











Immersive Free-Viewpoint Panorama Rendering from Omnidirectional Stereo Video

Moritz Mühlhausen,  Moritz Kappel,  Marc Kassubeck,  Leslie Wöhler,  Steve Grogorick,  Susana Castillo, 
Martin Eisemann  and Marcus Magnor 

Institut für Computergraphik, TU Braunschweig, Braunschweig, Germany
{muehlhausen, kappel, kassubeck, woehler, grogorick, castillo, eisemann, magnor}@cg.cs.tu-bs.de

Abstract

In this paper, we tackle the challenging problem of rendering real-world 360° panorama videos that support full 6 degrees-of-freedom (DoF) head motion from a prerecorded omnidirectional stereo (ODS) video. In contrast to recent approaches that create novel views for individual panorama frames, we introduce a video-specific temporally-consistent multi-sphere image (MSI) scene representation. Given a conventional ODS video, we first extract information by estimating framewise descriptive feature maps. Then, we optimize the global MSI model using theory from recent research on neural radiance fields. Instead of a continuous scene function, this multi-sphere image (MSI) representation depicts colour and density information only for a discrete set of concentric spheres. To further improve the temporal consistency of our results, we apply an ancillary refinement step which optimizes the temporal coherency between successive video frames. Direct comparisons to recent baseline approaches show that our global MSI optimization yields superior performance in terms of visual quality. Our code and data will be made publicly available.

Keywords: 360 videos, immersive VR, multi-sphere image, neural network, omnidirectional stereo rendering

CCS Concepts: • Computing methodologies → Virtual reality; Neural networks; Ray tracing; Perception; • Human-centred computing → Graphics input devices

1. Introduction

The uprise of consumer-grade head-mounted displays (HMDs) made Virtual Reality (VR) multimedia content accessible to the wide public [Eli94, KAS*19]. Besides commercial entertainment media formats like video games, sport broadcasts and live-action movies [TBLG16, IH17, Epi, Moz18], the availability of mass-market cameras and HMDs enable users to capture and replay custom content in immersive VR environments. This increased interest in Virtual Reality (VR) technology resulted in numerous research advancements regarding 360° content and adaptation of deep learning technology for VR [WLZ*20, WLLZ20]. Currently, one of the most popular data formats for capturing such personalized content is omnidirectional stereo (ODS) video, as it is supported by most modern panorama camera systems.

The ODS video format uses two spherical panoramas, one for each eye, with a fixed baseline, which can directly be visualized in VR headsets with 360° head rotation and a natural impression

of depth provided by stereo parallax. However, ODS panorama videos do not natively support viewer head motion (and resultant motion parallax) due to the fixed camera centre and implicit depth information, and even head rotation is limited to yaw and pitch directions, which can strongly decrease the perceived immersion or even cause motion sickness for some viewers [TG18, PRS*18]. For this reason, several strategies for adding head motion parallax to panorama video using explicit scene geometry have been investigated [MST*20, BFO*20, ALG*20, MKK*20]. Recent approaches propose to decompose the panorama video and synthesize novel viewpoints using multi-sphere images (MSIs) [BFO*20, ALG*20]. Analogous to multi-plane images (MPIs), that represent a scene using multiple fronto-parallel planes at different depths, MSIs comprise spheres with predefined radii which enable fast rendering in an inside-out fashion. Inspired by the Stereo Magnification method [ZTF*18] for MPIs, the recent MatryODShka approach [ALG*20] estimates MSI layer blending weights from ODS panoramas using sphere-sweep-volumes. While this method

utilizes transform-inverse MSI regularization resulting in general consistency for small transformations, it does not account for scene-dependent temporal-consistency.

In this paper, we propose a method that converts a omnidirectional stereo video to the MSI representation using an encoder-decoder neural network. In order to guarantee temporal consistency, our method combines insights of MPI generation for regular images [LFS*21] with scene-specific temporal refinement for panorama videos [MKK*20]. We employ a two-step optimization procedure: In the first step, we optimize the conversion for each frame of the input ODS video to an MSI representation individually. In the second step, we introduce temporal coherence between pairs of consecutive frames to minimize jittering artefacts.

After conversion to the MSI format, a panorama video can be rendered in real time with full head-mounted display (HMD) head motion support using classical volume rendering. At the same time, our approach does not only improve over current ODS-based panorama rendering systems in terms of visual quality, but can easily be extended to handle context-aware dynamically resizing depth layers.

Our novel learning-based optimization approach converts omnidirectional stereo video footage to more powerful MSI panorama videos and enables:

- temporally consistent video playback,
- full HMD head motion support,
- superior visual quality compared to current ODS-based panorama rendering systems,
- real-time volume rendering using ray-tracing.

Our code and data will be made publicly available.

2. Related Work

This paper focuses on transferring videos in the widely-used ODS scene representation into a MSI to support head motion parallax for immersive video playback in HMDs. Therefore, in the following we cover the related fields of capturing and different techniques for novel view synthesis.

ODS recording: Capturing ODS content is an active field of research yielding a large number of specialized approaches. Early systems have been able to generate stereo panorama images from either specialized setups [IYT92] or movement paths of monocular cameras [RPZSH13]. Later, more sophisticated camera systems have been proposed to enhance the recorded scenes with head motion parallax enabling free 6 degrees-of-freedom (DoF) movements [LXRY18, OEE*18]. Using techniques from the fields of scene reconstruction [BCR19, BYLR20] or flow-based blending [HASK17], not only stereo but also head motion parallax can be obtained from singular cameras. Since all of these approaches generally require a longer time to capture the scene, they are not suited for dynamic videos. Following the growth of the virtual reality user base, several panorama video cameras were developed that are able to capture ODS video footage utilizing several outward facing cameras, like the Insta 360, Jaunt One and GoPro Odyssey. While these are not able to directly generate content that enables head motion parallax, there are different possibilities to adapt the recordings us-

ing scene reconstruction approaches [HCCJ17, IHR*16, Ven16]. For our scene recordings we used the Insta 360 Pro, but any ODS capturing device could be used.

Novel view synthesis: In order to view ODS content with head motion parallax, it is necessary to render novel view points in real time. One possibility to generate new views with high quality is *light field rendering* [MB95, LH96, BBM*01]. Techniques following this line utilize a densely sampled scene which usually requires specialized recording equipment or setups. In contrast, *pointcloud representations* can be obtained from arbitrary camera setups [TBLG16, TLWG17, BMK*20]. They can also easily be rendered but may suffer from artefacts introduced by disocclusions in the recorded scenes which appear as holes in the pointclouds. One representation that circumvents these limitations are *textured meshes* [HASK17, HK18, SKC*19, LXLL19, MKK*20]. However, mesh-based representations tend to introduce visible artefacts at depth discontinuities. The stacking of images in *layered representations* like MPIs has been successfully used for novel view synthesis [TRB*20]. Utilizing a set of fronto-parallel planes, MPIs can be rendered efficiently. This approach has been adapted for panorama videos by replacing 2D planes with spherical shell meshes [BFO*20]. This way several spherical layers can be combined into a layered mesh to reduce the required memory, allowing real time playback even for high resolution videos as needed for VR applications. However, this approach requires a specialized recording setup to obtain high-quality results, making it incompatible with already existing footage. In contrast, our approach does not require a special setup, but can be used for any arbitrary recorded ODS video.

Similar to our approach, MatryODShka [ALG*20] takes ODS frames and generates sphere-sweep volumes, an adaption of plane-sweep volumes for panoramas. These volumes are used as input and transformed into blending weights for the input ODS to form an MSI representation. While this overall produces reasonable results for single frames, it is lacking explicit scene-specific temporal consistency and struggles with more complex scenes. Therefore, our approach relies on optimizing a neural network in a scene-dependent manner instead of learning and generalizing for unknown scenery. This allows for temporally consistent optimization, making our approach better suited for ODS video footage.

3. Method

Our method generates temporally consistent VR video content with motion parallax by transferring each left and right image pair of an ODS video to the MSI representation as shown in Figure 1. For this, we use an encoder-decoder neural network, but instead of training it to generalize for arbitrary scenes, we utilize this architecture for video optimization per scene. An overview of our MSI creation approach with its inputs and outputs is illustrated in Figure 2.

