewing. Moreover, the solution is compatible with methods employed for conventional stereo panoramas captured with moving rigs [40]. It should be noted that our view synthesis machinery would also be compatible with a run-time presentation of slices sampled by taking into account per-frame precise per-pixel view directions across the entire field of view. For general head displacements, Pintore et al. [29], for instance, proposed to generate novel panoramas on demand on a server in response to head motion changes. The solution, however, can only update panoramas at around 10Hz and with a latency of about 0.1s. Despite our faster networks, the expected speed-up is less than a factor of two, and, in any case, we would still require a high-speed connection to a fast rendering server. Since we only need to respond to rotation, an alternative solution would be to upload the entire set of slices to cover all possible eye directions. Using the same sampling rate employed in this paper would require, however, the uploading of 360 images, increasing bandwidth and storage requirements by over two orders of magnitude. However, using a lower sampling rate would increase ghosting artifacts [13] when employing simple blending or would require more complex precomputations and blending operations [41].

Omnidirectional stereo display. While 360° surround-view panoramas are limited to only the three rotational degrees of freedom, with the location being fixed, stereo presentation on HMDs requires different images for the left and right eyes to provide the stereo depth cue. Omnidirectional stereo projection, used in this work, is a multiperspective technique [42] based on circular projection stereo [7] that aims to combine in a single representation all the information required for stereo. For viewing, each vertical column of an equirectangular image has a different center of projection, corresponding to the position of the eye viewing it. By generating an image for the left eye and another one for the right eye, stereo is achieved. However, when viewing such an image in VR, stereo is only correct at the center of the image and degrades for peripheral vision. For this reason, recent work has concentrated on generating images that dynamically adapt to the user's gaze, in particular through the view-dependent rendering of depth images [43]. Our solution could also be adapted to those methods, given our capability to infer good depth maps. However, using plain omnidirectional stereo-pairs remains an appealing approach for indoor environments viewed on HMDs, since degradation mostly appears at the poles of the equirectangular image, which generally do not contain suitable content due to typical indoor environment shape and capture constraints, and in the peripheral vision, that also incurs in degradation due to foveation [44].

3. Method overview

Our method automatically and rapidly converts a single monoscopic panoramic image in equirectangular format into an omnidirectional stereo panorama, also in equirectangular format, that can be rapidly explored with stereo and motion parallax on an HMD.

As depicted in Fig. 2a, we consider that during exploration, the two eyes will be on a circular trajectory centered at the head's rotation center. Thus, their specific positions are defined by the head radius (r), the inter-pupillary distance (IPD), and the angular position of the head (θ). Without loss of generality, the central gaze direction of both eyes is considered in this paper aligned with the head rotation, even though the method can be easily adapted to other gaze directions (see Sec. 6).

Given this geometric configuration, during head rotation, any given position on the circle may thus become the center of projection for the left or the right eye. Thus, see Fig. 2b, from this point of view, there is a constant angular slice of size ω that is guaranteed to contain both the right and the left gaze direction. The angular size of such a slice can be solely determined by r and the IPD (see Sec. 6). As shown in Fig. 2c, it is thus sufficient to compute all angular slices placed at closely spaced positions on the circle.

We exploit this geometric configuration to define an efficient approach to synthesize all these views and combine them into an omnidirectional stereo panorama.

The first step of our method is to estimate the per-pixel depth of the input panoramic image that we assume is placed at the head center. This depth is computed in a single step by a gated deep network designed to exploit interior environment constraints and trained on large sets of synthetic examples, as detailed in Sec. 4.

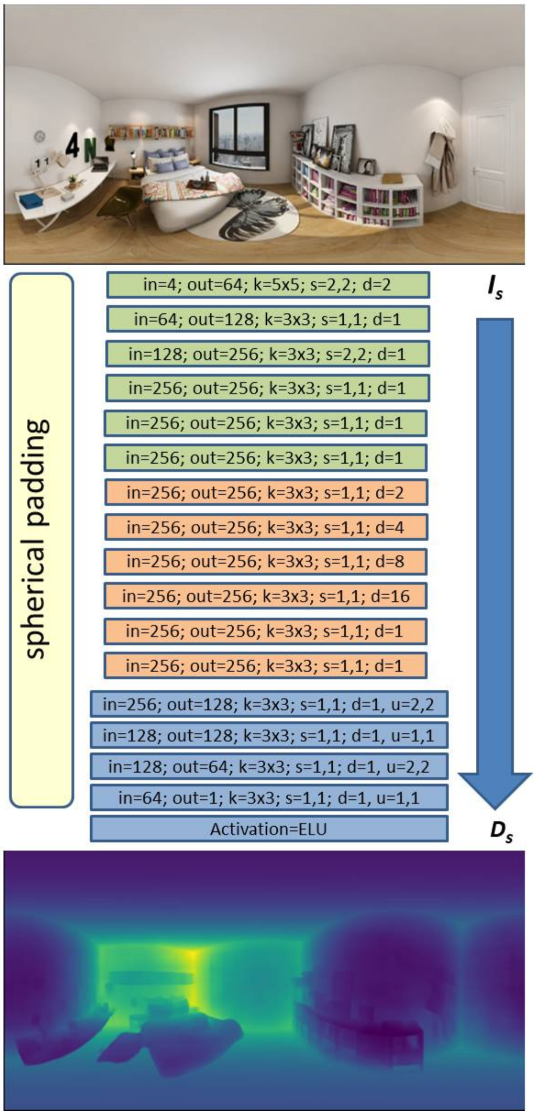
Given this depth and the original color, we synthesize each of the required shifted panoramic slices. For each of these slices, we start by reprojecting the original image into the required slice, defining a bounded vertical section of a panoramic image in an equirectangular format, using as a center of projection the relevant eye position. View-synthesis is performed using a deep network that shares the same design features and training set of the depth estimation one, as detailed in Sec. 5.

Finally, an omnidirectional stereoscopic image pair in equirectangular format is composed by suitably blending the precomputed slices and used for display in a lightweight WebXR viewer, producing images suitable for HMD consumption. (Sec. 6).

In the following, details are provided for each of the individual components.

4. Single panorama depth estimation

Augmenting a single image with depth is essential to establish the 3D position of visible points in space to compute their novel position when the viewpoint changes.



I_s
spherical padding
in=4; out=64; k=5x5; s=2,2; d=2
in=64; out=128; k=3x3; s=1,1; d=1
in=128; out=256; k=3x3; s=2,2; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=2
in=256; out=256; k=3x3; s=1,1; d=4
in=256; out=256; k=3x3; s=1,1; d=8
in=256; out=256; k=3x3; s=1,1; d=16
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=128; k=3x3; s=1,1; d=1, u=2,2
in=128; out=128; k=3x3; s=1,1; d=1, u=1,1
in=128; out=64; k=3x3; s=1,1; d=1, u=2,2
in=64; out=1; k=3x3; s=1,1; d=1, u=1,1
Activation=ELU
D_s

Fig. 3: **Panoramic depth estimation.** Legend: *in,out* channels; k convolution kernel; s stride; u upsample; d dilation. A gated architecture is used to predict depth from a single panoramic image. The network exploits gated-dilated convolutions for encoding and gated convolutions for decoding. It should be noted how this approach preserves details on the closest objects.

Many techniques have been documented in the literature to estimate depth from a single panoramic image (Sec. 2). Given the inherent ambiguity in depth estimation from single images, all approaches necessitate leveraging prior information to steer the reconstruction towards plausible architectural forms that align with the input. Notably, there has been a remarkable advancement in data-driven methods within this context, wherein these methods acquire knowledge of such priors from large collections of labeled exemplar data [11, 22]. Following this research trend, we designed a network for depth prediction that was an efficient compromise between accuracy and computational cost, and with an architecture that can be reused for the view synthesis part (Sec. 5). Using a lean and scalable network design is also important to support, in the future, larger and larger image sizesand to provide a low latency from the acquisition time to the presentation time, especially when using low-end machines.

To predict depth D_s from the source image I_s, we designed a gated architecture, illustrated in Fig. 3. The encoder-decoder scheme follows the same design adopted for view-synthesis (see Sec. 5), but with several differences to adapt it to the specific task of spherical depth estimation.

In particular, here, gating acts as a *self-attention weight mask*, differently from inpainting, where, instead, the mask is given as input to indicate the pixels to be inpainted (Sec. 5). Moreover, given the spherical nature of the input, we adopt circular padding along the horizon for convolutions, thus removing longitudinal boundary discontinuity and reflection padding to alleviate the singularities at the poles [45]. Furthermore, considering that our output will be a single channel, we use 32 internal channels instead of the default 64 channels in standard inpainting networks. Finally, since we produce depth, the last layer activation function is an *ELU*, instead of *tanh*.

The input equirectangular image, which is encoded through a sequence of light-weight gated convolutions having different strides (i.e., gated blocks in Fig. 3), so that the original size is reduced by a factor of four in each direction. Each encoding convolution is followed by instance normalization [46] and ReLU activation. Generally, compared to view-synthesis baselines [47, 48], our design has fewer parameters, with a lighter single branch, and it includes several solutions, described below, to improve accuracy for the depth estimation task and reduce computational complexity.

