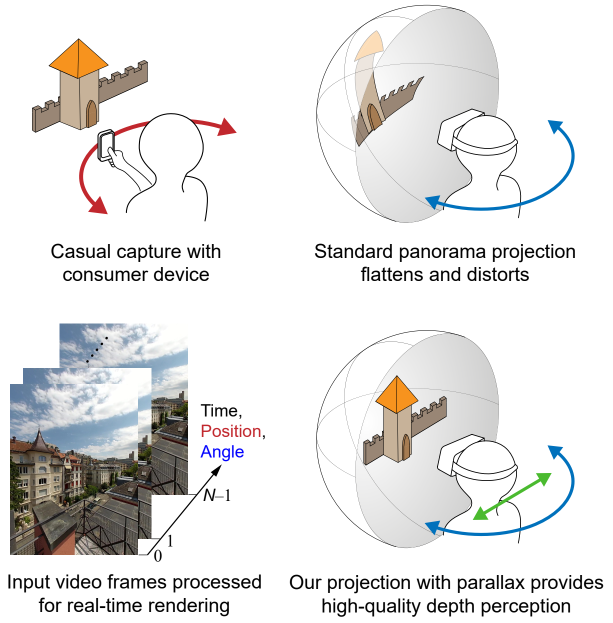
MegaParallax: Casual 360° Panoramas with Motion Parallax

Tobias Bertel, Neill D. F. Campbell, and Christian Richardt, *Member, IEEE*


Casual capture with consumer device
Standard panorama projection flattens and distorts
Time,
Position,
Angle
N–1
1
0
Input video frames processed for real-time rendering
Our projection with parallax provides high-quality depth perception

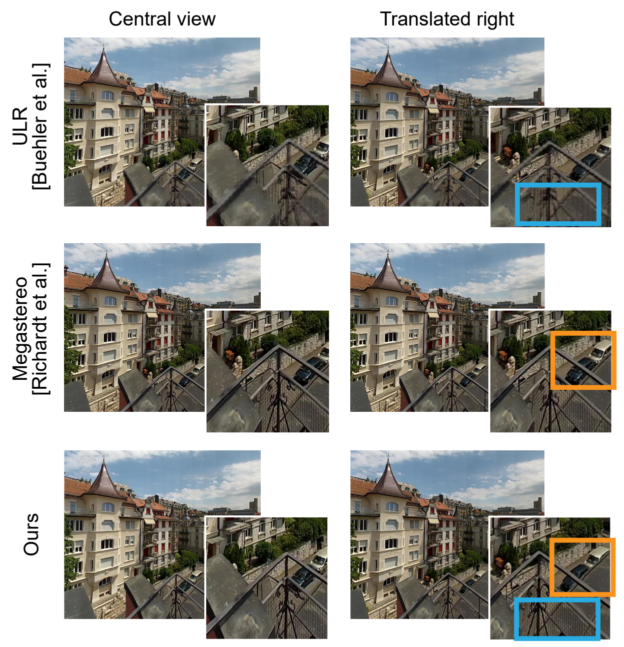


Fig. 1. The rise of consumer cameras enabled ubiquitous capture of 360° panoramas as users want to share and relive experiences. However, existing projection techniques flatten the panorama and remove motion parallax – an important depth cue for the human visual system. We present a new method that provides high-quality parallax rendering that restores this depth cue and allows the viewer to explore static scenes with translation as well as rotation. Our method is able to correctly occlude the cars as the viewer translates to the right (orange box), and improves rendering quality over previous techniques (blue box).

Abstract—The ubiquity of smart mobile devices, such as phones and tablets, enables users to casually capture 360° panoramas with a single camera sweep to share and relive experiences. However, panoramas lack motion parallax as they do not provide different views for different viewpoints. The motion parallax induced by translational head motion is a crucial depth cue in daily life. Alternatives, such as omnidirectional stereo panoramas, provide different views for each eye (binocular disparity), but they also lack motion parallax as the left and right eye panoramas are stitched statically. Methods based on explicit scene geometry reconstruct textured 3D geometry, which provides motion parallax, but suffers from visible reconstruction artefacts. The core of our method is a novel multi-perspective panorama representation, which can be casually captured and rendered with motion parallax for each eye on the fly. This provides a more realistic perception of panoramic environments which is particularly useful for virtual reality applications. Our approach uses a single consumer video camera to acquire 200–400 views of a real 360° environment with a single sweep. By using novel-view synthesis with flow-based blending, we show how to turn these input views into an enriched 360° panoramic experience that can be explored in real time, without relying on potentially unreliable reconstruction of scene geometry. We compare our results with existing omnidirectional stereo and image-based rendering methods to demonstrate the benefit of our approach, which is the first to enable casual consumers to capture and view high-quality 360° panoramas with motion parallax.

Index Terms—Casual 360° scene capture, plenoptic modeling, image-based rendering, novel-view synthesis, virtual reality

1 Introduction

Capturing 360° panoramas has become straightforward as this functionality is implemented on every phone. However, standard (monoscopic) panoramas appear flat when viewed in virtual reality headsets. This is because panoramas do not provide important depth cues such as *binocular disparity* (different views for our left and right eyes) or *motion parallax* (different views as the viewpoint changes). Binocular disparity and motion parallax are important cues for correctly perceiving depth [14] and feeling *presence* in virtual environments [33]. We propose a new approach for capturing largely static real-world environments and rendering high-quality, immersive 360° panoramas that provide both binocular disparity and motion parallax (see Figure 1).

Most techniques for capturing panoramas for virtual reality build on omnidirectional stereo [2, 23, 26, 29]. These approaches stitch multiple views of a moving camera or from a multi-camera rig into separate left- and right-eye panoramas. This way, omnidirectional stereo provides different views for each eye (binocular disparity), but it cannot provide any motion parallax to a viewer. Moreover, the rendered views exhibit vertical distortion resulting in curved lines, most notably for nearby

• *Tobias Bertel, Neill D. F. Campbell and Christian Richardt are with the Department of Computer Science at the University of Bath, UK. E-mail: T.B.Bertel@bath.ac.uk, N.Campbell@bath.ac.uk, christian@richardt.name.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

scene objects. Huang et al. [15] introduce parallax into a 360° image using depth-based warping; however, the visual quality of their results is limited by easily visible warping artefacts. Luo et al. [18] propose another 360° scene representation with motion parallax based on image warping, but motion parallax is only achieved near the captured viewpoints, and they require thousands of input images captured with a robotic arm, which prevents casual capture. Hedman et al. [13] instead reconstruct textured 3D geometry that provides motion parallax directly. Their approach strongly relies on dense depth estimation, which suffers from reconstruction errors in traditionally challenging cases such as uniformly coloured regions or highly detailed geometry.

We introduce an end-to-end pipeline and specifically tailored techniques for turning a casually captured monoscopic video into an enriched 360° panoramic experience with motion parallax that can be explored in real time. Our new approach generalises omnidirectional stereo techniques to free-viewpoint image-based rendering. Specifically, our new novel-view synthesis approach (Section 6) computes each pixel's colour based on its optimal pair of input images, which are combined using view-dependent flow-based blending. These contributions make it possible for casual consumers to capture and view high-quality 360° panoramas with motion parallax in real time.

2 Related work

Our work builds on the extensive literature on panorama stitching, omnidirectional stereo, 3D reconstruction, novel-view synthesis, light fields and image-based rendering. Here, we briefly discuss a selection of related work from each of these areas. Like most techniques, we assume a static environment.