Following in Section 3.1, we first highlight the differences between two panorama projections, which we utilize for our approach. In Section 3.4 the MSI scene representation and its relation towards these projections are elaborated. Subsequently we present our ODS to MSI translation network in Section 3.7. Hereby, we present the in- and outputs of the encoder and decoder as well as processing steps in between. Section 3.10 describes our optimization process

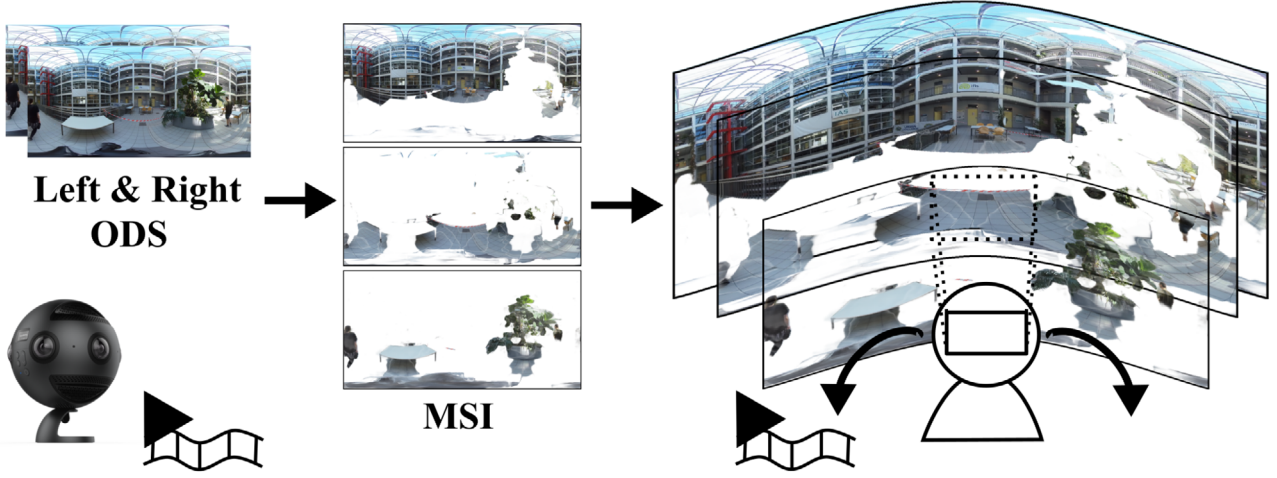


Figure 1: We present a method that converts consumer-grade omnidirectional stereo (ODS) recordings (left) to a multi-sphere image (MSI) representation (middle). To this end we employ an encoder–decoder neural network structure as a tool for scene-dependent optimization to estimate a temporally consistent MSI representation. While omnidirectional stereo (ODS) footage supports head rotation with realistic depth impression, it does not support head motion and free viewpoint rendering. In contrast the MSI representation enables immersive rendering with full head motion support for virtual reality applications (right).

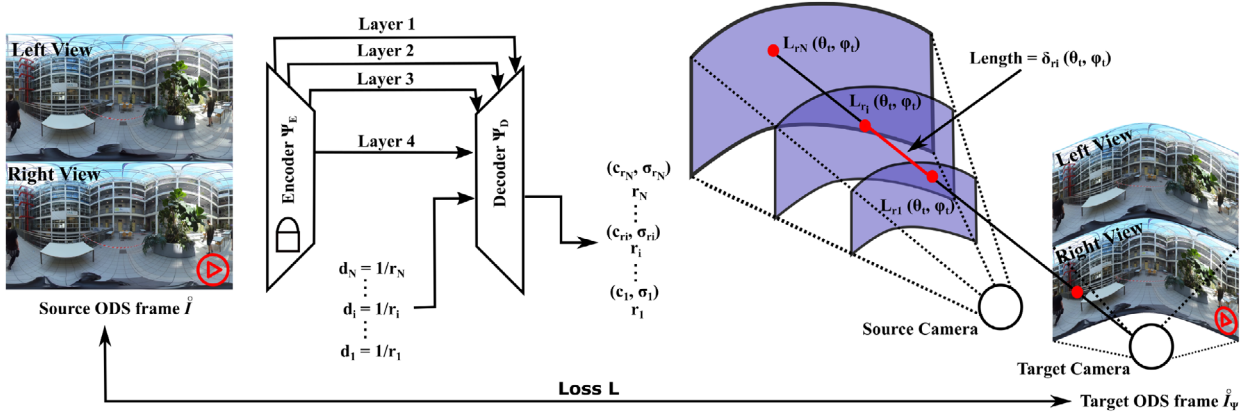


Figure 2: Overview of our omnidirectional stereo (ODS) to multi-sphere image (MSI) translation network: The input consists of a left and right eye video frame of an ODS recording \hat{I} , which are first encoded using a ResNet50 [HZRS16] encoder Ψ_E , see Section 3.7.1. The sum of these deep image features of both views and the disparity encoding of a single spherical layer's radius r as $d_i = 1/r_i$ is fed into our decoder Ψ_D to generate the corresponding multi-sphere image (MSI) layer L_r , as described in Section 3.7.2. Utilizing several MSI layers, we can render input ODS frames \hat{I}_ψ or novel viewpoints utilizing ray-tracing for volume rendering (see Equation 1).

and presents the loss functions for the training signal. Lastly, in Section 3.16 we display our rendering approach from an MSI representation.

3.1. Standard panorama projections

Since our method works with video footage in ODS representation, we first give a brief overview of the differences between the standard equirectangular (ER) panorama projection and the recently widespread used ODS projection, as visualized in Figure 3.

Equirectangular projection: An ER projection of a scene in centre c results in a single image \hat{I} in which each pixel $p(\theta, \phi)$ maps directly to the surface of the unit sphere. The x-coordinate corresponds to the azimuthal angle $\theta \in [0, 2\pi]$ and the y-coordinate to the polar angle $\phi \in [0, \pi]$. Figure 3a depicts the camera rays of a single row in ER projection.

ODS projection: With the advent of HMDs, the ODS projection gained popularity. In contrast to the ER projection, it features implicit depth perception when viewed within an HMD as it produces two distinct images $\hat{I}_{l/r}$, one for the right and one for the left eye. Analogous to ER projections, the x and y-coordinates resemble the

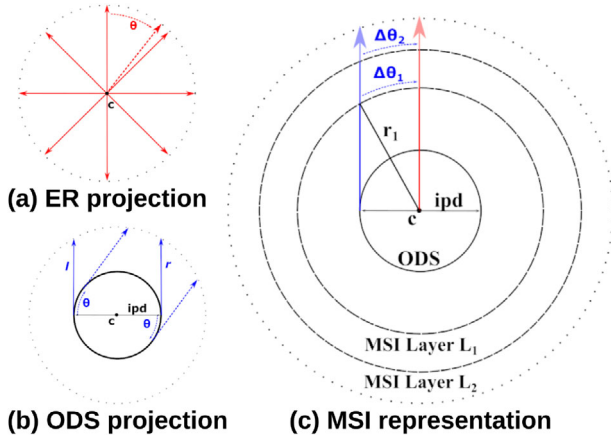


Figure 3: Projection of a single pixel row in two commonly used panorama projections and their relation to the MSI scene representation. The ER projection (a) samples the full sphere around a fixed camera projection centre c , resulting in a single image \dot{I} . In contrast, the ODS projection (b) produces two distinct images $\dot{I}_{l/r}$, one for each eye, to offer an implicit depth perception when viewed in an HMD. When rotating the camera centre, ray pairs are generated, which are tangential to the circle of diameter equal to the interpupillary distance (ipd). An MSI scene representation (c) consists of several concentric spherical layers. It is easy to render the scene in ER (red camera ray) from this representation. Rotating the layers L_{r_i} by $\Delta\theta_{r_i}$ according to their radius r_i , an ODS view can be rendered in the same way as an ER view (blue and red camera rays).

azimuthal θ and polar angle ϕ , however, the camera rays originate from a circle instead of a single point. Hereby, the origin depends on the azimuthal angle θ and whether it is projecting for the left or the right eye. The direction of the ray is tangential to the ODS circle and the origin for the left and right eye rays lie on opposing sides. This circle is centred around the projection centre c and its diameter corresponds to the ipd between the eyes. Interestingly, an ODS projection with an ipd of 0 results in two identical images which are also equivalent to the ER projection with the same centre: $\dot{I}_l = \dot{I}_r = \dot{I}$. Figure 3b shows the camera rays for the left and right eye on the viewing circle for two points in a single row in the ODS projection.