The adopted gated convolution (GC) approach [49] is expressed as:

$$\begin{aligned} G &= conv(W_g, I) \\ F &= conv(W_f, I) \\ O &= \sigma(G) \bigodot \psi(F) \end{aligned} \qquad (1)$$

where σ is the Sigmoid function, which outputs values in the range $[0, 1]$, ψ is an activation function (ReLU in our case), and W_g and W_f are two different sets of convolutional filters, which are used to compute the gates and features respectively. GC enables the network to learn a dynamic feature selection mechanism. In order to simplify training and guarantee low latency at inference time, our network uses a modified version of GC called Light Weight Gated Convolutions (LWGC), which reduces the number of parameters and processing time while maintaining the effectiveness [50]. Specifically, we decompose G from Equation 1 into a depth-wise convolution [50] (i.e., 3×3) followed by a 1×1 convolution, having, as a result, the same gating step but with only $k_h \times k_w \times C_{in} + C_{in} \times C_{out}$ parameters. Repeated dilations [51] are used for the bottleneck (see $d = 2, 4, 8, 16$ in Fig. 3), thus increasing the area that each layer can use as input. It should be noted that this is done without increasing the number of learnable weights but obtained by spreading the convolution kernel across the input map. The *dilated convolution operator* is then implemented as a gated convolution (i.e., Equation 1), but with some differences. It is expressed as:

$$D_{y,x} = \sigma(b + \sum_{i=-k'_h}^{k'_h} \sum_{j=-k'_w}^{k'_w} W_{k'_h+i,k'_w+j} \cdot I_{y+\eta i,x+\eta j}) \qquad (2)$$

where η is a dilation factor, $\sigma(\dot{})$ is a component-wise non-linear transfer function and $b \in \mathbb{R}^{C_{out}}$ is the layer bias vector. With $\eta = 1$, the equation becomes the standard convolution operation. In our model, we adopt, respectively, $\eta = 2, 4, 8, 16$ for the four bottleneck layers. Using this strategy, we aggregate multi-scale contextual information without losing resolution, thus capturing the global context efficiently by expanding the receptive field, avoiding additional parameters, and preventing information loss. This is important for both depth estimation and the image completion task (Sec. 5), as capturing sufficient context is critical for realism. By using dilated convolutions at lower resolutions, the model can effectively cover a larger area of the input image when computing each output pixel than with standard convolutional layers [47].

The network decoder (the four blue layers in Fig. 3), based on gated convolutions without dilation, restores the resolution of the output to the original input resolution.

The effectiveness of such a versatile baseline also depends on its training. In our approach, we adopt as a loss function for the depth prediction task the robust *Adaptive Reverse Huber Loss (BerHu)* [52], combined with a Structural Similarity Index Measure (SSIM), which measures the preservation of highly structured signals with strong neighborhood dependencies. As a result, such a panoramic depth prediction approach returns accurate depth maps for the input pose, as demonstrated by our results (Sec. 7).

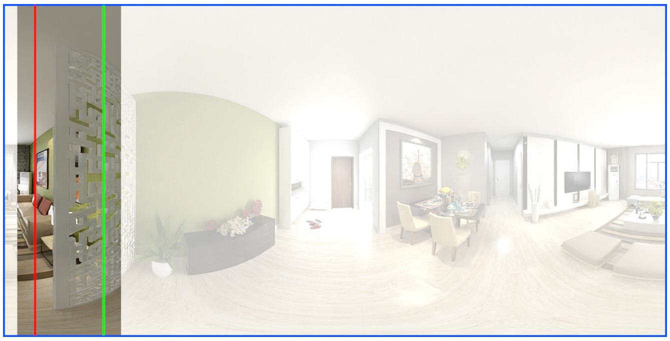


Fig. 4: **Panoramic slice** One of the slices produced by the network, placed at its position within the equirectangular image. The red line shows the area sampled by the right eye, while the green line shows the area sampled by the left eye (see Fig. 2b). In the background, we see the original panorama from the central viewpoint (note the large shift due to parallax effects).

5. Synthesis of novel views

Taking as input the original panoramic image and the registered panoramic depth map estimated by our deep network, this task aims to synthesize a collection of panoramic slices through re-projection and view synthesis. These slices are distributed around the central viewpoint, with each slice's projection center placed on the circular path covered by each eye during a head rotation. Furthermore, each slice encompasses an angular extent sufficient to accommodate the potential gaze directions of both the left and right eye (Sec. 6).

To this end, given a full angular extent of 180 degrees along the vertical direction and a limited viewpoint ω along the horizon, we generate novel spherical images in viewports $S_{\theta,\omega}$ (i.e., *slices*)

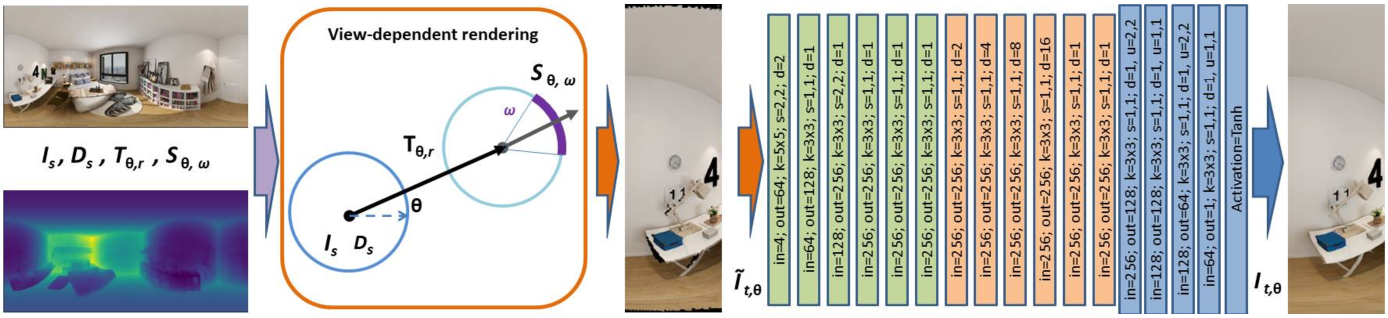

$I_s, D_s, T_{\theta,r}, S_{\theta,\omega}$
View-dependent rendering
$S_{\theta,\omega}$
ω
$T_{\theta,r}$
θ
I_s D_s
$\tilde{I}_{t,\theta}$
in=4; out=64; k=5x5; s=2,2; d=2
in=64; out=128; k=3x3; s=1,1; d=1
in=128; out=256; k=3x3; s=2,2; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=2
in=256; out=256; k=3x3; s=1,1; d=4
in=256; out=256; k=3x3; s=1,1; d=8
in=256; out=256; k=3x3; s=1,1; d=16
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=256; k=3x3; s=1,1; d=1
in=256; out=128; k=3x3; s=1,1; d=1, u=2,2
in=128; out=128; k=3x3; s=1,1; d=1, u=1,1
in=128; out=64; k=3x3; s=1,1; d=1, u=2,2
in=64; out=1; k=3x3; s=1,1; d=1, u=1,1
Activation=Tanh
$I_{t,\theta}$

Fig. 5: **Novel view synthesis.** Given the source image I_s and its predicted depth D_s, a new, sliced viewport $\widetilde{I_{t,\theta}}$ is rendered from a new viewpoint translated by $T_{\theta,r}$, according to a given direction θ and an offset r. The field-of-view ω, designed to cover both eyes' viewport (Sec. 6), is assumed constant. To generate an $I_{t,\theta}$ slice, we exploit the gated architecture already exploited for depth but adapted to have greater accuracy on the 3 RGB channels of the spherical section S. Legend: *in,out* channels; k convolution kernel; s stride; u upsample; d dilation.

along the circular path, by translating the original, central view I_s by an offset $T_{\theta,r}$ for each direction θ and distance r from the input view. For a typical human-size configuration, with a head radius of 100mm and an IPD of 65mm, we can safely assume 45 degrees as a portion of the image covering both eyes, which would correspond to 128 pixels for a 1024×512 equirectangular image. We further expand this region by ±5 degrees to provide context for reconstructing missing areas and to support eye convergence at a finite distance. For a 1024×512 image (corresponding to a 512×1024 tensor), each slice in our experiments is assumed to be 160 pixels wide, while the overall angle ω is about 56 degrees. An example of a slice produced by the network is depicted in Fig. 4.

The View-synthesis pipeline is depicted in Fig. 5 and includes two steps. The first is a view-dependent rendering step, which exploits the predicted depth D_s and translation $T_{\theta,r}$ to move pixel information to the new position. T_θ is given by polar coordinates, which depend on the head radius r (i.e., assuming in our experiment an average radius of 100mm) and by the angle θ, while $\widetilde{I_{t,\theta}}$ is the portion of translated pixels viewed from the viewport $S_{\theta,\omega}$. The second step consists in a view-synthesis deep network, which takes as input the translated pixels $\widetilde{I_{t,\theta}}$ and their disocclusion mask B_t (i.e, black pixels in Fig. 5), returning as output the novel viewport $I_{t,\theta}$. Fig. 6 shows, for two different scenes, a detail of a reprojected slice, with missing pixels in black, and the corresponding area of the output of the view-synthesis network.

In our case, by design, pixel rendering is not part of the learnable layers, and we assume we can directly project visible points according to D_s depth and T_θ translation using regular z-buffering to obtain the starting view to be optimized. This solution is better suited to our case than more elaborate splatting methods [53], since for stereo rendering, the limited displacement of the eyes from the center generates much narrower disocclusion zones than in the case of free viewpoint motion.