Image and video stitching Stitching of panoramic images [34] and 360° videos [25] has become widely available and can be found in consumer cameras. However, stitching aims to fuse different views into a single spherical view of the world by overcoming the parallax between input views. This removes important depth cues and results in a flat appearance. Recent work introduces binocular disparity by warping using structure-from-motion [15] or stitching the views of two 360° cameras [19]; both suffer from warping and stitching artefacts, without providing any motion parallax.

Omnidirectional stereo Techniques for omnidirectional stereo are stitching stereoscopic multi-perspective panoramas [36] from a moving camera [16, 23, 26] or multi-camera rigs [2, 29], but suffer from vertical distortion [31] and lack of motion parallax [2, 13]. Thatte et al. [35] proposed adding depth maps to enable motion parallax, but estimating depth maps for general scenes remains challenging. Our approach extends multi-perspective panoramas by introducing novel-view synthesis with motion parallax.

Light fields By representing a densely-sampled subset of the plenoptic function [1], light fields enable the synthesis of novel views with binocular disparity and motion parallax [11, 17]. In practice, the high sampling density required by light fields prevents casual scene capture, as it necessitates special camera hardware [8, 22] or user guidance during capture [4, 7]. Concentric panoramas [31] reduce the dimensionality of light fields, but are difficult to capture in the real world. Like omnidirectional stereo, concentric mosaics also suffer from vertical distortion, which is difficult to compensate as it depends on depth. Overbeck et al. [22] present a full pipeline for capturing, preprocessing, transmission and rendering of light field stills. They use a revolving rig to capture about a thousand images and present a novel per-pixel splatting algorithm built on view-dependent geometry.

3D reconstruction Existing approaches for VR photography, such as Layered Depth Panoramas [37] and Casual 3D Photography [13], extend off-the-shelf structure-from-motion and multi-view stereo techniques [10, 27] for recovering camera poses and scene geometry, which enables rendering of accurate motion parallax. However, the reconstruction of accurate 3D geometry of arbitrary, unconstrained environments remains challenging, as reconstruction techniques often fail in poorly or repetitively textured scenes. This results in errors in the reconstructed geometry. The layered geometry that is reconstructed limits the visual quality of disocclusions that are caused by motion parallax, compared to our approach which handles disocclusions naturally by sampling suitable image pixels from a dense ray database. Outdoor environments are also particularly difficult because of their large size relative to the distance between camera viewpoints. Nevertheless, more accurate proxy scene geometry will further improve the quality of our image-based novel-view synthesis.

Image-based rendering (IBR) As a hybrid approach between 3D reconstruction and light-field rendering [32], IBR builds on coarse global 3D geometry [4, 9, 15] or per-view local 3D geometry [5, 12, 18] to synthesise realistic novel views from warped input images, with correct motion parallax. In doing so, IBR optimises for the visual result and does not rely on accurate geometry reconstruction, which is often fragile in practice. IBR thus also works in cases where dense 3D reconstruction fails. IBR approaches have demonstrated high-quality results thanks to local warps [5, 9] and geometry reconstruction using a depth sensor [12]. As with light fields, this work is so far constrained to rather limited capture volumes with outside-in camera arrangements, and does not extend to general outdoor environments with distant objects. Luo et al.'s Parallax360 approach [18] supports view synthesis with motion parallax for 360° panoramas, but only on a spherical surface of captured views, not *inside* the volume like our approach.

Novel-view synthesis At the core of image-based rendering lies the synthesis of novel viewpoints from a set of captured input views [6, 30]. View synthesis requires correspondence between input views – either explicit like depth maps [e.g. 24], implicit like using optical flow [e.g. 18], or learned [e.g. 38]. We use flow-based novel-view synthesis, as it produces high-quality results without requiring explicit or learned correspondence. Most previous flow-based interpolation techniques are limited to synthesising novel views on a line between two viewpoints [30] or within a triangle of viewpoints [18]. Our view synthesis technique is not restricted to view interpolation on the capturing surface, but extrapolates new viewpoints using ray-based view synthesis.

Most related techniques Our approach is most related to omnidirectional stereo techniques [2, 16, 23, 26, 29], which do, however, not provide any motion parallax. Parallax360 [18] produces motion parallax with image-based rendering, but only for viewpoints close to the sphere of captured views. In particular, unlike our approach, theirs cannot achieve motion parallax inside the captured sphere. Casual 3D Photography [13] achieves motion parallax by virtue of its layered 3D reconstruction. However, this also limits the visual quality of disocclusions compared to our image-based novel-view synthesis, which does not rely on explicit scene reconstruction.

3 Overview

Our approach can generate 360° datasets containing motion parallax from hand-held video input without active guidance. We employ image-based novel-view synthesis that produces high-quality, high-resolution views with motion parallax in real time. Our approach does not rely on any explicit scene geometry apart from a very coarse scene proxy such as a plane or a cylinder. This positions our method naturally between plenoptic and image-based rendering approaches [4, 20].Our approach starts from videos that are casually captured with consumer cameras on approximately circular paths. We process this footage to generate multi-perspective panoramic datasets that can then be viewed with binocular disparity and motion parallax thanks to high-quality, real-time, image-based rendering. We next outline the steps of our approach (see also Figure 2) and give details in Sections 4 to 6.

Capture (Section 4) To capture sufficient motion parallax within 360° panoramas, we assume a sweeping camera motion on a roughly circular trajectory with radially outward-looking viewing directions. We also assume the scene is static. Such videos can be captured *casually* by end-users, and we show results for datasets we captured with different consumer cameras like a GoPro Hero 4 and an Insta360 One.

Preprocessing (Section 5) We first reconstruct camera poses for all video frames using structure-from-motion [27]. We manually select a single captured ring of reconstructed cameras and register them to an ideal circular trajectory. This imposes an ordering on the cameras that facilitates the identification of neighbouring viewpoints. We finally

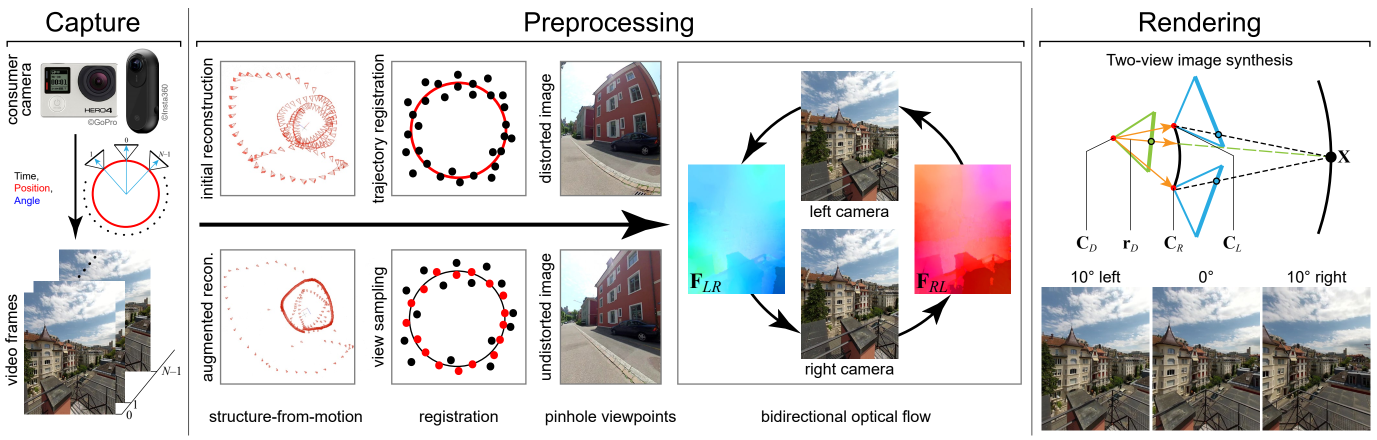


Fig. 2. Overview of our approach. **Capture:** We acquire an input video with a hand-held consumer camera (Section 4). **Preprocessing:** Compute per-frame camera poses using structure-from-motion in a two-step reconstruction procedure (Section 5.1). The views are registered to an ideal circular camera trajectory (Section 5.2) and sampled for uniform angular spacing (Section 5.3). We then compute optical flow between each pair of adjacent cameras to establish dense correspondences (Section 5.4). **Rendering:** The rendering is formulated per pixel using a novel two-view view-synthesis approach that uses flow-based blending based on position and viewing direction of the novel view and the input views (Section 6).

compute bidirectional optical flow [3] between each pair of adjacent cameras, which will enable live view synthesis during rendering.