3.2. MSI scene representation

While neural radiance fields are not directly feasible for real-time rendering for novel view synthesis, they changed the perspective of multi-plane images (MPIs) research from the usual MPI layers at fixed depths towards planar neural radiance fields [LFS*21]. Instead of densely sampling the space along a camera ray, MPI rendering only samples the space at a sparse set of fronto-parallel planes. This allows for real-time novel view rendering, but is restricted to a constraint set of viewing directions, making it inapplicable for VR applications with free head motion and rotation.

As an evolution of MPIs for VR applications, MSIs are also feasible for real-time rendering and further support full head rotations. Instead of fronto-parallel planes, they feature a set of multiple concentric spherical layers $L_r = (c_r, \sigma_r)$ with different radii r around a centre c in an ER-like projection. Each layer consists of RGB colour $c_r : (\theta, \phi)^T \rightarrow \mathbb{R}^3$ and volume density values $\sigma_r : (\theta, \phi)^T \rightarrow \mathbb{R}$. Hereby, the volume density $\sigma_r(\theta, \phi)$ corresponds to the probability of a camera ray terminating at the infinitesimal thin layer $L_r(\theta, \phi)$. Similar to planar neural radiance fields, this representation is able to acquire colour and volume density values for any 3D position, as each spherical layer can have an arbitrary radius r . Generally, for rendering views from an MSI representation, all camera rays are traced through all N layers $\{L_{r_i} = (c_{r_i}, \sigma_{r_i}) \mid i = 1, \dots, N\}$ to compute the resulting colour comparable to classical volume rendering [MST*20, LFS*21]:

$$I = \sum_{i=1}^N T_i (1 - \exp(-\sigma_{r_i} \delta_{r_i})) c_{r_i}, \quad (1)$$

with

$$T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_{r_j} \delta_{r_j}\right)$$

resembling the probability of radiance from layer L_{r_i} to reach the camera without hitting any object. δ_{r_i} denotes the distance the ray travelled between layer $L_{r_{i-1}}$ and L_{r_i} , as depicted on the right hand side in Figure 2.

Rendering in ER projection: Rendering a full panorama in ER projection from an MSI representation in centre c is straight-forward and does not require actual ray-tracing. As all MSI layers L_r are already in ER projection, each ray will hit the same pixel coordinate in each layer. Furthermore, with concentric spherical layers, the distance travelled between layers is known $\delta_{r_i} = r_i - r_{i-1}$, see the red camera rays in Figure 3c.

Rendering in ODS projection: Overall, we want our estimated MSI to closely resemble the original ODS images to avoid a loss of information or the introduction of artifacts. Therefore, we need a way to compare the MSI to the original input ODS. To easily enable this comparison, we reproject the output MSI into an ODS projection.

This reprojection involves intersecting the ODS camera rays with the given MSI layers. As visible in Figure 3c, the azimuthal intersection difference $\Delta\theta_{r_i}$ between an ER and an ODS reprojection depends on the radius r_i , the ipd and the polar angle ϕ (since the effective projected radius depends on the polar angle):

$$\Delta\theta_{r_i, \phi} = \arcsin\left(\frac{ipd}{2 \cdot \sin(\phi) \cdot r_i}\right). \quad (2)$$

Consequently, rotating all spherical MSI layers accordingly, an ODS projection can be rendered as easy as an ER projection. Hereby the rotation direction for the left and right eye views of the ODS are opposing, that is, for the left eye view the rotation has to be done clockwise, for the right eye counter-clockwise.

The distance δ_{r_i} travelled by camera rays between layers depend on the radii r_i and r_{i-1} :

$$\delta_{r_i}^s = \sqrt{r_i^2 - \left(\frac{\text{ipd}}{2}\right)^2} - \sqrt{r_{i-1}^2 - \left(\frac{\text{ipd}}{2}\right)^2},$$

with
$$r_0 = \frac{\text{ipd}}{2}.$$

These two adaptations allow to render an ODS projection of the scene similar to an ER projection, since now every ray will also hit the same pixel coordinate in each layer. Therefore, it omits the need for actual ray-tracing.

3.3. Network

Inspired by previous work on rectangular images [GMAFB19, LFS*21], we adopted a two-folded network architecture: an encoder network Ψ_E and a decoder network Ψ_D , as depicted in Figure 2. To create an MSI we first use a ResNet50 encoder, which generates meaningful local feature maps on decreasing image resolutions, allowing the decoder network to first extract global features at low resolution and incorporates local features during upconvolutions. Using the left and right view of an input ODS video frame $\hat{I}_{l/r}$ along with a set of radii $\mathbf{R} = \{r_i \mid i = 1, \dots, N\}$, our network Ψ produces an MSI with N spherical layers $\mathbf{L}_R = \{L_r \mid r \in \mathbf{R}\}$ that enable the user to experience head-motion parallax in VR environments. While the encoder network takes the ODS frames to generate multi-resolution descriptive feature maps for each view, the decoder network uses these feature maps along with a radius r_i to estimate the corresponding RGB colour and volume density values of layer L_{r_i} . Therefore, executing the decoder network with a set of radii \mathbf{R} will result in an MSI representation:

$$\Psi\left(\hat{I}_{l/r}, \mathbf{R}\right) = \Psi_D\left(\Psi_E\left(\hat{I}_{l/r}\right), \mathbf{R}\right) = \mathbf{L}_R. \quad (3)$$

To this end, the user first needs to define the number and radii for the layers of the MSI or use the same layout from training. While the number of layers needs to be balanced between depth precision (higher) and computational load (lower), the radii of the layers only influence depth quality. Besides manual placement, a sufficient automatic approach is to equally distribute the layers within the disparity range of the scene [ALG*20].

3.3.1. Encoder

We use a pretrained ResNet50 network [HZRS16] as the encoder network, whose weights are fixed and not updated during optimization. It generates meaningful feature maps on decreasing image resolutions. While the image resolution decreases, the receptive field of the feature maps increases, allowing for extraction of more global features at low resolution and local image features at higher resolutions. It is executed independently for the left and right eye view of the ODS $\hat{I}_{l/r}$. As depicted in Figure 2, we use the descriptive features of multiple ResNet50-layers with different resolutions, which we feed into the decoder network in ascending order at the corresponding decoder resolution. Specifically, we take the outputs (*Layer1*, *Layer2*, *Layer3*, *Layer4*) of ResNet50 [LFS*21].

3.3.2. MSI decoder

The decoder network takes the feature maps of both eyes $\Psi_E(\hat{I}_{l/r})$ of a frame and a single radius r as input and estimates the RGB and volume density values of a spherical layer with the corresponding radius. Therefore, in contrast to the encoder which is only executed once per frame, the decoder runs once per MSI layer \mathbf{L} . For this, the output of each layer of the encoder is passed to the corresponding layer of the decoder. To reduce the number of input channels, we sum the feature layers of both eyes, after rotating them to match the MSI layer representation.

Rotating features: Similar to MatryODShka [ALG*20], which uses a sweep sphere volume of the ODS images as input to their network, we rotate the feature layers of the encoder. Instead of creating a complete sweep sphere volume from those features, we only rotate them according to the single input radius r of the decoder. Rotating the layers before summing them, will superimpose image features with depth r , making it easier for the decoder to identify relevant features for this layer. Notably, this rotation correlates to the inverse rotation of an MSI layer into an ODS projection. Therefore, we rotate the left eye features counter-clockwise and right eye features clockwise according to Equation (2).

Disparity encoding: As suggested by previous research [MST*20], instead of directly feeding the radius r into the network, we use an encoding function on the disparity $d = 1/r$, which improves the ability of our network to optimize for high frequencies in natural imagery. This encoding function maps the single disparity value into a higher dimensional space with high frequency functions:

$$\gamma(d) = (\sin(2^0\pi d), \cos(2^0\pi d), \dots, \sin(2^9\pi d), \cos(2^9\pi d)).$$

The disparity encoding is then concatenated to the rotated features before being fed into the decoder.

3.4. Optimization

The optimization of our method consists of two steps: First, we train the decoder for each frame of the input video independently. In the second step, we use two consecutive frames to introduce temporal consistency and further refine the decoder output.