In an equirectangular image, columns correspond to constant longitude/azimuth θ angles, while rows to constant latitude/elevation ϕ angles. Each pixel can be mapped to angular spherical coordinates and vice-versa. This mapping between image domain pixels and spherical domain angular coordinates allows for direct transitions between image-based and spherical-based operations [54]. Omitting the straightforward relationship between Cartesian and spherical coordinates, the following equation relates spatial (i.e, $T_{\theta,r}$) with angular displacements (i.e., $\widetilde{I_{t,\theta}}$ pixels):

$$\begin{bmatrix} \partial d \\ \partial \phi \\ \partial \theta \end{bmatrix} = \begin{bmatrix} \sin(\phi)\sin(\theta) & \cos(\theta) & \cos(\phi)\sin(\theta) \\ \frac{\cos(\phi)}{d\sin(\theta)} & 0 & \frac{-\sin(\phi)}{d\sin(\theta)} \\ \frac{\sin(\phi)\cos(\theta)}{d} & \frac{-\sin(\theta)}{d} & \frac{\cos(\phi)\cos(\theta)}{d} \end{bmatrix} \begin{bmatrix} \partial x \\ \partial y \\ \partial z \end{bmatrix} \quad (3)$$

where d is the depth of the given pixel.

For the view-synthesis task, we assume $\widetilde{I_{t,\theta}}^{3\times h\times w}$ as input. As in typical inpainting approaches, we define a binary inpainting mask $B_t^{1\times h\times w}$, identifying missing parts in the rendered image. This mask is computed directly in the reprojection step. B_t is then concatenated to $\widetilde{I_t}$ (i.e., along the batch dimension - 4 layers input (Fig. 5).

To predict the output $I_{t,\theta}$ slice, we adopt the lightweight gated architecture exploited for depth estimation (Sec. 4) but adapted for having greater accuracy on the RGB channels of the current spherical viewport $S_{\theta,\omega}$. Here, spherical padding is replaced by replicate padding in all layers. Similarly to other works (e.g., DeepFillV2 [49]), we use $f(x) = \max(0, \tanh(x))$ as activation function for the output layer. Limited to the [0..1] range, this function behaves similarly to *ReLU* near the lower bound while smoothly saturating at the upper bound.

As shown in Fig. 5, this network has a higher density at the inner channel level, whose starting value is 64 (first encoder layer in Fig. 5). This increase in layers, compared to the configuration used for depth, is compensated, from the computational point of view, by the fact that the network processes a smaller portion of the image than the full equirectangular image, leading to a contained computational cost, as demonstrated in Sec. 7. This is particularly important since, for each input panorama, the generation of omnidirectional stereo representation requires the generation of hundreds of slices.

We train the inpainting network by including losses that measure the photorealistic quality of the output slice. It should be noted that, in contrast to full image prediction, here the loss is calculated by comparing the predicted slice $I_{t,\theta}$ with the corresponding crop $I_{gt,\theta}$ of the ground truth equirectangular image. Our loss function is expressed as:

$$\mathcal{L}_{vis} = \lambda_{px}\mathcal{L}_{px} + \lambda_{perc}\mathcal{L}_{perc} + \lambda_{style}\mathcal{L}_{style} + \lambda_{adv}\mathcal{L}_{adv} - \lambda_{lpips}\mathcal{L}_{lpips}. \quad (4)$$

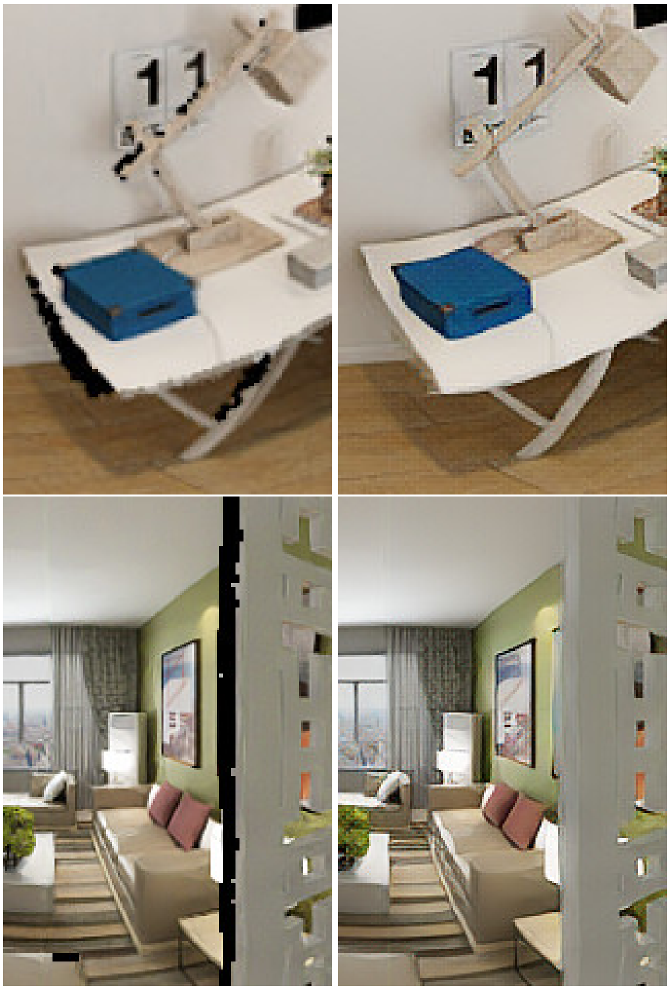


Fig. 6: **Reprojected vs. synthesized image** Two examples from different scenes. On the left, we see a detail of a reprojected image slice, where disoccluded areas are apparent. On the right, we see the output of the view synthesis network.

where the first term is a pixel-based $L1$ loss between the predicted RGB slice $I_{t,\theta}$ and the ground truth target crop $I_{gt,\theta}$, $\mathcal{L}_{perc}$ and $\mathcal{L}_{style}$ are the data-driven perceptual and style losses [55], enforcing I_{out} and I_{gt} to have a similar representation in the feature space as computed by a pre-trained $VGG-19$ [56], while $\mathcal{L}_{adv}$ is a discriminator-based loss (i.e., PatchGAN [57]). Furthermore, in addition to conventional inpainting losses, we introduce a loss based on Learned Perceptual Image Patch Similarity (LPIPS) [58] to enforce similarity due to the restricted field-of-view of the slice. λ weights are common for many single-pose inpainting problems [48]: $\lambda_{px} = 1.0$, $\lambda_{style} = 100.0$, $\lambda_{perc} = 1.0$, $\lambda_{adv} = 0.2$, $\lambda_{lpips} = 1.0$.

6. Omnidirectional stereo generation and rendering

Starting from the slices synthesized by the network, we have all the information to provide stereoscopic viewing during head rotations, as these slices contain a plausible scene reconstruction for all the possible points of view of both the left and the right eye. While previous works used a small set of these synthesized images and combined them at rendering time [13], here we densely sample the position space and construct off line a compact omnidirectional stereoscopic representation by appropriately fusing these slices. The issue of constructing stereoscopic panoramic image pairs has already been addressed in the literature (Sec. 2). In this work, we selected to achieve the stereoscopic effect by generating two aligned multiple-center-of-projection (MCOP) images encoded in equirectangular format. As longitude varies in these images, the corresponding pixel column is generated from a different camera position, which corresponds to the position of the eye when it is looking straight in this direction.

The calculation for each eye starts, thus, from the generation of n vertical slices radially uniformly distributed around the head, as shown in Fig. 2c. Since our networks are computationally efficient, and all the computation is performed offline, we can generate very dense angular samplings in a short time (ideally, even with n equal to the output image width). In practice, we have seen that an angular sampling of one degree (i.e., 360 slices) is sufficient to obtain a very high-quality reconstruction.

The single MCOP image for each eye is constructed from the union of vertical slices associated with each θ angle of longitude. The stereo effect is ensured by the fact that, for a given rotation θ of the head, the eyes will point in the same front-facing direction, but each with a slightly different perspective due to the offset caused by the inter-pupillary distance IPD. As seen in Fig. 2a, when we rotate the head around its vertical axis, for a given θ, the eyes will be positioned on the circle of radius r at angles $\theta - \phi$ (left eye) and $\theta + \phi$ (right eye), where the value of ϕ is given by $\phi = asin(\frac{IPD}{2r})$.

When computing the omnidirectional stereo representation for the left eye, we thus loop over all the output columns. For each vertical column at an angle θ, we identify the eye position as $\theta - \phi$ and find the two synthesized panoramic slices with the closest centers of projection (i.e., one to the left and one to the right). The pixels of these slices are then blended with a Gaussian weight based on the angular distance to the output column. The same process is done for the right eye, with the only difference being that the eye position is $\theta + \phi$. Since we are using a very high angular sampling rate in this paper to place slices, i.e., 360 slices per image, the blending area is extremely small – pixel-sized for our typical network outputs, and using such a simple blending does not lead to any noticeable ghosting artifacts, as illustrated in Sec. 7.