Rendering (Section 6) At run time, our approach performs on-the-fly view synthesis in real time. First, for every pixel, we identify the optimal pair of camera views given the ray from the target viewpoint to the pixel. Then, each pixel's colour is interpolated from its pair of optimal views, depending on both the ray through the pixel and the position of the desired view. For this interpolation, we propose a novel view-dependent flow-based blending technique tailored for our image-based rendering method.

4 Capture

The starting point of our approach is the capture of a single input video, which is recorded with a moving consumer camera. For 360° omnidirectional stereo panoramas, the ideal camera trajectory has been shown to be a circle with cameras pointing radially outwards [23]; the same applies to our panoramas with motion parallax. We assume the camera to be calibrated intrinsically, but we need to estimate camera poses for the moving camera to accurately synthesise novel views in subsequent stages of our approach.

In theory, the camera poses can be obtained using off-the-shelf techniques like SLAM [e.g. 21] or structure-from-motion [e.g. 27]. However, for the inside-out camera configuration we are using, these techniques face two main problems: (1) pairwise camera baselines are small compared to scene depth, which results in ill-posed triangulations that lead to tracking failures, and (2) most cameras have no pairwise overlap as they are pointing in opposite directions, which often leads to loop closure problems. In addition, it is difficult to produce perfectly closing video loops with hand-held cameras because of natural variability in camera pose over time. For these reasons, camera geometry cannot always be reconstructed reliably and we only show results for which the reconstruction in Section 5.1 succeeded. A detailed examination of the influence of camera paths, camera intrinsics and scene characteristics on reconstructibility is beyond the scope of this paper. Hedman et al. [13] report success with fish-eye lenses that increase the overlap between views, but such lenses are not usually found on most consumer cameras. Nonetheless, we believe that additional work on 3D reconstruction from narrow-baseline inside-out imagery is required for creating casual user-centric VR and AR experiences more reliably.

5 Preprocessing

In this section, we take as input a video captured using the capturing process described in the previous section, and prepare it for our real-time image-based rendering technique in Section 6. We prepare a 360° dataset with motion parallax via the following steps:

1. Reconstruction of camera geometry (Section 5.1).
2. Registration of cameras to the ideal trajectory (Section 5.2).
3. Sampling of cameras based on the ideal trajectory (Section 5.3).
4. Computation of optical flow between cameras (Section 5.4).

5.1 Reconstruction of camera geometry

We obtain the sparse reconstruction of scene geometry and camera poses using COLMAP [27], assuming fixed camera intrinsics. Because of small camera baselines and limited overlap between most views, we perform reconstruction in two stages.

First, we perform a reconstruction from only a set of keyframes, for example every tenth frame of the input video (depending on capturing speed and circle radius), which yields sufficiently large camera baselines to increase the robustness of the reconstruction procedure.

In a second stage, we then *register* all video frames to the existing scene model, followed by bundle adjustment. This improves the conditioning of the pose estimation of the densely sampled video frames, resulting in more consistent reconstructions. We then undistort the input video frames using the known intrinsic calibration parameters, giving us fully-calibrated pinhole viewpoints.

5.2 Trajectory registration

We next register the reconstructed camera poses to an idealised continuous camera trajectory, in our case a circle. This imposes an ordering on the cameras, making it easy to find adjacent cameras and to index them linearly using the polar angle $\varphi \in [0, 360)$. We show results for circular trajectories, but more general paths are possible as well.

We use the centroid of all camera centres as the centre of the circle and origin of our coordinate system. The circle radius r is set to the average distance of camera centres from the origin. We rescale our dataset to match the real physical set-up when the radius is known, and use $r = 0.8$ m for hand-held datasets. We next fit a plane to the camera centres to obtain the normal direction $\mathbf{n}$ of the circle. We compute the polar angle φ_i for each camera i by first projecting the camera centre $\mathbf{C}_i$ and the x-axis onto the plane of the circle:

$$\mathbf{C}_i^* = \mathbf{C}_i - \mathbf{n} \cdot (\mathbf{C}_i \cdot \mathbf{n}) \tag{1}$$

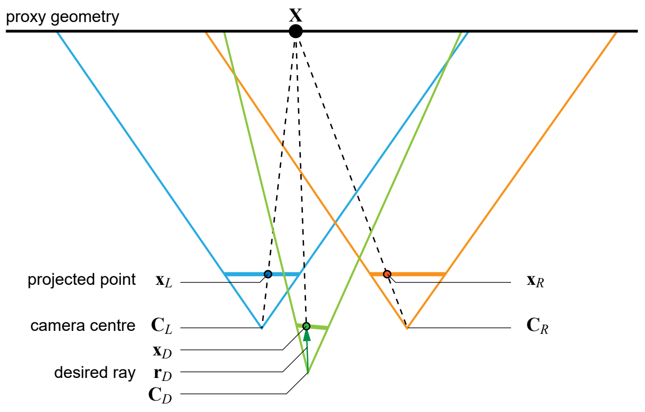
$$\mathbf{x}^* = [1,0,0]^\top - \mathbf{n} \cdot ([1,0,0]^\top \cdot \mathbf{n}) \tag{2}$$

and then obtain the signed angle between their directions using

$$\varphi_i = \operatorname{atan2}\big((\mathbf{x}^* \times \mathbf{C}_i^*) \cdot \mathbf{n},\, \mathbf{x}^* \cdot \mathbf{C}_i^*\big). \tag{3}$$

5.3 Camera sampling

Given the registered and parametrised camera trajectory, we next sample a subset of cameras that are approximately uniformly spaced in parametric space. This ensures that input views sample the entire environment as uniformly as possible, regardless of the speed at which the camera was moving during the capture process. For our datasets,



proxy geometry
X
projected point $\mathbf{x}_L$
$\mathbf{x}_R$
camera centre $\mathbf{C}_L$
$\mathbf{C}_R$
$\mathbf{x}_D$
desired ray $\mathbf{r}_D$
$\mathbf{C}_D$

Fig. 3. Camera geometry for two-view novel-view synthesis of the novel view (green) using the left (blue) and the right views (orange). The desired ray $\mathbf{r}_D$ from the camera centre $\mathbf{C}_D$ through the pixel $\mathbf{x}_D$ intersects the proxy geometry at a point $\mathbf{X}$. This point is then projected into the left and right cameras, yielding pixels for blending ($\mathbf{x}_L$ and $\mathbf{x}_R$, respectively).

we use an angular baseline β of one to two degrees, corresponding to about 180 to 360 camera viewpoints per circle.