Before we start the training procedure, we define the number and radii of layers for the resulting MSI. While we predefine 16 radii, during optimization each step randomly samples all layer radii between the corresponding and previous layer radius as defined in this set. This random sampling of the MSI radii allows our network to learn a continuous scene representation instead of a fixed layered one [LFS*21]. We manually choose these values according to the depth distribution in each scene, whereby we prefer denser sampling at closer depths. As a default option, they are distributed uniformly across the disparity, similar to MatryODShka [ALG*20]; however, this can result in less parallax effect in the background. Therefore, our network learns a fully continuous scene representation instead of only a set of fixed spherical layers, enabling to change the radius at inference time without further training.

3.4.1. Loss functions

For training we use a sum of five loss functions. These compare the input ODS images \hat{I} with the ODS rendering from the output MSI layers \hat{I}_ψ according to Equation (2). Since optimizing with only two RGB images for each frame is underconstrained, we also employ loss functions regarding depth information. Our total loss function is defined as:

$$L = \lambda_{\text{data}}L_{\text{data}} + \lambda_{\text{VGG}}L_{\text{VGG}} + \lambda_{\text{disp}}L_{\text{disp}} + \lambda_{\text{empty}}L_{\text{empty}} + \lambda_{\text{temp}}L_{\text{temp}},$$

with $\lambda_{\text{data}}, \dots, \lambda_{\text{temp}}$ as hyperparameters to weigh the different terms.

Colour reconstruction loss: We use an L1 loss for the colour values between the input ODS and the one computed by the network. This encourages the networks MSI representation to match the input information.

$$L_{\text{data}} = \sum_{\theta, \phi} \left| \hat{I}_\psi(\theta, \phi) - \hat{I}(\theta, \phi) \right|.$$

Colour VGG loss: The L1 colour loss on its own tends to only slowly converge towards the expected colour values, as the L1 difference does not reflect the human perception well. In line with findings from previous research, additional feature maps of a VGG network Θ improve the results towards human perception [SZ15]. Therefore, we also employ an L1 loss on different output layers of a pretrained VGG network:

$$L_{\text{VGG}} = \sum \left| \Theta(\hat{I}_\psi) - \Theta(\hat{I}) \right|.$$

Disparity MSE loss: In addition to the ODS input frames we use depth information obtained from correspondences between the left and right eye of the stereo panorama to further constrain the problem during training. Even though in ODS imagery correspondences should only occur in the same row, with the generally distorted nature of panorama images, we find that optical flow approaches are better suited for this correspondence search task. In particular, we use FlowNet 2.0 [IMS*17] between the stereo pair of each frame. From these correspondences we calculate depth information $\hat{\mathbf{D}}$ for left and right eye ODS input frames [WSGD18].

Rendering a depth map from an MSI representation can be done similar to Equation (1):

$$\mathbf{D}_\psi = \sum_{i=1}^N T_i(1 - \exp(-\sigma_{r_i} \delta_{r_i})) r_i.$$

Accordingly, depth information in ODS projection $\hat{\mathbf{D}}_\psi$ can be rendered after rotating all layers as described in Equation (2).

We then use a mean squared error loss on disparities rather than depth to increase the importance of accurate foreground information:

$$L_{\text{disp}} = \frac{1}{N} \sum_{\theta, \phi} \left(\frac{1}{\hat{\mathbf{D}}_\psi(\theta, \phi)} - \frac{1}{\hat{\mathbf{D}}(\theta, \phi)} \right)^2.$$

For numerical stability and comparability between different scenes, when calculating L_{disp} , we normalize the depth $\hat{\mathbf{D}}$ between $[1, 100]$.

Empty space loss: Additionally, we adopt the empty space loss [XHKK21]. This loss penalizes density values on layers closer than the depth estimated from the stereo correspondences. This way, we avoid assigning density values in front of the estimated depth, reducing semi-transparent detached artefacts which sometimes occur from those wrongly estimated densities.

$$L_{\text{empty}} = \sum_{r \in \mathbf{R}} \sum_{\theta, \phi} \begin{cases} \hat{\sigma}_r(\theta, \phi) & \text{if } r < \hat{\mathbf{D}}(\theta, \phi) \\ 0 & \text{else} \end{cases}$$

Temporal smoothness loss: Our last term L_{temp} enforces temporal consistency between subsequent frames $(n-1, n)$ if the corresponding colour value of a pixel (θ, ϕ) in the ODS panorama frame \hat{I}^n did not change. This way, we can incorporate a temporal loss which penalizes large differences in colour values of the ODS images as proposed in previous work [MKK*20].

$$L_{\text{temp}} = \sum_{r \in \mathbf{R}} \sum_{\theta, \phi} \xi \left(\left(\hat{\mathbf{I}}_r^n(\theta, \phi) - \hat{\mathbf{I}}_r^{n-1}(\theta, \phi) \right) \exp \left(-\alpha_{\text{temp}} \cdot |\hat{I}^n - \hat{I}^{n-1}| \right) \right),$$

where α_{temp} is a free hyperparameter and ξ stands for the Charbonnier penalty function [BWS05, SRB10]. This loss is only active in the second part of the optimization finetuning. In the first part its corresponding weighting parameter is set to zero $\lambda_{\text{temp}} = 0$. After a fixed set of epochs without L_{temp} , we start the second part of the optimization including temporal consistency. For this, we use the same strategies as for the first step; however, we use a second, consecutive frame as additional input. To reduce memory requirements, we only update the weights of $\Psi_{\mathbf{D}}$ for $\hat{\mathbf{I}}^n$.

3.5. Rendering

For rendering novel views from an estimated MSI representation, we follow traditional volume rendering as shown in Equation (1). While rasterization is possible by texturing the spheres with the estimated RGB values, the alpha values can only be approximated, as they depend on the rays travelling distance δ between spheres. Therefore, we utilize GPU ray-tracing using CUDA for fast ray-sphere intersections per pixel, which allows for real-time rendering of novel views. Furthermore, we apply two additional adjustments for a more pleasant viewing experience.

Enforce minimum transparency: Since estimating an MSI representation from only two input images is highly underconstrained, high-frequency features with low density values can be misplaced on nearby MSI layers. This can result in unpleasant and disturbing semi-transparent floating artefacts which appear to be disconnected from the scene. To counteract this, we enforce pixel of each layer L_r to exceed a minimum transparency threshold $(1 - \exp(-\sigma_r \delta_r))$ to be considered for rendering.

Background merging: Lastly, we merge the estimated MSI representation of a video frame with the MSI representation of a



Figure 4: Representative frame cutouts for the four videos used for qualitative comparisons – our three scenes and the Mall scene from NDD [MKK*20] (names highlighted in blue and orange, respectively) – and the eleven scenes from the Replica dataset [SWM*19] (names highlighted in black) used in the quantitative analysis. Our two Plaza scenes feature a challenging indoor environment with difficult lighting conditions and fine structures. The Park scene offers a wide depth range in an outdoor environment.

precomputed background frame. For this we compute a background frame similar to other approaches which rely on an explicit background [SKC*19]: Particularly, for each pixel we compute the median over 10% of the pixel with the highest depth value. We feed these background images in ODS projection into our network to obtain an MSI representation of the background. For merging we exchange the RGB and volume density values of our estimated MSI frame by those of the background, when the accumulated transmittance T_i in ER projection is below 0.05 and a pixel's alpha value $T_i(1 - \exp(-\sigma_r \delta_r))$ is less than 0.01. This ensures we do not occlude foreground objects and also correctly handles disocclusions by filling in background information in the video, allowing rendered novel viewpoints to look behind dynamic moving objects.

4. Experimental Results

In the following, we provide more detailed insights of our optimization process for the depicted results as well as quantitative and qualitative evaluations of our approach when compared to other two recent approaches which also tackle the problem of novel view synthesis from ODS footage.

4.1. Data and implementation details

We recorded diverse scenes offering different challenging environments. Our recording setup consisted of a commercial Insta 360 Pro camera, which captured our stereo panorama videos in ODS projection at a resolution of 6400×3200 per eye. The videos feature two indoor (*Plaza1*, *Plaza2*) and one natural outdoor (*Park*). They offer different complexity in structures, movements of occluders, and depth range. Figure 4 provides cutouts of a representative frame for each scene. Video samples of our panorama sequences can be found in our supplemental material. For comparisons we also utilize the *Mall* scene of [MKK*20].