Note that while we have assumed here a view with both eyes looking in the same direction (zero parallax at infinity), we can apply the same approach for calculating MCOP images with eyes that have zero parallax at a finite distance for improved simulated stereoscopic vision in confined environments. The only variation would be in the extraction of the column, which would not be in the θ direction but towards the focal point.

As a final pre-processing step, we also perform upsampling of the images to match the quality of the display. This is because, currently, our synthesis is performed at a resolution smaller than the display size (i.e., a vertical slice resolution of 512 pixels vs 2048 for a typical headset). This limitation is not due to our easily scalable network architectures (see Sec. 4 and Sec. 5), but, rather, to limitations in available ground-truth training sets. In the current work, good quality results are obtained by applying available super-resolution generative adversarial networks capable of zooming images by creating plausible geometric and texture detail. To apply these methods to equirectangular images without boundary effects, we extend, before zooming, the original image to the left by incorporating portions from the right side and vice versa. This extension makes available to the network a sufficient context to define all important areas. After zooming, the image is then cropped only to contain the relevant portion of the equirectangular representation.

The two MCOP images resulting from this process can then be presented to the viewer using the same approach used for regular panoramas, presenting the left panorama to the left eye and the right panorama to the right eye, and using the same viewing transformation for both panoramas to adjust for head orientation. When looking in a specific viewing direction, the correct perspective for the left and right eyes will be projected in the headset, with the correct horizontal parallax for the front-facing pixels and a small degradation towards the periphery. The stereoscopic effect of the two images calculated in this way is guaranteed by the fact that human stereoscopic vision is concentrated mostly in the central portion of the field of view and does not exist in the outer peripheral zones.

The rendering on the headset is accomplished using a standard viewer for omnidirectional stereo images through a WebXR API browser. In practice, the high-resolution stereo panoramic images serve as textures for two spheres positioned in the scene — one centered around the left eye and the other around the right eye. The result is a kind of distinct environment map for each eye. As per the WebXR specifications, when XR rendering is enabled, the system retrieves parameters defining the head's position from the headset's sensors during each frame of the animation loop. These parameters are then used to create the correct perspective projections for each eye. For a given longitude θ, the left and right eyes will centrally align with images with the correct horizontal parallax disparity, thereby providing the effect of stereoscopic perception.

7. Results

The processing components to obtain stereo panoramic images for loading onto the headset have been developed in Python, using Pytorch to implement our custom networks, combined with standard image processing and computation libraries (NumPy, Pillow, OpenCV). The generative adversarial network used for zoom operations is *Real-ESRGAN* (with model *realesr-animevideov3*) [59], that has been directly integrated as a post-processing step in our system. The immersive rendering components, instead, have been realized in WebGL and WebXR.

7.1. Dataset and training

For training our solutions, we harness the availability of public panoramic scene datasets where ground truth is available. To train and test depth estimation, we exploit Structured3D [60]), a large-scale (21K photorealistic scenes) synthetic database of indoor scenes providing the ground truth depth for each panoramic image.

To train and test view synthesis, instead, we exploit PNVS [28], a subset of Structured3D scenes providing several translated views for each source panoramic image. Since the baseline for stereo view generation is very small, we opt for the PNVS subset known as *easy*, characterized by a maximum range of 300mm. This range comfortably exceeds our default radius of $r = 100$mm.

Train and test splits are maintained as in the original papers. The depth estimation network is trained and tested directly on the original images, while the view synthesis network is trained and tested on randomly oriented slices of the provided examples. The generation of randomly oriented slices is implemented as a data augmentation step.

Given that the available training and testing data sets are provided at a resolution of 1024×512 pixels, all our processing is done at that size. In the future, we plan to generate synthetic training and testing sets at higher resolution (2K-4K), exploiting the scalability of our networks to directly process images at typical native 360° camera resolution, removing the need for the downscaling and upscaling steps.

We trained both networks with the Adam optimizer [61], with $\beta_1 = 0.9, \beta_2 = 0.999$ and an adaptive learning rate from 0.0001, on an NVIDIA RTX A5000 (24GB VRAM) with a batch size of 8 for depth estimation and 4 for view synthesis. The average training time for the depth estimation network is $150 ms/image$, while for view synthesis is $160 ms/image$.

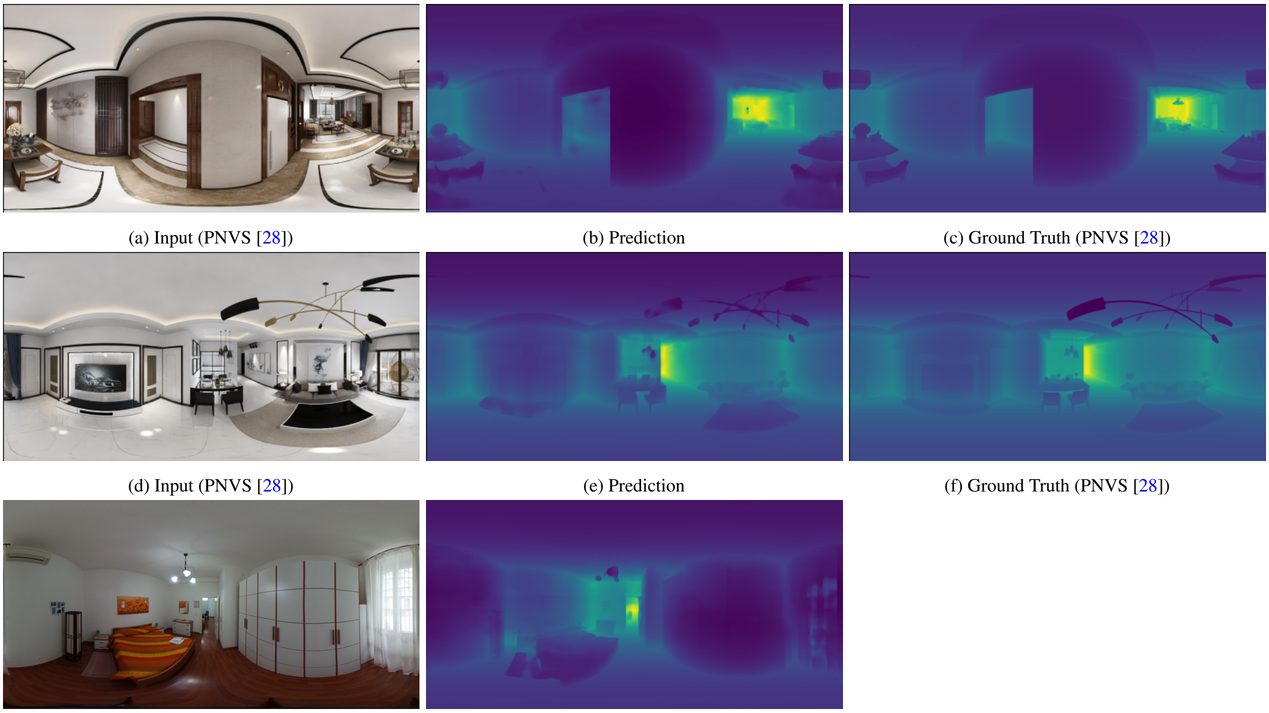
7.2. Computational performance

Our depth estimation and view synthesis baselines are extremely lightweight. Tab. 1 shows learnable parameters, GFlops, and milliseconds for different tasks and outputs. The benchmarks have been made on the same A5000 machine used for training.

In all presented tasks, we assume 512×1024 as the source image tensor resolution. Indeed, the output resolution is the same for depth estimation, where the network configuration uses 32 internal channels (Sec. 4). For the view synthesis task, instead, we compare computational stats to generate a full 360° image (i.e., 512×1024 tensor size) and to generate a slice (i.e., 512×160) that is the final utilization of the network. For this task, we adopt a network configuration with 64 internal channels, as well as other task-designed modifications (Sec. 5). The results clearly show the computational advantage in terms of GFlops and inference time. Subsequent results (Sec. 7.3) show that the choice of generating a slice maintains a performance advantage not only in computational terms. With the current approach, a full-quality omnidirectional stereo image, computed with an angular spacing of 1° between slice projection centers, can be

Table 1: **Computational performance.** We show the computational performance and latency time of our gated architecture for different tasks. In bold modes are the current architecture choices.

Mode	Output Res	Params	GFLOPS	ms/frame
Depth	512×1024	6.06 M	164.11	41
Synth	512×1024	6.93 M	326.71	95
Synth sliced	512×160	6.93 M	51.05	58



(a) Input (PNVS [28]) (b) Prediction (c) Ground Truth (PNVS [28])

(d) Input (PNVS [28]) (e) Prediction (f) Ground Truth (PNVS [28])

(g) Input (real capture) (h) Prediction

Fig. 7: **Depth estimation.** Two examples of depth prediction on PNVS [28] dataset scenes and an example of depth prediction from a user-acquired panoramic scene taken with a Ricoh Theta 360° camera.

generated and experienced in stereo with a latency of only about 21s from the time of the shot.

7.3. *View-synthesis performance*

As depth estimation is a fundamental task to achieve novel view synthesis, Tab. 2 presents the quantitative performance of our gated architecture compared to state-of-the-art panoramic depth solutions. We included in the evaluation the same error metrics used in many prior depth estimation works (e.g., [11, 10]): mean absolute error (MAE), mean relative error (MRE), root mean square error of linear measures (RMSE), and three relative accuracy measures δ_1, δ_2 and δ_3, defined, for an accuracy δ_n, as the fraction of pixels where the relative error is within a threshold of 1.25^n. The latter measures are useful to illustrate the error distribution.