We found empirically that the depth structure and scene appearance of the captured environment have a great influence on the minimum required sampling frequency. For example, a low sampling frequency is sufficient for textureless or distant regions, such as a blue sky or distant mountains, as motion parallax will be imperceptible and view synthesis can still be reliably performed on wider baselines. On the other hand, the appearance of thin and nearby objects is improved by increasing the sampling frequency. This suggests that an adaptive, scene-dependent sampling strategy would be a useful extension.

5.4 Optical flow

In our image-based novel-view synthesis (see Section 6), each pixel's colour is computed from a pair of adjacent cameras. We use the trajectory's parametrisation (φ_i) to identify all pairs of adjacent cameras, and then compute dense correspondences between each pair of images using bidirectional optical flow [3].

We precompute the optical flow fields to minimise computation requirements at run time during the rendering. To reduce computation times during preprocessing, as well as memory usage and disk bandwidth during rendering, we compute the flow at half the image resolution. This produces visual results comparable to full-resolution flows but is significantly more efficient.

6 Rendering

The goal of the final rendering stage is to synthesise novel views in real time within a viewing area inside the circle of captured views. We first present our new per-pixel image-based rendering technique that exploits known camera geometry and dense correspondences to interpolate views from a pair of cameras (Section 6.1). We then describe how to find the optimal camera pair for view synthesis within the hundreds of cameras of typical panoramic datasets (Section 6.2). We analyse the maximum size of the viewing area supported by our view-synthesis approach in Section 8.

6.1 Novel-view synthesis from two cameras

In this section, we introduce the core building block of our image-based rendering approach: novel-view synthesis from two input views using view-dependent flow-based blending. We assume that the pair of closest cameras is given in this section, and discuss the selection of the camera pair in Section 6.2.

We assume that we are given the two input images $\mathbf{I}_L$ and $\mathbf{I}_R$, corresponding to the left and right cameras of the camera pair, and we want to synthesise the image $\mathbf{I}_D$ of the desired camera view. As illustrated in Figure 3, these cameras are defined by their camera centres $\mathbf{C}_L$, $\mathbf{C}_R$ and $\mathbf{C}_D$, and their orientations.

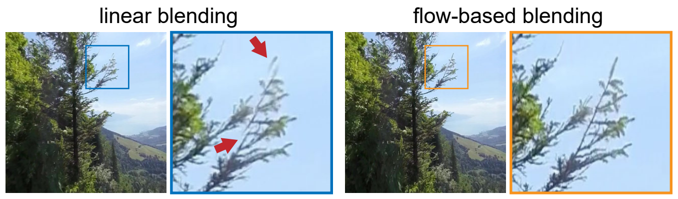


Fig. 4. Linear blending often results in ghosting artefacts (see red arrows), which are reduced with our view-dependent flow-based blending.

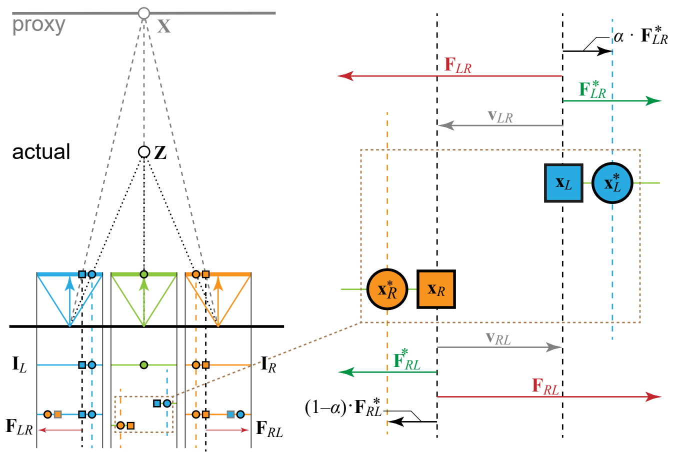


Fig. 5. View-dependent flow-based blending (for $\alpha = 0.5$). **Left:** A scene point $\mathbf{Z}$ (black circle)is imaged at $\mathbf{x}_D$ (green filled circle) and approximated by $\mathbf{X}$ (grey outline circle). $\mathbf{X}$ projects onto $\mathbf{x}_L$ and $\mathbf{x}_R$ (filled squares). Note that, in practice, scene points and proxy are much further away (several metres) compared to the scene baseline (a few cm) than shown. **Right:** Image plane of the novel view. Optical flow $\mathbf{F}_{LR}$ (upper red arrow) is used to account for wrongly re-projected pixels, e.g. $\mathbf{x}_L$ ($\mathbf{F}_{RL}$ analogously). Blending is performed with flow-adjusted pixels $\mathbf{x}^*_L$ and $\mathbf{x}^*_R$.

6.1.1 Linear blending

We synthesise the novel view $\mathbf{I}_D$ per pixel, so that it can be efficiently computed in parallel on the GPU. In our approach, we use a planar proxy geometry that is fixed at a given distance in front of the desired camera, and we thus do not require accurately estimated scene depth. To compute the colour of a pixel $\mathbf{x}_D$ in the desired view, we first obtain the world point $\mathbf{X}$ on the proxy geometry that projects to $\mathbf{x}_D$ in the desired camera (see diagram in Figure 3) by rasterising the proxy geometry using OpenGL. The world point $\mathbf{X}$ projects to $\mathbf{x}_L$ and $\mathbf{x}_R$ in the left and right cameras, respectively. For a baseline *linear blending*, we combine the colours sampled from the left and right images, $\mathbf{I}_L$ and $\mathbf{I}_R$, at $\mathbf{x}_L$ and $\mathbf{x}_R$, respectively, using a convex combination:

$$\mathbf{I}_D(\mathbf{x}_D) = (1-\alpha)\cdot\mathbf{I}_L(\mathbf{x}_L) + \alpha\cdot\mathbf{I}_R(\mathbf{x}_R), \tag{4}$$

where $\alpha \in [0,1]$ is the blending weight between the views for pixel $\mathbf{x}_D$, which we define in Section 6.2. This simple linear blending baseline often results in ghosting artefacts, as shown in Figure 4.

6.1.2 View-dependent flow-based blending

Coarse proxy geometry generally causes large re-projection errors that lead to blurry images [4, 9]. Richardt et al. [26] introduced flow-based ray interpolation to overcome this problem and synthesise high-quality panoramas on a cylindrical imaging surface. We extend their approach to synthesise novel views for desired viewpoints that are not constrained to the predefined viewing circles of the omnidirectional stereo framework or the camera circle [18]. To this end, we exploit the camera geometry and the precomputed dense optical flow (Section 5.4) to determine *view-dependent* image coordinates $\mathbf{x}^*_L$ and $\mathbf{x}^*_R$, at which to sample the left and right images (see Figure 5 and compare to Equation 4):

$$\mathbf{I}_D(\mathbf{x}_D) = (1-\alpha)\cdot\mathbf{I}_L(\mathbf{x}^*_L) + \alpha\cdot\mathbf{I}_R(\mathbf{x}^*_R). \tag{5}$$

We use precomputed optical flow (Section 5.4) to compensate for the depth mismatch between an imaged scene point $\mathbf{Z}$ and its approximation

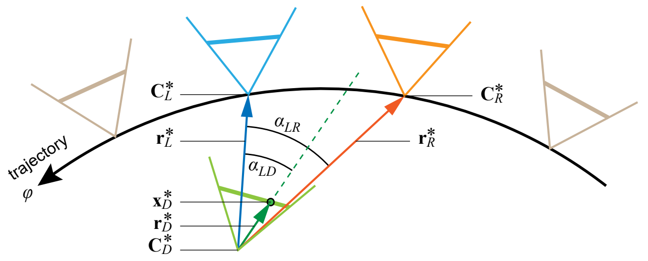


Fig. 6. Computation of the blending weight α using angles between rays.