We used PyTorch [PGC*17, PGM*19] for our network implementation and optimization and OpenGL [WNDS99] in combination with CUDA for our renderer to achieve real-time novel view synthesis using a single NVIDIA GTX1080 GPU. For optimization we used a single NVIDIA GeForce RTX 3090. Due to GPU memory restrictions, our implementation estimates MSI layers with a resolution of 1200×600 pixel, but can easily adapt to arbitrary resolutions given more memory. For the two-folded optimization

procedure, our hyperparameters (λ_{data} , λ_{VGG} , λ_{disp} , λ_{empty}) are set to (0.1, 0.9, 100.0, 1.0). λ_{temp} and α_{temp} are only used during the second optimization phase and set to $\lambda_{\text{temp}} = 1.0$ and $\alpha_{\text{temp}} = 70.0$, respectively. We found that the above parameters are suitable for many different real-world sequences. When generating 16 MSI layers, an optimization for a sequence of 600 frames took between 15 and 17 h for 100 epochs of optimization. In our results, we first optimized for 100 epochs without temporal smoothness loss, and then another 100 epochs with it.

4.2. Face validity

To highlight the benefits of our ODS to MSI video transforming approach, we show various stereo viewpoint renderings of our method on various challenging real-world scenes. Video examples and comparisons can be seen in our supplemental video.

Figure 5 displays stereo results featuring head translations for our exemplary video recordings. Hereby, the viewing directions are fixed, whereas we move the camera from our MSI centre to the left/right. In the middle view the camera's origin lies exactly on the ODS camera circles. We added a vertical green line in our visualizations as a guide to make motion parallax and (dis-)occlusions more salient. Especially, the self (dis-)occlusions from the leaves in the *Plaza 1* scene (a with a plant very close to our ODS camera, highlight the dynamic motion parallax of our ODS to MSI translation approach. The *Park* scene (middle) shows that even for a wide range of depths our approach is able to evoke a natural feeling of depth.

To highlight the influence of our two-step optimization process, Figure 6 depicts a visualization of a single MSI layer of our *Park* scene for three consecutive frames. The initial training without temporal refinement shows frame-wise instability even within the static background, like the sky, mainly due to inaccurate depth information acquired from stereo correspondences. However, our fine-tuned ODS to MSI translation network is able to improve the temporal coherency for static parts of the video, while still allowing for change at moving foreground objects. This reduces temporal jittering artefacts during rendering.

4.3. Qualitative comparison

We visually compare our method with two similar state-of-the-art approaches for novel view synthesis from ODS footage.

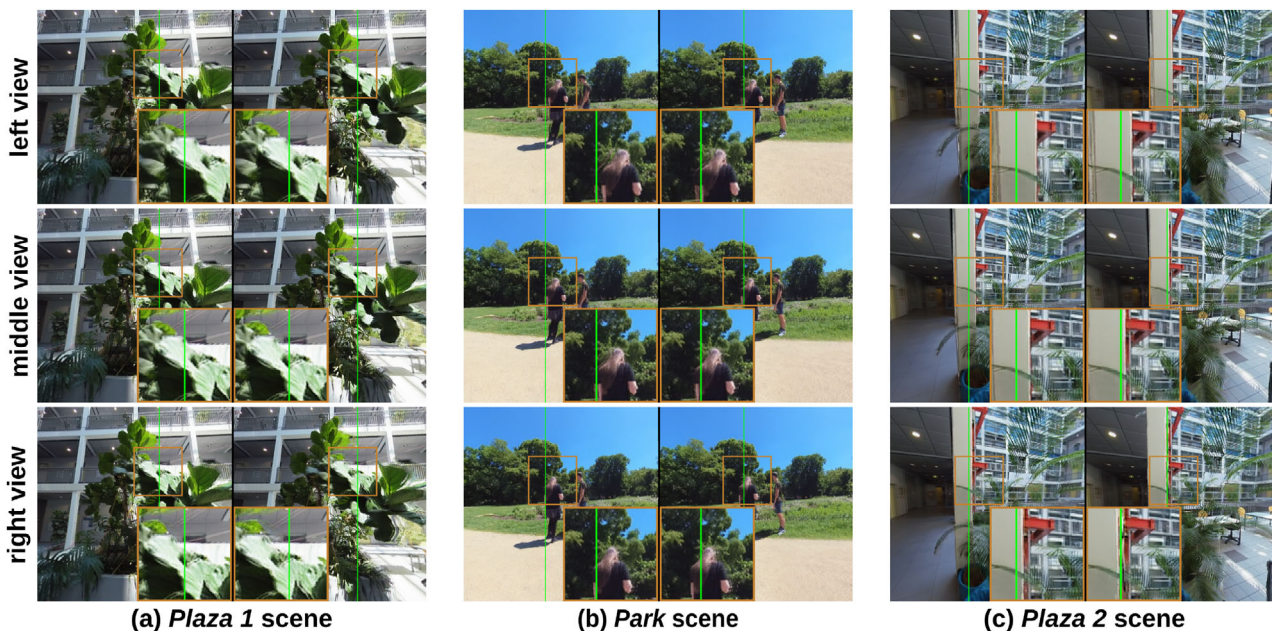


Figure 5: Stereo renderings from the multisphere image (MSI) representation for three different scenes. Hereby, the head position is shifted for each row, with the middle row taken from the MSI centre. We added a green vertical line to make the motion parallax and (dis-)occlusions more visible.

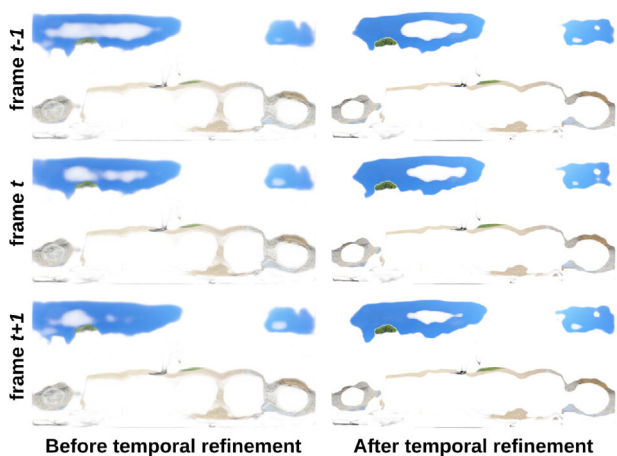


Figure 6: Visualization of a single MSI layer of the Park scene, before (left) and after (right) temporal refinement. Without temporal refinement, layers tend to quickly change (as seen in the sky), especially within frames with inaccurate depth information. This can lead to artefacts in the video where objects flicker between different MSI layers. Our temporal refinement improves the temporal coherence, only allowing smooth changes over time.

First, Figure 7 shows a comparison to the neural depth decoder (NDD) approach [MKK*20], which learns to inherently represent depth information with temporal consistency. For rendering, NDD utilizes a three-layered approach, similar to Serrano et al. [SKC*19]. In this comparison we used their *Mall* scene (see Figure 4) and op-

timized our translation network on it, as described in Section 3. We used the provided fully trained NDD network. Since NDD only learns to memorize depth information, the authors utilize the input ODS footage for rendering, whereas we directly estimate the **texture for layers**. While this leads to some loss in high frequencies in the images, the (dis-)occlusions of our results preserve cleaner silhouettes of the moving foreground objects. As can be seen on the right view in Figure 7, the front layer of the NDD rendering approach introduces salient and disturbing floating foreground pixel, which also reduces disocclusions of the background.

In Figure 8 we compare our approach to MatryODShka [ALG*20] on our *Plaza 2* scene. The authors also use ODS footage as input and produce an MSI representation, but they train to generalize to arbitrary scenes, while ours is an optimization approach on a per-scene-basis. Similar to NDD [MKK*20] they use the input ODS frames for rendering. Especially with close structures, our approach shows more natural motion parallax with self occlusion as seen at the leaves in the front. Furthermore, although MatryODShka utilizes 32 MSI layers, its results appear to have less (dis-) occlusion, as can be seen for the vertical red steel beams that are visible behind the pillar in the right view.