We compare our performance with SliceNet [11] and HoHoNet [10], which are state-of-the-art methods commonly used as benchmarks in the latest panoramic works [22, 62], and, particularly for the domain of this paper, can provide performance with low latency. In this case, we show the reconstruction performance on the main Structured3D [60] test set, for which results from those baselines are available. We show some qualitative results in Fig. 7 (top two rows) on the PNVS scenes adopted instead for view-synthesis benchmarking. The deformation in the views is present in the original images and is due to the equirectangular projection, which preserves horizontal lines and curves vertical ones. The same deformation is visible in all other images in equirectangular format included in this article. As an illustration of how the same model can be applied to casually captured images, the bottom row of Fig. 7 shows how our network successfully predicts the depth of a user-acquired scene captured with a hand-held Ricoh Theta 360° camera.

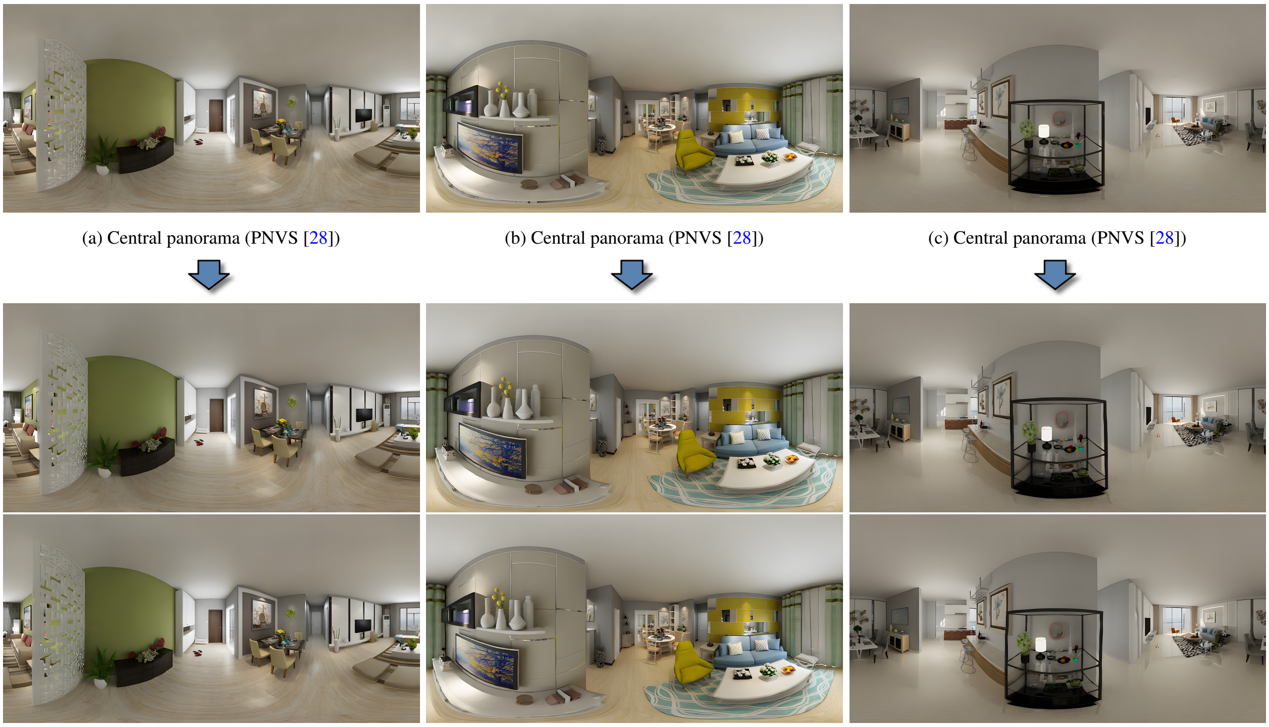
Table 2: **Quantitative performance comparison on depth reconstruction.** Our results are compared to other state-of-the-art works.

Method	MAE↓	MSE↓	RMSE↑	δ_1 ↑	δ_2 ↑	δ_3 ↑
HoHoNet [10]	0.081	0.065	0.206	0.958	0.987	0.993
SliceNet [11]	0.082	0.054	0.198	0.961	0.988	0.993
Our	0.061	0.008	0.038	0.962	0.989	0.994

As shown in the qualitative results of Fig. 7, the overall shape of the room is well preserved, and, similarly to other works [10, 11] the main prediction errors appear on very thin structure (e.g., the lamp in the second row). These thin structures are not very well resolved and, at run time, can cause visual artifacts during exploration. This problem is common to virtually all depth estimators from single images, and we expect to reduce them by increasing the resolution of images in the training set.

Tab. 3 summarizes our performance in terms of view synthesis accuracy, benchmarked on the PNVS [28] test dataset. . Despite presenting much more challenging translations than stereo parallax, this set provides a ground truth on which it is possible to compare with other state-of-the-art methods [63, 64] and among different versions of our architecture.

The results show that our method outperforms other baselines in generating a full equirectangular view (i.e., row 3). Furthermore, we show how reconstructing the single slice still achieves state-of-the-art performance even though the reconstruction is done by having a smaller context (i.e., 56 degrees vs. 360 degrees). The standard deviation of the error measures on the sliced



(a) Central panorama (PNVS [28])

(b) Central panorama (PNVS [28])

(c) Central panorama (PNVS [28])

(d) Left/right MCOP panoramas

(e) Left/right MCOP panoramas

(f) Left/right MCOP panoramas

Fig. 8: **Ominidirectional stereo panoramas.** Three representative scenes from the PNVS [28] dataset (testing split). Top row: source panorama. Middle row: automatically generated MCOP panorama for the left eye; Bottom row: automatically generated MCOP panorama for the right eye. The vertical alignment clearly shows the parallax effects.

Table 3: **View synthesis performance.** We show the quantitative performance of view synthesis compared to other state-of-the-art methods, all of which operate at a minimum resolution of 1024 × 512 image size (i.e., 512 × 1024 tensor size). The last line shows our sliced solution compared with our baseline trained to reconstruct the whole image.

Method	Output Res	PSNR↑	SSIM↑	LPIPS↓
SynSin [64]	512 × 1024	17.28	0.721	0.226
MPI [63]	512 × 1024	17.59	0.725	0.223
Our full	512 × 1024	21.55	0.731	0.202
Our full crop	512 × 160	22.77	0.738	0.196
Our sliced	512 × 160	23.00	0.744	0.186
Our-sliced-no-LPIPS	512 × 160	22.38	0.748	0.205

version amounts to 0.075 for LPIPS, 0.098 for SSIM, and 4.53 for PSNR, and is in very similar ranges for the other versions of the network. We noticed that the main errors found on the set of scenes are due to the imprecise depth reconstruction (see above), especially on thin structures. We thus identify depth estimation as one of the main avenues for improvement.

The last row shows the results obtained with an instantiation of our network trained without the $\mathcal{L}_{lpips}$ loss term. It should be noted how adding this term not only improves the LIPS metric but also has a beneficial effect on the PSNR. Since LPIPS strongly correlates with perceptual quality [58], its addition in the loss improves the final quality of presented images.

To compare the accuracy of the reconstruction, we measured performance by comparing random 512 × 160 crops on the full generated overview (i.e., training with 512 × 1024 tensor output), with slices generated with the dedicated network (i.e., training with 512 × 160). The experiments show that despite the significantly lower computational complexity, performance is on par, if not better, than generating a full equirectangular image for each angle.

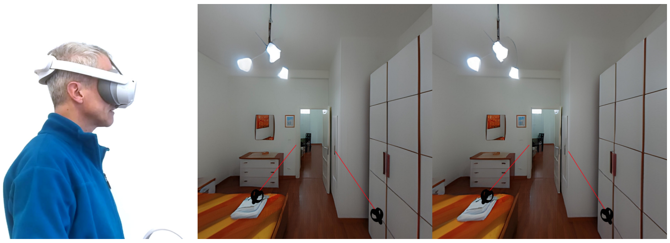
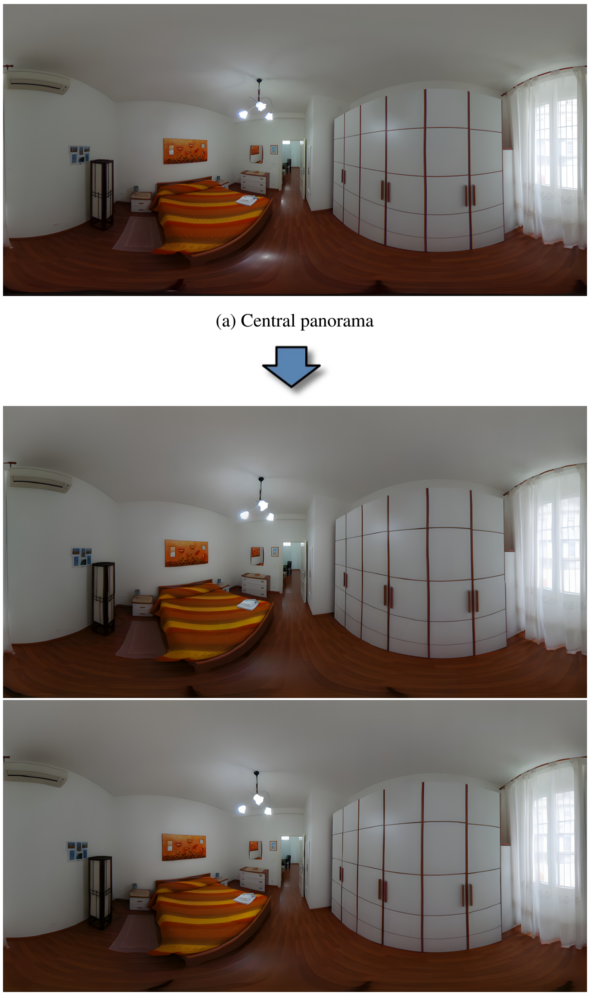


Fig. 9: **The WebXR viewer.** The user on the left wears a Pico4 HMD. The images to the right present the left and right images, as rendered by our WebXR viewer running on the PicoBrowser. The source image is a single-shot monoscopic 360° capture of a real environment, transformed to omnidirectional stereo by our framework.