by the proxy point $\mathbf{X}$, which is particularly large for nearby objects. This scenario is illustrated in Figure 5. Specifically, we use the plane-induced displacement between the projections $\mathbf{x}_L$ and $\mathbf{x}_R$:

$$\mathbf{v}_{LR} = \mathbf{x}_R - \mathbf{x}_L \quad \text{and} \quad \mathbf{v}_{RL} = \mathbf{x}_L - \mathbf{x}_R, \tag{6}$$

which are shown in grey, and the optical flow fields $\mathbf{F}_{LR}$ and $\mathbf{F}_{RL}$ (in red), to compute a local flow displacement (in green):

$$\mathbf{F}^*_{LR}(\mathbf{x}_L) = \mathbf{v}_{LR} - \mathbf{F}_{LR}(\mathbf{x}_L) \quad \text{and} \tag{7}$$

$$\mathbf{F}^*_{RL}(\mathbf{x}_R) = \mathbf{v}_{RL} - \mathbf{F}_{RL}(\mathbf{x}_R). \tag{8}$$

By scaling these displacements with the view-dependent blending weight $\alpha \in [0,1]$ that accounts for the pose of the desired camera, we compute image coordinates that effectively compensate for the depth mismatch between scene and proxy geometry (black arrows):

$$\mathbf{x}^*_L = \mathbf{x}_L + \alpha * \mathbf{F}^*_{LR}(\mathbf{x}_L) \quad \text{and} \tag{9}$$

$$\mathbf{x}^*_R = \mathbf{x}_R + (1-\alpha) * \mathbf{F}^*_{RL}(\mathbf{x}_R). \tag{10}$$

The magnitude of the local flow displacements $\mathbf{F}^*_{LR}$ and $\mathbf{F}^*_{RL}$ provides an indication of the distance of scene points: small displacements indicate that scene points are close to the proxy geometry, while large displacements generally indicate nearby scene objects. Figure 4 shows the benefit of our view-dependent flow-based blending approach. Note that ghosting artefacts occur more frequently in image areas with higher re-projection errors.

6.2 Finding the best camera pair

Our panoramic datasets comprise hundreds of input views. Instead of computing the convex combination of potentially hundreds of views, as in unstructured lumigraph rendering [4], we identify the best pair of adjacent cameras L and R to synthesise each pixel $\mathbf{x}_D$ of the novel view D. For this purpose, we consider the angles between the ray $\mathbf{r}_D$ to be synthesised and the rays $\mathbf{r}_L$ and $\mathbf{r}_R$ to the left and right cameras, $\mathbf{C}_L$ and $\mathbf{C}_R$, all of which are starting at the centre of the desired camera, $\mathbf{C}_D$. See the diagram in Figure 6 for illustration.

We first project all rays into the plane of the camera trajectory using Equation 1, and obtain the rays $\mathbf{r}^*_D$, $\mathbf{r}^*_L$ and $\mathbf{r}^*_R$. Next, we iterate over all pairs of adjacent cameras, L and R, and compute the signed angles between the left camera and the desired direction, $\alpha_{LD} = \angle(\mathbf{r}^*_L, \mathbf{r}^*_D)$, as well as between the right camera and the desired direction, $\alpha_{RD} = \angle(\mathbf{r}^*_R, \mathbf{r}^*_D)$. We have found the optimal camera pair if rays $\mathbf{r}^*_L$ and $\mathbf{r}^*_R$ (1) lie on either side of $\mathbf{r}^*_D$, i.e. $\alpha_{LD} \cdot \alpha_{RD} \leqslant 0$, and (2) are in the hemisphere centred on $\mathbf{r}^*_D$, i.e. $|\alpha_{LD}|, |\alpha_{RD}| < \frac{\pi}{2}$. We then use the identified cameras to synthesise the colour of the pixel $\mathbf{x}_D$ as described in Section 6.1.

To compute the blending weight α, we consider the ratio of angles between $\mathbf{r}^*_D$, $\mathbf{r}^*_L$ and $\mathbf{r}^*_R$. Specifically, we compute α using

$$\alpha = \frac{\alpha_{LD}}{\alpha_{LR}} = \frac{\angle(\mathbf{r}^*_L, \mathbf{r}^*_D)}{\angle(\mathbf{r}^*_L, \mathbf{r}^*_R)}. \tag{11}$$

It is worth noting that if a desired ray $\mathbf{r}^*_D$ passes through an input camera, it is collinear with either $\mathbf{r}^*_L$ or $\mathbf{r}^*_R$, resulting in a blending weight of $\alpha \in \{0,1\}$. The ray's colour will thus be entirely determined by the camera through which it passes. This satisfies the epipole consistency requirement articulated by Buehler et al. [4].

6.3 Sampling and reconstructing the plenoptic function

Every pixel of every input image is a sample of the plenoptic function [1]. The synthesis of a pixel in a novel view then corresponds to reconstructing the plenoptic function for a given position $\mathbf{C}_D$ and direction $\mathbf{r}_D$. In our approach, we combine the closest two plenoptic samples ($\mathbf{x}_L$ and $\mathbf{x}_R$), associated with the two closest cameras (see Figure 6). For camera ray reconstruction, we use view-dependent flow-based blending to identify suitable plenoptic samples and blend them linearly (Section 6.1) to synthesise the pixel in the output view.

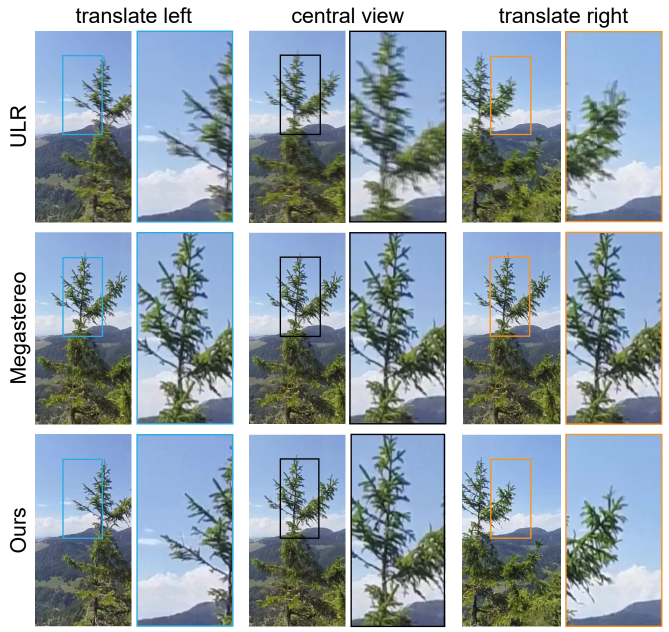


Fig. 7. Results on the Jaman dataset. We translate the novel view and compare our results with ULR [4] and Megastereo [26]. Our results show high visual fidelity *and* plausible motion parallax (see crops).

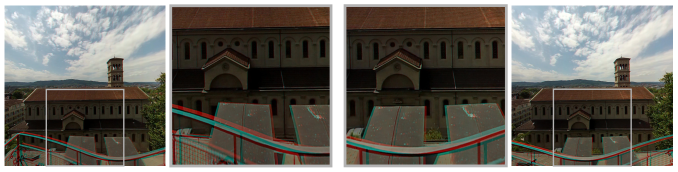


Fig. 8. Red-cyan anaglyph stereo views generated by our approach for two views with significant motion parallax. Dataset by Richardt et al. [26].