4.4. Quantitative comparison

For quantitative evaluation, we compare our method against ground truth, NDD [MKK*20], and MatryODShka [ALG*20]. Since MatryODShka is trained to generalize to arbitrary scenes while NDD and our approach are optimized, we compare the methods on the Replica dataset [SWM*19], which was used to train MatryODShka. Similar to our method, NDD is optimized per scene. However, their

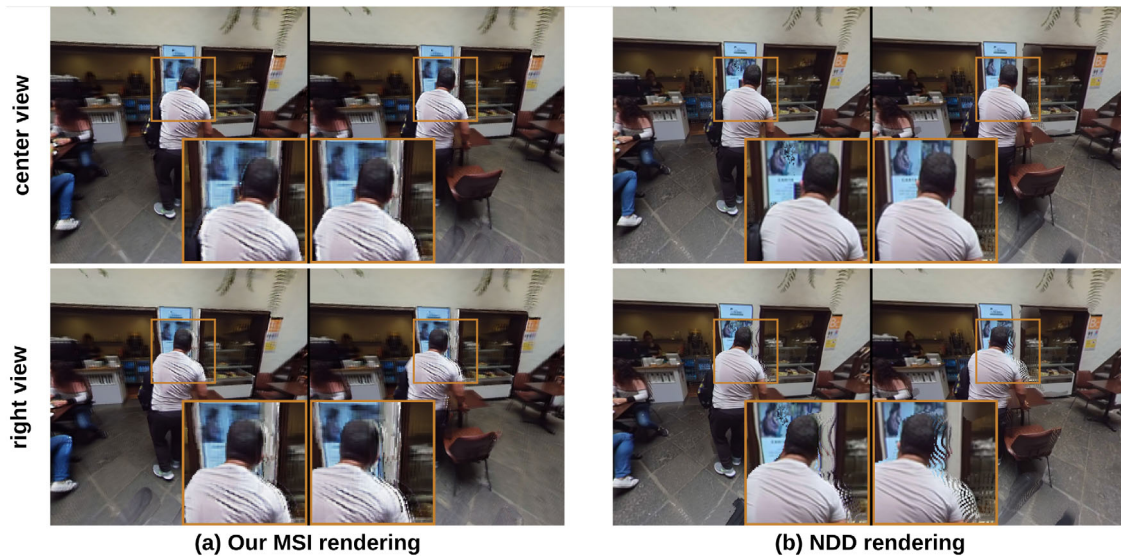


Figure 7: Comparison of stereo renderings of our approach and the neural depth decoder (NDD) layered rendering approach [MKK*20]. We used their trained neural depth decoder (NDD) network for their Mall scene, and also optimized our transformer network on this scene. While our approach has some troubles preserving high frequency information – since our network predicts the texture directly instead of utilizing the input frames for rendering – the (dis-)occlusions at depth discontinuities look more realistic. The NDD layered rendering approach introduces salient floating artefacts.



Figure 8: Comparison of stereo renderings of our approach and MatryODShka [ALG*20] on our Plaza 2 scene. Similar to NDD [MKK*20] rendering, MatryODShka [ALG*20] uses the input ODS images for their final rendering, which helps to preserve high frequencies. However, their approach shows less natural motion parallax as can be seen for the vertical red steel beams appearing in the right view (lower image row). Our approach show clear (dis-)occlusions at depth discontinuities and even self-occlusion of the plants leaves in the front. Our supplemental video includes an in motion comparison.

rendering technique is very different from our. Instead of utilizing MSIs for rendering, they use a three-layered rendering approach. Furthermore, both techniques, MatryODShka and NDD, only aim to reconstruct the depth and use the original images for rendering. In contrast, our method also optimizes for the texture. This dataset

consists of several static high-quality 3D indoor scenes that allow us to render arbitrary ground truth views. In particular, we drew a subset of 11 scenes for our comparisons, as depicted in Figure 4. For each scene we rendered a ground truth ODS projection at a resolution of 1200×600 for each eye, along with depths as input for each

Table 1: Quantitative comparison of our approach, neural depth decoder (NDD) approach [MKK*20], and MatryODShka [ALG*20] on a set of 11 Replica [SWM*19] ODS scenes at a resolution of 1200×600 . For each scene we randomly sampled 25 position within the ODS viewing circle (interpolated) and 50 position outside the viewing circle with a maximum distance of 0.1m (extrapolated). We use SSIM [WBSS04], PSNR and LPIPS [ZIE*18] error metrics. The values reported are the mean values \pm standard deviation over scenes on the averaged values over samples per scene.

		MatryODShka	NDD	Ours
SSIM \uparrow	interpolated	0.772 ± 0.035	0.809 ± 0.037	0.907 ± 0.032
	extrapolated	0.706 ± 0.062	0.716 ± 0.042	0.810 ± 0.043
	combined	0.739 ± 0.059	0.762 ± 0.061	0.859 ± 0.062
PSNR \uparrow	interpolated	22.038 ± 1.132	23.939 ± 1.631	29.552 ± 2.564
	extrapolated	19.194 ± 2.252	20.222 ± 1.866	24.051 ± 2.583
	combined	20.616 ± 2.268	22.081 ± 2.559	26.801 ± 3.773
LPIPS \downarrow	interpolated	0.255 ± 0.032	0.158 ± 0.039	0.149 ± 0.041
	extrapolated	0.340 ± 0.059	0.249 ± 0.045	0.214 ± 0.049
	combined	0.298 ± 0.063	0.204 ± 0.062	0.182 ± 0.055

method. Around the ODS centre we randomly sampled 25 positions within the viewing circle as well as 50 positions outside the viewing circle, with a maximum distance of 0.1m. This results in a total of 825 ER projection views for our comparisons.

Since the replica dataset only contains static scenes, we only utilized the first optimization phase for our approach with $(\lambda_{\text{data}}, \lambda_{\text{VGG}}, \lambda_{\text{disp}}, \lambda_{\text{empty}}) = (1.0, 0.0, 100.0, 1.0)$. The use of λ_{VGG} in static scenes resulted into faster overfitting, thus we opted for not using it for this kind of scenes. Our approach was optimized with 16 layers at depths in the range of $[0.5, 10.0]$. For MatryODShka, we used the same depth range with 32 MSI layers. For NDD rendering the same depth range was employed and for the optimization procedure the suggested default values were used.

In Table 1 we show the results for three evaluation metrics: SSIM [WBSS04], PSNR, and the perceptual metric

LPIPS [ZIE*18]. Our optimized method outperforms MatryODShka and NDD according to all these metrics in both interpolating views within the ODS circle as well as extrapolating views outside. Figure 9 shows an exemplary result of the comparisons in the *Office0* scene. While the overall parallax of MatryODShka and our method are similar, the generalized method can struggle at individual scene objects, like the whiteboard in the front or the chairs and table at the back. In contrast, NDD generates very high-quality results in terms of texture, but can fail to reflect the motion parallax correctly, as visible by the chairs. Our optimized approach specifically adapts to the scene, allowing for more natural motion parallax effects with less artefacts.

4.5. Limitations

The increased quality of the results does come at a cost, as the computation time required for our scene-dependent optimization method is comparatively higher than that of non-scene dependent approaches (e.g., MatryODShka [ALG*20]).

Our method sacrifices computational efficiency for the advantage of directly optimizing for the texture and, therefore, it opens the possibility to simultaneously utilize the capabilities of neural networks for scene dependent texture enhancement (e.g., inpainting). Thus, we believe that the added flexibility and possibilities of deep learning techniques outweigh the increased computation time.