7.4. Stereoscopic exploration on HMD

We tested the immersive viewer on various devices, including a Meta Quest 2 and an Android mobile device Samsung Galaxy S 22 with a Google Cardboard. Here we report on experiments made on a Pico4, a headset with two 2.56-inch Fast-LCD displays, a global resolution of 4320x2160 pixels (equivalent to 2160x2160 pixels per screen), a pixel density (PPI) of 1200, a variable refresh rate ranging between 72 and 90 Hz, and a diagonal Field of View (FOV) of 105 degrees (diagonal).

The web application for rendering is served by a web server that only has to transmit the two panoramic images to the HMD,



(a) Central panorama

(b) Left/right MCOP panoramas

Fig. 10: **Ominidirectional stereo panoramas.** Example from real-world capture. Top: source panorama, captured with a Ricoh Theta. Middle/Bottom: automatically generated MCOP panorama for the left and right eyes.

since the embedded client performs all the rest of the computation and rendering work. The client application is built on the ThreeJS framework, enabling the development of WebGL graphics components, and incorporates mechanisms for interaction with WebXR APIs. On the client side, the application, written in JavaScript ECMAScript 6 following a modular approach, is run on the native PicoBrowser when using Pico4, but the viewer can naturally be run on any headset compatible with WebXR specifications, as demonstrated by our tests on Android Phones with Google Cardboard and on the Meta Quest 2 (see accompanying video). Fig. 9 shows an image of the viewer. Other standard viewers supporting the omnidirectional stereo format can also be employed. Using a custom viewer allows us to implement specific operations (i.e., switching among scenes or from mono to stereo, or constraining/freeing the up vector during navigation).

When displayed on the HMD, the images provide immersive stereo cues, as also confirmed by an informal test with ten subjects who were requested to explore the stereoscopic environment on Oculus Quest and provide their opinion on immersion and stereoscopic perception of the generated scenes. The test was simply carried out by loading either the original 360° version or the omnidirectional stereo one. In all cases, users always differentiated the mono and stereo versions and confirmed that the stereo one was providing a more immersive experience.

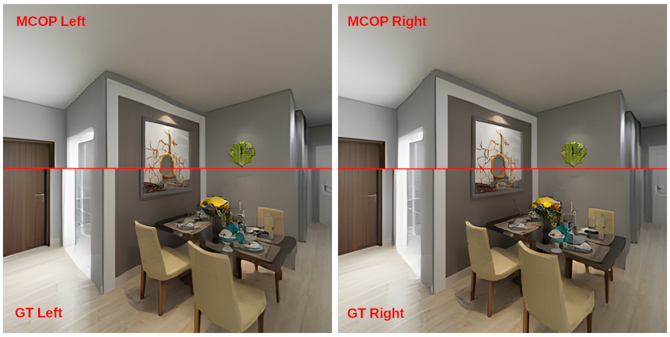
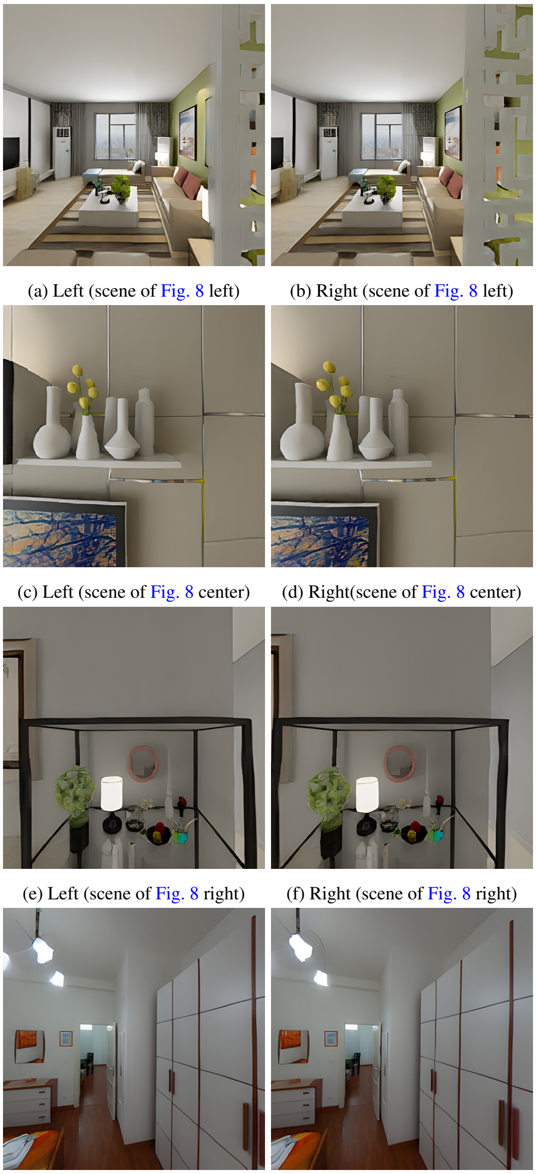


Fig. 11: **Comparison of omnidirectional stereo approximation with ground truth.** The top portion of the two images shows the perspective generated using the multiple-center-of-projection image for the left and the right eye, while the bottom portion shows the ground truth image generated with a center of projection placed at the eye position. As we can see, the perspective is indistinguishable at the center but slowly degrades in the periphery.

Fig. 8 shows the results of omnidirectional stereo generation for three representative scenes from the PNVS [28] dataset (testing split), while Fig. 10 shows the results of omnidirectional stereo generation for a real-world captured-scene. In both figures, the top row shows the source panorama, positioned at the center of the head, while the middle and bottom rows show the generated MCOP panoramas for the left and right eye, which incorporate stereo parallax effects.

Fig. 11 shows a comparison between the real-time rendering obtained from the omnidirectional stereo representation and a ground truth image. The top portion of the two images shows the perspective generated using the multiple-center-of-projection image for the left and the right eye, while the bottom portion shows the ground truth view, obtained by placing the center of projection placed at the eye position. As we can see, the two perspectives are indistinguishable at the center but slowly diverge when moving towards the periphery. We noticed that, while this effect is not perceived by the users for most of the scenes, in cases where a strong parallax exists (very nearby objects with details), the users perceive an effect of slight object motion when rotating the head while still being capable to perceive the parallax. This motion effect is due to the motion of the center of projection across the image. This is, however, an effect perceivable in all multiple-center-of-projection methods and is not introduced by our approach, which aims to present ways for the automated generation of those representations.

Representative frames recorded during navigation of the synthetic and real-world scenes are presented in Fig. 12. The differences in the views presented to the two eyes are noticeable, and the stereo parallax correctly responds to the geometry of the visible scene. To better illustrate the quality of displayed images, Fig. 13 shows representative details of Fig. 12 top left. Please refer to the accompanying video for additional examples.



(a) Left (scene of Fig. 8 left) (b) Right (scene of Fig. 8 left)

(c) Left (scene of Fig. 8 center) (d) Right(scene of Fig. 8 center)

(e) Left (scene of Fig. 8 right) (f) Right (scene of Fig. 8 right)

(g) Left (scene of Fig. 10) (h) Right (scene of Fig. 10)

Fig. 12: **Real-time navigation with omnidirectional stereo.** Representative stereo views for the scenes in Fig. 8 and Fig. 10.

8. Conclusions and future work

We presented a framework for the automatic generation of omnidirectional stereoscopic indoor environments to be used in immersive applications, especially consumed through head-mounted displays. Our method starts from a single panoramic image of an interior environment and uses data-driven architectures for depth estimation and novel view synthesis to quickly generate the images seen by both eyes during head rotation. For this work, these images are combined into an omnidirectional stereo representation, which is consumed on a lightweight WebXR viewer supporting stereoscopic exploration during head rotations.

The preliminary results show that the automatic generation components achieve state-of-the-art accuracy, and the visualization component can provide an immersive experience to casual users on a variety of devices. As a result, we can provide a quick method to enhance the exploration of environments acquired with the increasingly ubiquitous and affordable monoscopic panoramic cameras.