7 Results

In Figures 7 and 10 and our supplemental video, we show novel views synthesised by our approach for multiple panoramic datasets and compare them to Megastereo [26], a state-of-the-art omnidirectional stereo technique that is most closely related to our desired capturing setup. We also compare to unstructured lumigraph rendering (ULR) [4] as a baseline proxy-based image-based rendering technique. While ULR (using the four nearest cameras) generates views with motion parallax, they suffer from severe ghosting due to texture misalignment. Megastereo generates sharp views but without any motion parallax. Our results simultaneously exhibit high visual fidelity and motion parallax.

This is best seen in the supplemental video, where we show circular camera paths and a forward–backward translation to demonstrate that our approach produces plausible motion parallax in this case as well. We can easily render stereoscopic images using our approach, which we show using red-cyan anaglyph stereo images in Figure 8. Note that our approach delivers both binocular disparity and motion parallax.

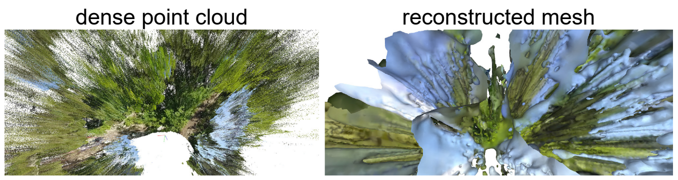


Fig. 9. The dense reconstruction result obtained from our Jaman dataset using COLMAP [28] is unsuitable for view synthesis because of its incompleteness. Natural outdoor datasets remain challenging to reconstruct.

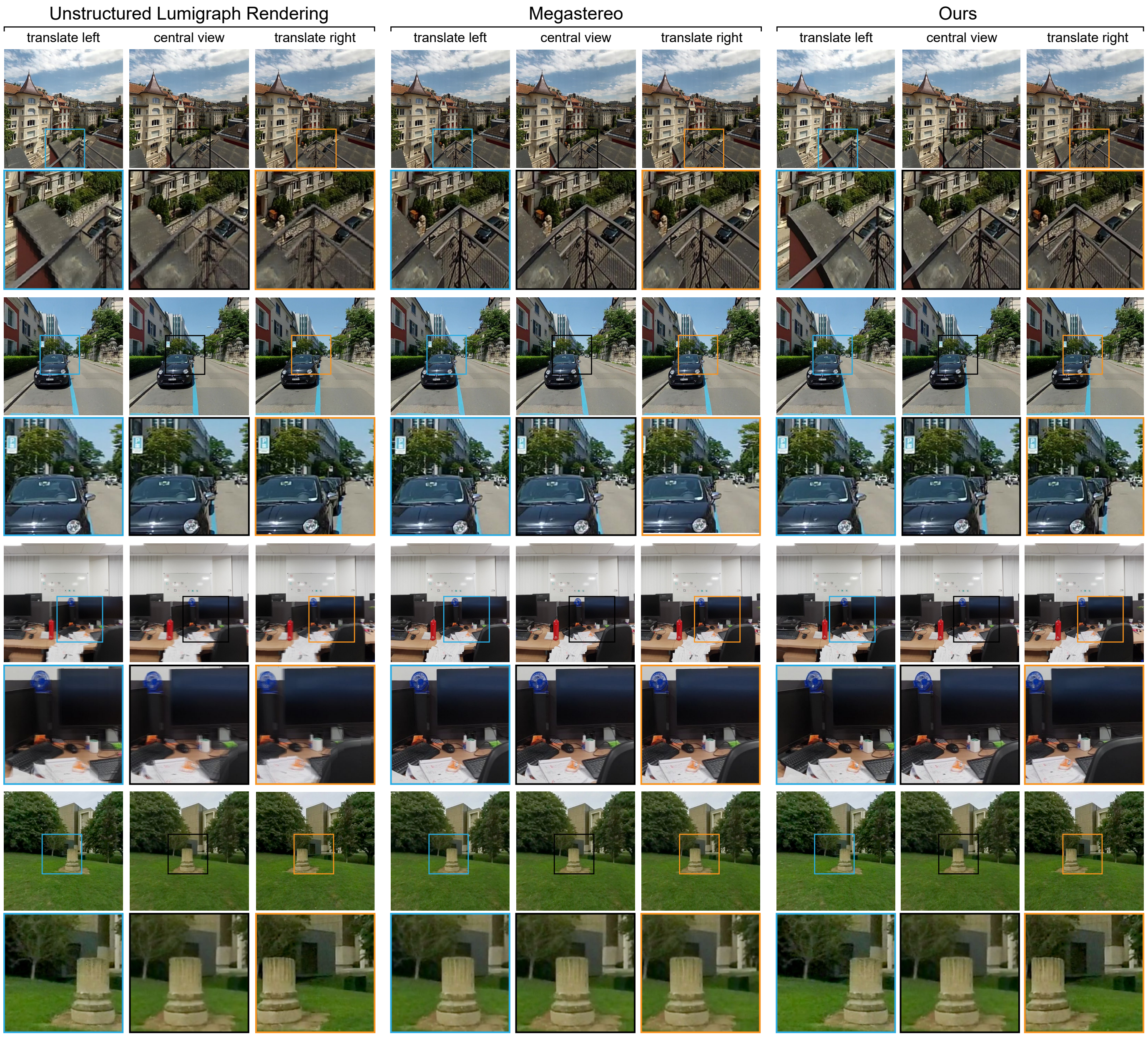


Fig. 10. Comparison of synthesised views between ULR [4], Megastereo [26] and our approach. Datasets (from top to bottom): Rooftop, Street, Office and Campus. Our results have high visual fidelity and produce motion parallax (see crops), which reveals occluded objects such as cars occluded by the roof (top row) or the cars parked behind the black car (second row). We show results for the BBQ dataset in the supplemental video.

As demonstrated by Hedman et al. [13], even state-of-the-art 3D reconstruction techniques still produce patchy reconstructions with many holes, e.g. for the sky, which make them unsuitable for novel-view synthesis. We show such an example in Figure 9. The related approaches of Hedman et al. [13] and Luo et al. [18] are not designed to work on our input data, as they assume a different capture strategy: sparse fisheye views and capturing 4,032 images with a robotic arm on a sphere, respectively. Conversely, their sparse input views do not sample viewing directions sufficiently densely for our approach to work well. However, in our supplemental video, we compare the core view synthesis approach of Luo et al. [18] to ours. The former uses the desired camera's viewing direction instead of the per-pixel ray direction $\mathbf{r}_D$ in our case. This results in a synthesised image *on* the camera circle as a mixture of its two neighbouring cameras.

Datasets The datasets we show were captured with a range of different cameras and capture strategies, see Table 1 for a complete list, including their main parameters (illustrated in Figure 11). Rooftop, Street and BBQ were captured with a single circular sweep using a GoPro 2. Office was captured with a single circular sweep using a Samsung Galaxy S9+. Campus and Jaman were captured with a single circular sweep using an Insta360 One 360° camera. From the stitched equirectangular video, we extracted perspective views with a field of view of 120°×120° for use with our approach. The wider field-of-view makes the reconstruction of camera geometry more reliable in practice.

Table 1. Datasets shown in our paper.