5. Conclusion

We presented an optimization based approach to translate conventional omnidirectional stereo (ODS) video footage into an MSI representation. This representation offers full 6 DoF head motion support and can be rendered in real-time. Our method utilizes an encoder-decoder network as an optimization tool to estimate colour and density values for an arbitrary layer. Its twofolded optimization process generates temporal consistent MSI estimations. Additionally, our background MSI merging facilitates disocclusion filling for dynamic objects in the video. The experimental results demonstrate

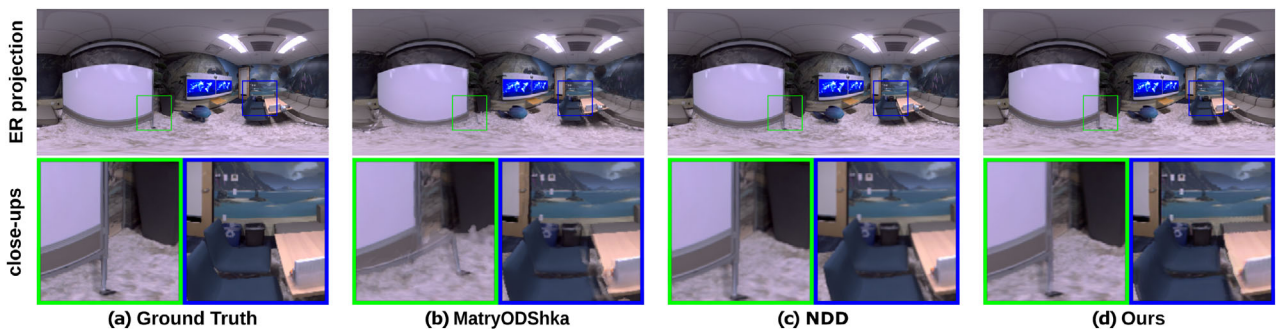


Figure 9: Exemplary result of the comparisons on the Replica [SWM*19] dataset. The depicted result was rendered at an extrapolated position, i.e. outside the viewing circle of the input ODS. While MatryODShka [ALG*20] captures the overall geometry of the scene, this generalized approach can struggle at individual scene objects, like the whiteboard and the table. Relying on the input frames for rendering NDD [MKK*20] has high-quality textures, but can sometimes fail to precisely capture the parallax as visible with the chairs. Our scene optimized approach adapts to the scene, allowing for more realistic motion parallax.

the appealing visual quality and natural motion parallax within our MSI videos.

Our work could be expanded to inherently store an MSI video representation within a network, completely removing the need for the input ODS video after optimization. Another possible extension would be a combination with real-time single image super resolution on the final output. This would allow to achieve higher spatial resolution for more pleasant VR experiences.

Acknowledgements

The authors gratefully acknowledge funding by the German Science Foundation (DFG MA2555/15-1 “Immersive Digital Reality” and DFG INST 188/409-1 FUGG “ICG Dome”).

Open access funding enabled and organized by Projekt DEAL.

References

- [AGB*16] ANDERSON R., GALLUP D., BARRON J. T., KONTKANEN J., SNAVELY N., HERNÁNDEZ C., AGARWAL S., SEITZ S. M.: Jump: virtual reality video. *ACM Transactions on Graphics* 35, 6 (Nov. 2016). <https://doi.org/10.1145/2980179.2980257>.
- [ALG*20] ATTAL B., LING S., GOKASLAN A., RICHARDT C., TOMPKIN J.: MatryODShka: Real-time 6DoF video view synthesis using multi-sphere images. In *European Conference on Computer Vision* (Cham, Aug. 2020), ECCV’20, Glasgow, UK, Springer International Publishing. https://doi.org/10.1007/978-3-030-58452-8_26.
- [BBM*01] BUEHLER C., BOSSE M., McMILLAN L., GORTLER S., COHEN M.: Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH ’01, Los Angeles, CA, USA, Association for Computing Machinery, pp. 425–432. <https://doi.org/10.1145/383259.383309>.
- [BCR19] BERTEL T., CAMPBELL N. D. F., RICHARDT C.: Mega-Parallax: Casual 360° panoramas with motion parallax. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 25, 5 (May 2019), 1828–1835.
- [BFO*20] BROXTON M., FLYNN J., OVERBECK R., ERICKSON D., HEDMAN P., DUVALL M., DOURGARIAN J., BUSCH J., WHALEN M., DEBEVEC P.: Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics* 39, 4 (Aug. 2020). <https://doi.org/10.1145/3386569.3392485>.
- [BMK*20] BERTEL T., MÜHLHAUSEN M., KAPPEL M., BITTNER P. M., RICHARDT C., MAGNOR M.: Depth augmented omnidirectional stereo for 6-DoF VR photography. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops* (May 2020), VRW’20, Atlanta, GA, USA, IEEE, pp. 660–661. <https://doi.org/10.1109/VRW50115.2020.00181>.
- [BWS05] BRUHN A., WEICKERT J., SCHNÖRR C.: Lucas/Kanade Meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision* 61 (Feb. 2005), 211–231.
- [BYLR20] BERTEL T., YUAN M., LINDROOS R., RICHARDT C.: OmniPhotos: Casual 360° VR photography. *ACM Transactions on Graphics* 39, 6 (Dec. 2020). <https://doi.org/10.1145/3414685.3417770>.
- [Eli94] ELLIS S. R.: What are virtual environments? *IEEE Computer Graphics and Applications (CG&A)* 14, 1 (1994), 17–22.
- [Epi] Epic Games: Unreal Engine for AR, VR & MR. <https://www.unrealengine.com/en-US/vr>. [Online; Last accessed 21-March-2023].
- [GMAFB19] GODARD C., MAC AODHA O., FIRMAN M., BROSTOW G. J.: Digging into self-supervised monocular depth estimation. In *IEEE/CVF International Conference on Computer Vision* (2019), ICCV’19, Seoul, Korea (South), IEEE, pp. 3827–3837. <https://doi.org/10.1109/ICCV.2019.00393>.
- [HASK17] HEDMAN P., ALSISAN S., SZELISKI R., KOPF J.: Casual 3D photography. *ACM Transactions on Graphics* 36, 6 (Nov. 2017). <https://doi.org/10.1145/3130800.3130828>.
- [HCCJ17] HUANG J., CHEN Z., CEYLAN D., JIN H.: 6-DOF VR videos with a single 360-camera. In *IEEE Virtual Reality* (Mar. 2017), VR’17, Los Angeles, CA, USA, IEEE, pp. 37–44. <https://doi.org/10.1109/VR.2017.7892229>.
- [HK18] HEDMAN P., KOPF J.: Instant 3D photography. *ACM Transactions on Graphics* 37, 4 (Jul. 2018). <https://doi.org/10.1145/3197517.3201384>.
- [HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2016), CVPR’16, Las Vegas, NV, USA, IEEE, pp. 770–778. <https://doi.org/10.1109/cvpr.2016.90>.
- [IH17] Intel, HypeVR Technology: 2017. [Online; Last accessed 21-March-2023]. <https://hypevr.com/>.
- [IHR*16] IM S., HA H., RAMEAU F., JEON H.-G., CHOE G., KWEON I. S.: All-around depth from small motion with a spherical panoramic camera. In *European Conference on Computer Vision* (Cham, 2016), ECCV’16, Amsterdam, The Netherlands, Springer International Publishing, pp. 156–172. https://doi.org/10.1007/978-3-319-46487-9_10.
- [IMS*17] ILG E., MAYER N., SAIKIA T., KEUPER M., DOSOVITSKIY A., BROX T.: FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2017), CVPR’17, Honolulu, HI, USA, IEEE, pp. 1647–1655. <https://doi.org/10.1109/CVPR.2017.179>.
- [IYT92] ISHIGURO H., YAMAMOTO M., TSUJI S.: Omni-directional stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 14, 2 (Feb. 1992), 257–262.