One of the limitations of the current approach stems from the mismatch between the resolution of the synthesized images and the achievable resolution with nowadays cameras and displays. This mismatch is currently handled by downsampling images before construction and a deep-learning-assisted upsampling before display presentation. The limitation is not due to the lightweight network architecture, which promises to be scalable to much larger image sizes, but instead to the availability of training sets for the depth estimation and view synthesis networks. We plan to tackle this problem by generating higher-resolution training data.

In terms of display, we have taken the approach of generating omnidirectional stereo images, which have the major advantage of requiring very limited rendering resources but also introduce a little degradation in the peripheral areas and when the view direction converges towards the poles. Since we have depth available, we can easily improve the method by incorporating state-of-the-art depth-dependent adaptations that have been designed for real captures [43]. In this context, it will be interesting to explore how our deep-learning-based solutions could be further adapted to directly produce the data required for depth-dependent adaptation. We will also evaluate the possibility of exploiting this approach to support a limited amount of horizontal and vertical head motion in addition to rotation, exploiting the fact that our networks can synthesize arbitrarily displaced images. Finally, we plan to use our panoramic capture and immersive rendering system as a building block for constructing applications that perform actions in shared physical and digital realities. One important direction of work will be to exploit these explorable panoramic environments to serve as interfaces for digital twins of buildings constructed from casually captured real data, that can provide location awareness and be easily annotated in a VR interface.

Acknowledgments

GP, FM, EG acknowledge the contribution of the Italian National Research Center in High Performance Computing, Big Data and Quantum Computing. FB acknowledges the contribution of Sardinian regional authorities under the XDATA project. MA received funding from NPRP-Standard (NPRP-S) 14th Cycle grant 0403-210132 AIN2 from the Qatar National Research Fund (a member of Qatar Foundation). The findings herein reflect the work, and are solely the responsibility, of the authors.

References

[1] Jokela, T, Ojala, J, Väänänen, K. How people use 360-degree cameras. In: Proc. MUM. 2019, p. 18:1–18:10. doi:10.1145/3365610.3365645.

[2] Xu, M, Li, C, Zhang, S, Le Callet, P. State-of-the-art in 360° video/image processing: Perception, assessment and compression. IEEE STSP 2020;14(1):5–26. doi:10.1109/JSTSP.2020.2966864.

[3] Matzen, K, Cohen, MF, Evans, B, Kopf, J, Szeliski, R. Low-cost 360 stereo photography and video capture. ACM TOG 2017;36(4):148:1–148:12. doi:10.1145/3072959.3073645.

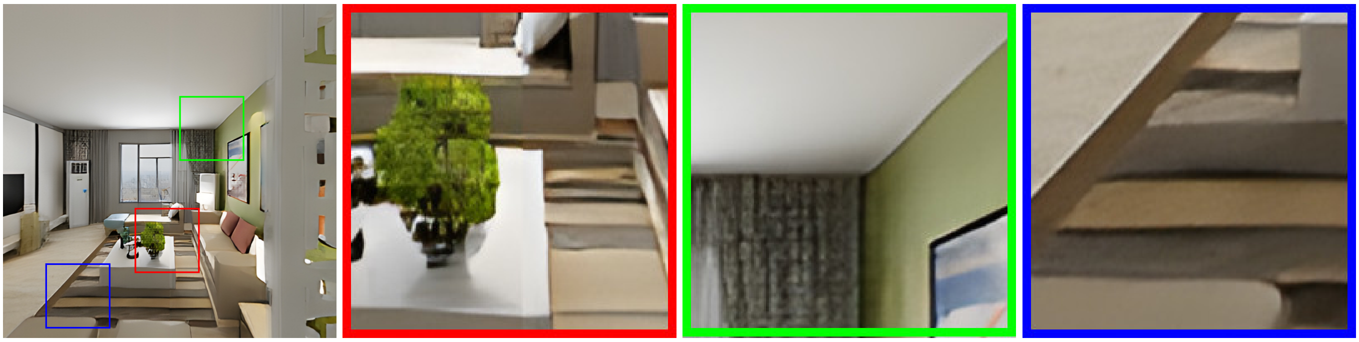


Fig. 13: Details of Fig. 12 top left

[4] Sulaiman, MZ, Aziz, MNA, Bakar, MHA, Halili, NA, Azuddin, MA. Matterport: virtual tour as a new marketing approach in real estate business during pandemic COVID-19. In: Proc. IMDES. 2020, p. 221–226. doi:10.2991/assehr.k.201202.079.

[5] Dong, H, Lee, JSA. The metaverse from a multimedia communications perspective. IEEE MultiMedia 2022;29(4):123–127. doi:10.1109/MMUL.2022.3217627.

[6] Waidhofer, J, Gadgil, R, Dickson, A, Zollmann, S, Ventura, J. PanoSynthVR: Toward light-weight 360-degree view synthesis from a single panoramic input. In: Proc. ISMAR. 2022, p. 584–592. doi:10.1109/ISMAR55827.2022.00075.

[7] Peleg, S, Ben-Ezra, M. Stereo panorama with a single camera. In: Proc. CVPR. 1999, p. 395–401. doi:10.1109/CVPR.1999.786969.

[8] Richardt, C, Tompkin, J, Wetzstein, G. Capture, reconstruction, and representation of the visual real world for virtual reality. In: Magnor, M, Sorkine-Hornung, A, editors. Real VR – Immersive Digital Reality: How to Import the Real World into Head-Mounted Immersive Displays. Springer International Publishing; 2020, p. 3–32. doi:10.1007/978-3-030-41816-8_1.

[9] Bertel, T, Yuan, M, Lindroos, R, Richardt, C. OmniPhotos: Casual 360° VR photography. ACM TOG 2020;39(6):266:1–266:12. doi:10.1145/3414685.3417770.

[10] Sun, C, Sun, M, Chen, HT. HoHoNet: 360° indoor holistic understanding with latent horizontal features. In: Proc. CVPR. 2021, p. 2573–2582. doi:10.1109/CVPR46437.2021.00260.

[11] Pintore, G, Agus, M, Almansa, E, Schneider, J, Gobbetti, E. SliceNet: deep dense depth estimation from a single indoor panorama using a slice-based representation. In: Proc. CVPR. 2021, p. 11536–11545. doi:10.1109/CVPR46437.2021.01137.

[12] Wang, X, Yu, K, Wu, S, Gu, J, Liu, Y, Dong, C, et al. ESRGAN: Enhanced super-resolution generative adversarial networks. In: Proc. ECCVW. 2018, p. 63–79. doi:10.1007/978-3-030-11021-5_5.

[13] Pintore, G, Jaspe Villanueva, A, Hadwiget, M, Gobbetti, E, Schneider, J, Agus, M. PanoVerse: automatic generation of stereoscopic environments from single indoor panoramic images for metaverse applications. In: Proc. ACM Web3D. 2023, p. 2:1–2:10. doi:10.1145/3611314.3615914.

[14] Pintore, G, Mura, C, Ganovelli, F, Fuentes-Perez, L, Pajarola, R, Gobbetti, E. State-of-the-art in automatic 3D reconstruction of structured indoor environments. Comput Graph Forum 2020;39(2):667–699. doi:10.1111/cgf.14021.

[15] Gao, S, Yang, K, Shi, H, Wang, K, Bai, J. Review on panoramic imaging and its applications in scene understanding. IEEE TIM 2022;71:1–34. doi:10.1109/TIM.2022.3216675.

[16] da Silveira, TLT, Pinto, PGL, Murrugarra-Llerena, J, Jung, CR. 3d scene geometry estimation from 360° imagery: A survey. ACM Comput Surv 2022;55(4):68:1–68:39. doi:10.1145/3519021.

[17] Zioulis, N, Karakottas, A, Zarpalas, D, Daras, P. OmniDepth: Dense depth estimation for indoors spherical panoramas. In: Proc. ECCV. 2018, p. 453–471. doi:10.1007/978-3-030-01231-1_28.

[18] Su, YC, Grauman, K. Learning spherical convolution for fast features from 360 imagery. In: Proc. NIPS. 2017, p. 529–539.

[19] Tateno, K, Navab, N, Tombari, F. Distortion-aware convolutional filters for dense prediction in panoramic images. In: Proc. ECCV. 2018, p. 732–750. doi:10.1007/978-3-030-01270-0_43.

[20] Coors, B, Condurache, AP, Geiger, A. SphereNet: Learning spherical representations for detection and classification in omnidirectional images. In: Proc. ECCV. 2018, p. 518–533. doi:10.1007/978-3-030-01240-3_32.

[21] Martin, D, Serrano, A, Masia, B. Panoramic convolutions for 360° single-image saliency prediction. In: Proc. CVPR workshop on computer vision for augmented and virtual reality. 2020, p. 1–4.

[22] Rey-Area, M, Yuan, M, Richardt, C. 360MonoDepth: High-resolution 360° monocular depth estimation. In: Proc. CVPR. 2022, p. 3752–3762. doi:10.1109/CVPR52688.2022.00374.

[23] Pintore, G, Almansa, E, Sanchez, A, Vassena, G, Gobbetti, E. Deep panoramic depth prediction and completion for indoor scenes. Computational Visual Media 2024;doi:10.1007/s41095-023-0358-0.

[24] Huang, J, Chen, Z, Ceylan, D, Jin, H. 6-DOF VR videos with a single 360-camera. In: Proc. IEEE VR. 2017, p. 37–44. doi:10.1109/VR.2017.7892229.