Dataset	Capture	Images (N)	Radius r	β	γ
Jaman	manual	275	0.80 m	1.3°	120.0°
Rooftop [26]	rig	360	1.22 m	1.0°	87.66°
Street [26]	manual	180	0.80 m	2.0°	87.66°
Office	manual	200	0.80 m	1.8°	39.99°
Lake	rig	360	0.84 m	1.0°	120.0°
BBQ	manual	144	0.80 m	2.5°	87.66°

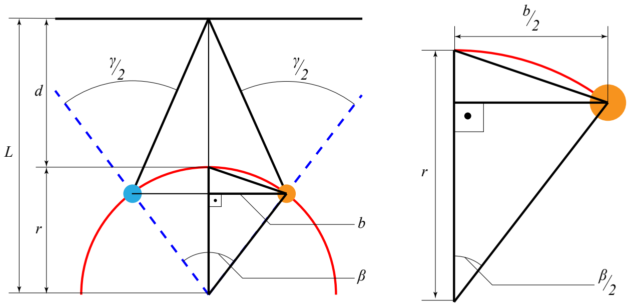


Fig. 11. Idealised dataset geometry. The angular baseline β represents the spacing of views on the camera circle (in red) with radius r. The angle γ denotes the horizontal field-of-view of the input camera.

Run times Our preprocessing takes about 4 hours for 400 images with a resolution of 1440×1920 pixels on an Intel i7-7700K quad-core CPU with 32 GB RAM and an NVIDIA GTX 1080 GPU. The computational bottleneck of the preprocessing is structure-from-motion. Peak rendering time for a one-megapixel image is 5 ms, resulting in a frame rate of 200 fps, which is suitable for VR head-mounted displays.

8 Viewing Area Analysis

Based on the assumption of an ideal circular camera configuration (illustrated in Figure 11) with a radius r, angular baseline β and (horizontal) field of view of γ, we now derive an analytical upper bound for the the supported viewing area size and verify it experimentally. For a derivation of the minimum visible depth, we refer the reader to Schroers et al.'s Appendix C [29].

The range of supported views is constrained by the reconstructed camera geometry to an approximately circular viewing area that is concentric with the camera circle. This generalizes the viewing circle commonly used in omnidirectional stereo approaches [2, 16, 23, 26, 29] or the viewing sphere used by Luo et al. [18].

For a fixed viewpoint, such as $\mathbf{C}_0$ in Figure 12, the quality of the synthesised ray depends on the degree of overlap between the field-of-views of the enclosing camera pair (light blue and orange). If both cameras overlap, the ray's colour is synthesised as a mixture of the two cameras (green error band at top of figure). If they do not overlap, the colour mixture fades to black, as one camera's view is out-of-bounds and its contribution is thus zero or black (yellow error band).

Specifically, let's consider a viewpoint such as $\mathbf{C}_1$ or $\mathbf{C}_2$ in Figure 12, which is translated horizontally away from $\mathbf{C}_0$ by a distance x while keeping the viewing direction the same. To compute an upper bound on the maximum viewing area, we assume that the cameras are sampled densely on the camera circle, i.e. $\beta \to 0°$. Under this assumption, the viewing rays from $\mathbf{C}_i$ intersect the camera circle exactly at a camera, indicated by a dark green circle. For view synthesis to succeed, the viewing ray needs to fall within the camera's field of view. In other words, the angle δ_i between the viewing ray and the dark green camera's

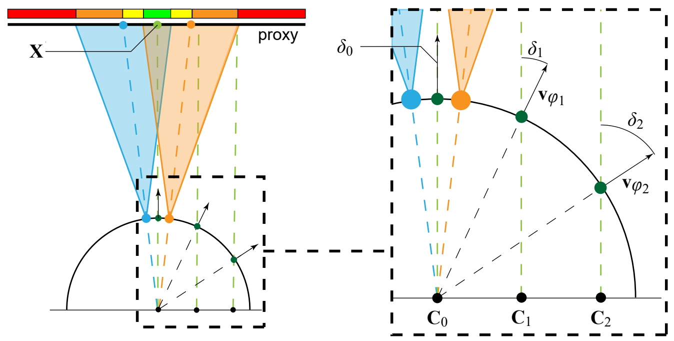


Fig. 12. Viewing area analysis. **Left:** The desired camera is at the centre of the camera circle. The error bar indicates blending performance depending on $\mathbf{X}$ and the desired view $(\mathbf{C}_D, \mathbf{r}_D)$. **Right:** The desired camera is translated from $\mathbf{C}_0$ to $\mathbf{C}_1$ and $\mathbf{C}_2$, while keeping the viewing direction $\mathbf{r}_D$ fixed. The angle δ_i (for $i \in \{0,1,2\}$) restricts the viewing area.

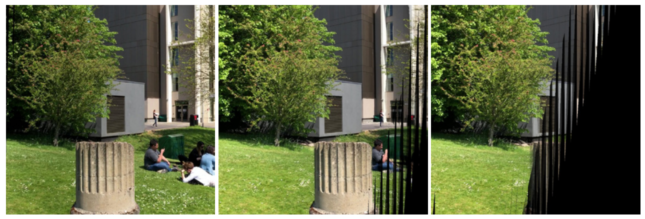


Fig. 13. Limitation: our two-view novel-view synthesis fails if the synthesised rays do not reproject into the input camera views.

viewing direction $\mathbf{v}_{\varphi_i}$ needs to be less than half the field of view γ, i.e. $\delta < \frac{\gamma}{2}$. We can express the angle δ using $\delta = \arcsin \frac{x}{r}$ and then solve for x to obtain

$$x < r \cdot \sin \frac{\gamma}{2}. \tag{12}$$

Equation 12 provides an upper bound on the radius of the maximum viewing area, depending only on the radius r of the camera circle and the camera's field-of-view γ. Leaving the viewing area produces artefacts as in Figure 13.

Experimental validation In practice, imprecise capture and narrow-field-of-view consumer cameras often reduce the available viewing area compared to the theoretical bound. We measure the actual size of the viewing area for different datasets in Figure 14. The datasets vary in terms of their radius $r \in [0.8, 1.22]$ and field of view $\gamma \in [40, 120]$, but also by scene type, e.g. indoors (Office) vs outdoors (Lake).

The left-most set of bars in Figure 14 show the theoretical upper bound of the viewing area according to Equation 12, and the following sets of bars show the observed size for different angular baselines $\beta \in [1, 24]$. The desired camera is translated until the view synthesis breaks (see Figure 13). We do this in both directions and average both distances to estimate the radius of the actual viewing area. Our method relies on $\beta \leq 2°$ for best results while the viewing area shrinks quickly for $\beta > 6°$. Rooftop and Lake come close to the theoretical limit, since the rig capture yields nearly ideal camera paths. Hand-held datasets, such as Jaman, Street and Office, show much smaller viewing areas due to less precise capturing. Five out of our six datasets support a viewing area with a diameter of 50 cm or more, and up to 1.5 m for the Rooftop, which would be sufficient for exploration in virtual reality while seated.

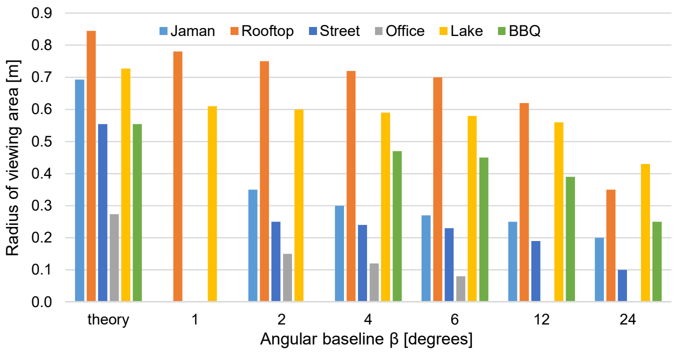


Fig. 14. Viewing area comparison: theory versus practice for different angular baselines β (i.e. subsampled datasets). Note that datasets are only tested for β larger than in Table 1, i.e. if there are enough views.