- [KAS*19] KOULIERIS G. A., AKŞIT K., STENDEL M., MANTIUK R. K., MANIA K., RICHARDT C.: Near-eye display and tracking technologies for virtual and augmented reality. *Computer Graphics Forum (CGF)* 38, 2 (May 2019), 493–519.
- [LFS*21] LI J., FENG Z., SHE Q., DING H., WANG C., LEE G. H.: MINE: Towards continuous depth MPI with NeRF for novel view synthesis. In *IEEE/CVF International Conference on Computer Vision (2021), ICCV'21*, Montreal, QC, Canada, IEEE, pp. 12558–12568. <https://doi.org/10.1109/ICCV48922.2021.01235>.
- [LH96] LEVOY M., HANRAHAN P.: Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, New Orleans, LA, USA, Association for Computing Machinery, pp. 31–42. <https://doi.org/10.1145/237170.237199>.
- [LXLL19] LAI P. K., XIE S., LANG J., LAGANIÈRE R.: Real-Time panoramic depth maps from omni-directional stereo images for 6 DoF videos in virtual reality. In *IEEE Conference on Virtual Reality and 3D User Interfaces* (Mar. 2019), VR'19, Osaka, Japan, IEEE, pp. 405–412. <https://doi.org/10.1109/VR.2019.8798016>.
- [LXRY18] LUO B., XU F., RICHARDT C., YONG J.-H.: Parallax360: Stereoscopic 360° scene representation for head-motion parallax. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 24, 4 (2018), 1545–1553.
- [MB95] McMILLAN L., BISHOP G.: Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, Los Angeles, CA, USA, Association for Computing Machinery, pp. 39–46. <https://doi.org/10.1145/218380.218398>.
- [MKK*20] MÜHLHAUSEN M., KAPPEL M., KASSUBECK M., BITTNER P. M., CASTILLO S., MAGNOR M.: Temporal consistent motion parallax for omnidirectional stereo panorama video. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology* (New York, NY, USA, 2020), VRST '20, Virtual Event, Canada, Association for Computing Machinery. <https://doi.org/10.1145/3385956.3418965>.
- [Moz18] Mozilla: Mozilla hubs, 2018. [Online; Last accessed 21-March-2023]. <https://hubs.mozilla.com/>.
- [MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARON J. T., RAMAMOORTHY R., NG R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision* (Cham, 2020), Vedaldi A., Bischof H., Brox T., Frahm J.-M., (Eds.), ECCV'20, Online, Springer International Publishing, pp. 405–421. https://doi.org/10.1007/978-3-030-58452-8_24.
- [OEE*18] OVERBECK R. S., ERICKSON D., EVANGELAKOS D., PHARR M., DEBEVEC P.: A system for acquiring, processing, and rendering panoramic light field stills for virtual reality. *ACM Transactions on Graphics* 37, 6 (Dec. 2018). <https://doi.org/10.1145/3272127.3275031>.
- [PGC*17] PASZKE A., GROSS S., CHINTALA S., CHANAN G., YANG E., DEVITO Z., LIN Z., DESMAISON A., ANTIGA L., LERER A.: Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff* (Long Beach, California, USA, 2017). <https://openreview.net/forum?id=BJJsrnfCZ>.
- [PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KÖPF A., YANG E., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: *PyTorch: An Imperative Style, High-performance Deep Learning Library*. NeurIPS'19, Vancouver, Canada. Curran Associates Inc., Red Hook, NY, USA, 2019, pp. 8024–8035. <https://doi.org/10.5555/3454287.3455008>.
- [PRS*18] PADMANABAN N., RUBAN T., SITZMANN V., NORCIA A. M., WETZSTEIN G.: Towards a machine-learning approach for sickness prediction in 360° stereoscopic videos. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 24, 4 (Apr. 2018), 1594–1603.
- [RPZSH13] RICHARDT C., PRITCH Y., ZIMMER H., SORKINE-HORNUNG A.: Megastereo: Constructing high-resolution stereo panoramas. In *IEEE Conference on Computer Vision and Pattern Recognition* (2013), CVPR'13, Portland, OR, USA, IEEE, pp. 1256–1263. <https://doi.org/10.1109/CVPR.2013.166>.
- [SKC*19] SERRANO A., KIM I., CHEN Z., DiVERDI S., GUTIERREZ D., HERTZMANN A., MASIA B.: Motion parallax for 360° RGBD video. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 25, 5 (May 2019), 1817–1827.
- [SRB10] SUN D., ROTH S., BLACK M. J.: Secrets of optical flow estimation and their principles. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), CVPR'10, San Francisco, CA, USA, IEEE, pp. 2432–2439. <https://doi.org/10.1109/CVPR.2010.5539939>.
- [SWM*19] STRAUB J., WHELAN T., MA L., CHEN Y., WIJMAN E., GREEN S., ENGEL J. J., MUR-ARTAL R., REN C., VERMA S., CLARKSON A., YAN M., BUDGE B., YAN Y., PAN X., YON J., ZOU Y., LEON K., CARTER N., BRIALES J., GILLINGHAM T., MUEGLER E., PESQUEIRA L., SAVVA M., BATRA D., STRASDAT H. M., NARDI R. D., GOESELE M., LOVEGROVE S., NEWCOMBE R. A.: The Replica dataset: A digital replica of indoor spaces. *CoRR abs/1906.05797* (2019). <http://arxiv.org/abs/1906.05797>, arXiv:1906.05797.
- [SZ15] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015), Bengio Y., LeCun Y. (Eds.). <http://arxiv.org/abs/1409.1556>.
- [TBLG16] THATTE J., BOIN J.-B., LAKSHMAN H., GIROD B.: Depth augmented stereo panorama for cinematic virtual reality with head-motion parallax. In *IEEE International Conference on Multimedia and Expo* (July 2016), ICME'16, Seattle, WA, USA, IEEE, pp. 1–6. <https://doi.org/10.1109/ICME.2016.7552858>.

- [TG18] THATTE J., GIROD B.: Towards perceptual evaluation of six degrees of freedom virtual reality rendering from stacked OmniStereo representation. In *Proceedings IS&T International Symposium on Electronic Imaging: Photography, Mobile, and Immersive Imaging* (2018), vol. 30 of *EI'18, Burlingame, CA, USA*, Society for Imaging Science and Technology, pp. 352–1–6. <https://doi.org/10.2352/ISSN.2470-1173.2018.05.PMII-352>.
- [TLWG17] THATTE J., LIAN T., WANDELL B., GIROD B.: Stacked omnistereos for virtual reality with six degrees of freedom. In *IEEE Visual Communications and Image Processing* (Dec. 2017), VCIP'17, St. Petersburg, FL, USA, IEEE, pp. 1–4. <https://doi.org/10.1109/VCIP.2017.8305085>.
- [TRB*20] TOMOTO Y., RAO S., BERTEL T., CHANDE K., RICHARDT C., HOLZER S., ORTIZ-CAYON R.: Casual real-world VR using light fields. In *SIGGRAPH Asia 2020 Posters* (New York, NY, USA, 2020), SA '20, Virtual Event, Republic of Korea, Association for Computing Machinery. <https://doi.org/10.1145/3415264.3425452>.
- [Ven16] VENTURA J.: Structure from motion on a sphere. In *European Conference on Computer Vision* (Cham, 2016), Leibe B., Matas J., Sebe N., Welling M., (Eds.), ECCV'16, Amsterdam, The Netherlands, Springer International Publishing, pp. 53–68. https://doi.org/10.1007/978-3-319-46487-9_4.
- [WBSS04] WANG Z., BOVIK A., SHEIKH H., SIMONCELLI E.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.
- [WLLZ20] WANG M., LYU X.-Q., LI Y.-J., ZHANG F.-L.: VR content creation and exploration with deep learning: A survey. *Computational Visual Media* 6 (March 2020), 3–28.
- [WLZ*20] WANG M., LI Y.-J., ZHANG W.-X., RICHARDT C., HU S.-M.: Transitioning360: Content-aware NFoV virtual camera paths for 360° video playback. In *2020 IEEE International Symposium on Mixed and Augmented Reality* (2020), ISMAR'20, Porto de Galinhas, Brazil, IEEE, pp. 185–194. <https://doi.org/10.1109/ISMAR50242.2020.00040>.
- [WNDS99] WOO M., NEIDER J., DAVIS T., SHREINER D.: *OpenGL Programming Guide: The Official Guide to Learning OpenGL, version 1.2*, 3rd edition, Addison-Wesley Longman Publishing Co., Inc., USA, 1999.
- [WSGD18] WEGNER K., STANKIEWICZ O., GRAJEK T., DOMAŃSKI M.: Depth estimation from stereoscopic 360-degree video. In *IEEE International Conference on Image Processing* (Oct. 2018), ICIP'18, Athens, Greece, IEEE, pp. 2945–2948. <https://doi.org/10.1109/ICIP.2018.8451452>.
- [XHKK21] XIAN W., HUANG J.-B., KOPF J., KIM C.: Space-time neural irradiance fields for free-viewpoint video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), CVPR'21, Nashville, TN, USA, IEEE, pp. 9416–9426. <https://doi.org/10.1109/CVPR46437.2021.00930>.
- [ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), CVPR'18, Salt Lake City, UT, USA, IEEE, pp. 586–595. <https://doi.org/10.1109/CVPR.2018.00068>.
- [ZTF*18] ZHOU T., TUCKER R., FLYNN J., FYFFE G., SNAVELY N.: Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics* 37, 4 (Jul 2018). <https://doi.org/10.1145/3197517.3201323>.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Supporting Information