[25] Tukur, M, Pintore, G, Gobbetti, E, Schneider, J, Agus, M. SPIDER: Spherical indoor depth renderer. In: Proc. Smart Tools and Applications in Graphics (STAG). 2022, p. 131–138. doi:10.2312/stag.20221267.

[26] Tukur, M, Pintore, G, Gobbetti, E, Schneider, J, Agus, M. SPIDER: A framework for processing, editing and presenting immersive high-resolution spherical indoor scenes. Graphical Models 2023;128:101182:1–101182:11. doi:10.1016/j.gmod.2023.101182.

[27] Luo, B, Xu, F, Richardt, C, Yong, JH. Parallax360: Stereoscopic 360° scene representation for head-motion parallax. IEEE TVCG 2018;24(4):1545–1553. doi:10.1109/TVCG.2018.2794071.

[28] Xu, J, Zheng, J, Xu, Y, Tang, R, Gao, S. Layout-guided novel view synthesis from a single indoor panorama. In: Proc. CVPR. 2021, p. 16438–16447. doi:10.1109/CVPR46437.2021.01617.

[29] Pintore, G, Bettio, F, Agus, M, Gobbetti, E. Deep scene synthesis of Atlanta-world interiors from a single omnidirectional image. IEEE TVCG 2023;29. doi:10.1109/TVCG.2023.3320219.

[30] Trinidad, MC, Brualla, RM, Kainz, F, Kontkanen, J. Multi-view image fusion. In: Proc. ICCV. 2019, p. 4101–4110. doi:10.1109/ICCV.2019.00420.

[31] Reda, F, Kontkanen, J, Tabellion, E, Sun, D, Pantofaru, C, Curless, B. FILM: Frame interpolation for large motion. In: Proc. ECCV. 2022, p. 250–266. doi:10.1007/978-3-031-20071-7_15.

[32] Hedman, P, Kopf, J. Instant 3D photography. ACM TOG 2018;37(4):101:1–101:12. doi:10.1145/3197517.3201384.

[33] Serrano, A, Kim, I, Chen, Z, DiVerdi, S, Gutierrez, D, Hertzmann, A, et al. Motion parallax for 360° RGBD video. IEEE TVCG 2019;25(5):1817–1827. doi:10.1109/TVCG.2019.2898757.

[34] Lin, KE, Xu, Z, Mildenhall, B, Srinivasan, PP, Hold-Geoffroy, Y, DiVerdi, S, et al. Deep multi depth panoramas for view synthesis. In: Vedaldi, A, Bischof, H, Brox, T, Frahm, JM, editors. Proc. ECCV. 2020, p. 328–344. doi:10.1007/978-3-030-58601-0_20.

[35] Broxton, M, Flynn, J, Overbeck, R, Erickson, D, Hedman, P, DuVall, M, et al. Immersive light field video with a layered mesh representation. ACM TOG 2020;39(4):86:1–86:15. doi:10.1145/3386569.3392485.

[36] Zhou, T, Tucker, R, Flynn, J, Fyffe, G, Snavely, N. Stereo magnification: Learning view synthesis using multiplane images. ACM TOG 2018;37(4):68:1–68:12. doi:10.1145/3197517.3201323.

[37] Tucker, R, Snavely, N. Single-view view synthesis with multiplane images. In: Proc. CVPR. 2020, p. 548–557. doi:10.1109/CVPR42600.2020.00063.

[38] Li, Q, Khademi Kalantari, N. Synthesizing light field from a single image with variable MPI and two network fusion. ACM TOG 2020;39(6):229:1–229:10. doi:10.1145/3414685.3417785.

[39] Attal, B, Ling, S, Gokaslan, A, Richardt, C, Tompkin, J. MatryODShka: Real-time 6dof video view synthesis using multi-sphere images. In: Proc. ECCV. 2020, p. 441–459. doi:10.1007/978-3-030-58452-8_26.

[40] Bourke, P. Capturing omni-directional stereoscopic spherical projections with a single camera. In: Proc. IEEE VSMM. 2010, p. 179–183. doi:10.1109/VSMM.2010.5665988.

[41] Le, H, Liu, F. Appearance flow completion for novel view synthesis. Computer Graphics Forum 2019;38(7):555–565. doi:https://doi.org/10.1111/cgf.13860.

[42] Rademacher, P, Bishop, G. Multiple-center-of-projection images. In: Proc. SIGGRAPH. 1998, p. 199–206. doi:10.1145/280814.280871.

[43] Marrinan, T, Papka, ME. Real-time omnidirectional stereo rendering: generating 360° surround-view panoramic images for comfortable immersive viewing. IEEE TVCG 2021;27(5):2587–2596. doi:10.1109/TVCG.2021.3067780.

[44] Mohanto, B, Islam, A, Gobbetti, E, Staadt, O. An integrative view of foveated rendering. Computers & Graphics 2022;102:474–501. doi:10.1016/j.cag.2021.10.010.

[45] Gkitsas, V, Sterzentsenko, V, Zioulis, N, Albanis, G, Zarpalas, D. PanoDR: Spherical panorama diminished reality for indoor scenes. In: Proc. CVPR Workshops. 2021, p. 3711–3721. doi:10.1109/CVPRW53098.2021.00412.

[46] Ulyanov, D, Vedaldi, A, Lempitsky, V. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:160708022 2016;.

[47] Iizuka, S, Simo-Serra, E, Ishikawa, H. Globally and locally consistent image completion. ACM TOG 2017;36(4):107:1–107:14. doi:10.1145/3072959.3073659.

[48] Yu, J, Lin, Z, Yang, J, Shen, X, Lu, X, Huang, TS. Generative image inpainting with contextual attention. In: Proc. CVPR. 2018, p. 5505–5514. doi:10.1109/CVPR.2018.00577.

[49] Yu, J, Lin, Z, Yang, J, Shen, X, Lu, X, Huang, TS. Free-form image inpainting with gated convolution. In: Proc. ICCV. 2019, p. 4471–4480. doi:10.1109/ICCV.2019.00457.

[50] Yi, Z, Tang, Q, Azizi, S, Jang, D, Xu, Z. Contextual residual aggregation for ultra high-resolution image inpainting. In: Proc. CVPR. 2020, p. 7505–7514. doi:10.1109/CVPR42600.2020.00753.

[51] Yu, F, Koltun, V. Multi-scale context aggregation by dilated convolutions. In: Bengio, Y, LeCun, Y, editors. Proc. ICLR. 2016, p. 1–13.

[52] Lambert-Lacroix, S, Zwald, L. The adaptive BerHu penalty in robust regression. Journal of Nonparametric Statistics 2016;28(3):1–28. doi:10.1080/10485252.2016.1190359.

[53] Tulsiani, S, Tucker, R, Snavely, N. Layer-structured 3D scene inference via view synthesis. In: Proc. ECCV. 2018, p. 302–317. doi:10.1007/978-3-030-01234-2_19.

[54] Zioulis, N, Karakottas, A, Zarpalas, D, Alvarez, F, Daras, P. Spherical view synthesis for self-supervised 360° depth estimation. In: Proc. 3DV. 2019, p. 690–699. doi:10.1109/3DV.2019.00081.

[55] Gatys, LA, Ecker, AS, Bethge, M. Image style transfer using convolutional neural networks. In: Proc. CVPR. 2016, p. 2414–2423. doi:10.1109/CVPR.2016.265.

[56] Simonyan, K, Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:14091556 2014;.

[57] Isola, P, Zhu, JY, Zhou, T, Efros, AA. Image-to-image translation with conditional adversarial networks. In: Proc. CVPR. 2017, p. 1125–1134. doi:10.1109/CVPR.2017.632.

[58] Zhang, R, Isola, P, Efros, AA, Shechtman, E, Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In: Proc. CVPR. 2018, p. 586–595. doi:10.1109/CVPR.2018.00068.

[59] Wang, X, Xie, L, Dong, C, Shan, Y. Real-ESRGAN: Training real-world blind super-resolution with pure synthetic data. In: Proc. ICCVW. 2021, p. 1905–1914. doi:10.1109/ICCVW54120.2021.00217.

[60] Zheng, J, Zhang, J, Li, J, Tang, R, Gao, S, Zhou, Z. Structured3D: A large photo-realistic dataset for structured 3D modeling. In: Proc. ECCV. 2020, p. 519–535. doi:978-3-030-58545-7_30.

[61] Kingma, DP, Ba, J. Adam: A method for stochastic optimization. ArXiv e-print arXiv:14126980 2014;.

[62] Li, Y, Guo, Y, Yan, Z, Huang, X, Duan, Y, Ren, L. Omnifusion: 360 monocular depth estimation via geometry-aware fusion. In: Proc. CVPR workshop on computer vision for augmented and virtual reality. 2022, p. 2801–2810.

[63] Tucker, R, Snavely, N. Single-view view synthesis with multiplane images. In: Proc. CVPR. 2020, p. 551–560. doi:10.1109/CVPR42600.2020.00063.

[64] Wiles, O, Gkioxari, G, Szeliski, R, Johnson, J. Synsin: End-to-end view synthesis from a single image. In: Proc. CVPR. 2020, p. 7467–7477. doi:10.1109/CVPR42600.2020.00749.