9 Discussion and Conclusion

The sampling density of views needed to create high-quality immersive experiences is scene-dependent and needs to be sufficiently high for visually optimal results. Better scene understanding would help to identify critical areas and could lead to more reliable reconstructions.

The data we precompute in Section 5 could be efficiently encoded into the input video. The optical flow fields, for example, are high-quality versions of standard motion estimation as employed in video codecs. Camera calibration data and trajectory information could be stored in the metadata. This would result in an augmented video file that is only a small fraction larger than a standard video.

Limitations Our two-view synthesis approach relies on the pair of cameras actually seeing the proxy point $\mathbf{X}$ determined by the desired ray. This assumption may be violated if input views are slanted away from the radially-outwards direction (see increasing δ in Figure 12 and example in Figure 13). This issue can be ameliorated using wide-angle optics to increase the input camera's field of view. In addition, as our approach is purely image-based, the captured scene cannot be edited, but is reproduced as is. The viewing area is currently limited to a roughly circular region in the 2D plane (see Section 8), which prevents out-of-plane motions. Our approach treats the desired camera like a perspective camera during synthesis, but as each pixel is potentially synthesised from a different pair of input views, the synthesised image combines the perspectives of multiple views. This may result in distorted scene objects in novel viewpoints (see handrail in Figure 8). Vertical distortion is known from state-of-the-art omnidirectional techniques [2, 29] and is most noticeable for nearby scene objects, and when synthesising wide field-of-views. This naturally leads to the question of what influence vertical distortion has on the user's viewing comfort, which requires further study in a perceptual experiment.

Conclusion We presented a new solution for generating and displaying high-quality 360° panoramas with motion parallax in real time from just a single input video. Our method produces convincing, high-quality results despite not using any explicitly reconstructed proxy geometry. However, our approach would benefit from the availability of more accurate proxy geometry, which would help reduce vertical distortion and increase the viewing area to support 6-degrees-of-freedom head motions. A computational bottleneck of our approach is the reconstruction of extrinsic camera geometry, which appears to be particularly difficult for inside-out captures like ours.

Acknowledgments

This work was supported by EU Horizon 2020 MSCA grant FIRE (665992), the EPSRC Centre for Doctoral Training in Digital Entertainment (EP/L016540/1), RCUK grant CAMERA (EP/M023281/1), a UKRI Innovation Fellowship (EP/S001050/1), a Rabin Ezra Scholarship and an NVIDIA Corporation GPU Grant.

References

[1] E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing*, pages 3–20. MIT Press, 1991.

[2] R. Anderson, D. Gallup, J. T. Barron, J. Kontkanen, N. Snavely, C. Hernandez, S. Agarwal, and S. M. Seitz. Jump: Virtual reality video. *ACM Trans. Graph.*, 35(6):198, 2016. DOI.

[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004. DOI.

[4] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph rendering. In *SIGGRAPH*, 2001. DOI.

[5] G. Chaurasia, S. Duchêne, O. Sorkine-Hornung, and G. Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. Graph.*, 32(3):30, 2013. DOI.

[6] S. E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH*, 1993. DOI.

[7] A. Davis, M. Levoy, and F. Durand. Unstructured light fields. *Comput. Graph. Forum*, 31(2):305–314, 2012. DOI.

[8] P. Debevec, G. Downing, M. Bolas, H.-Y. Peng, and J. Urbach. Spherical light field environment capture for virtual reality using a motorized pan/tilt head and offset camera. In *SIGGRAPH Posters*, 2015. DOI.

[9] M. Eisemann, B. D. Decker, M. Magnor, P. Bekaert, E. de Aguiar, N. Ahmed, C. Theobalt, and A. Sellent. Floating textures. *Comput. Graph. Forum*, 27(2):409–418, 2008. DOI.

[10] Y. Furukawa and C. Hernández. Multi-view stereo: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 9 (1–2):1–148, 2015. DOI.

[11] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *SIGGRAPH*, 1996. DOI.

[12] P. Hedman, T. Ritschel, G. Drettakis, and G. Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph.*, 35(6): 231, 2016. DOI.

[13] P. Hedman, S. Alsisan, R. Szeliski, and J. Kopf. Casual 3D photography. *ACM Trans. Graph.*, 36(6):234, 2017. DOI.

[14] I. P. Howard and B. J. Rogers. *Seeing in Depth*. Oxford University Press, 2008. DOI.

[15] J. Huang, Z. Chen, D. Ceylan, and H. Jin. 6-DOF VR videos with a single 360-camera. In *IEEE VR*, 2017. DOI.

[16] H. Ishiguro, M. Yamamoto, and S. Tsuji. Omni-directional stereo for making global map. In *ICCV*, 1990. DOI.

[17] M. Levoy and P. Hanrahan. Light field rendering. In *SIGGRAPH*, 1996. DOI.

[18] B. Luo, F. Xu, C. Richardt, and J.-H. Yong. Parallax360: Stereoscopic 360° scene representation for head-motion parallax. *IEEE Trans. Vis. Comput. Graph.*, 24(4):1545–1553, 2018. DOI.

[19] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski. Low-cost 360 stereo photography and video capture. *ACM Trans. Graph.*, 36(4):148, 2017. DOI.

[20] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH*, 1995. DOI.

[21] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. DOI.

[22] R. S. Overbeck, D. Erickson, D. Evangelakos, M. Pharr, and P. Debevec. A system for acquiring, compressing, and rendering panoramic light field stills for virtual reality. *ACM Trans. Graph.*, 37(6):197, 2018. DOI.

[23] S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic stereo imaging. *IEEE TPAMI*, 23(3):279–290, 2001. DOI.

[24] E. Penner and L. Zhang. Soft 3D reconstruction for view synthesis. *ACM Trans. Graph.*, 36(6):235, 2017. DOI.

[25] F. Perazzi, A. Sorkine-Hornung, H. Zimmer, P. Kaufmann, O. Wang, S. Watson, and M. Gross. Panoramic video from unstructured camera arrays. *Comput. Graph. Forum*, 34(2):57–68, 2015. DOI.

[26] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung. Megastereo: Constructing high-resolution stereo panoramas. In *CVPR*, 2013. DOI.

[27] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016. DOI.

[28] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. DOI.

[29] C. Schroers, J.-C. Bazin, and A. Sorkine-Hornung. An omnistereoscopic video pipeline for capture and display of real-world VR. *ACM Trans. Graph.*, 37(3):37, 2018. DOI.

[30] S. M. Seitz and C. R. Dyer. View morphing. In *SIGGRAPH*, 1996. DOI.

[31] H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. In *SIGGRAPH*, 1999. DOI.

[32] H.-Y. Shum, S.-C. Chan, and S. B. Kang. *Image-Based Rendering*. Springer, 2007. DOI.

[33] M. Slater, M. Usoh, and A. Steed. Depth of presence in virtual environments. *Presence: Teleoperators and Virtual Environments*, 3(2):130–144, 1994. DOI.

[34] R. Szeliski. Image alignment and stitching: a tutorial. *Found. Trends Comput. Graph. Vis.*, 2(1):1–104, 2006. DOI.

[35] J. Thatte, J.-B. Boin, H. Lakshman, and B. Girod. Depth augmented stereo panorama for cinematic virtual reality with head-motion parallax. In *ICME*, 2016. DOI.

[36] D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. H. Salesin. Multiperspective panoramas for cel animation. In *SIGGRAPH*, 1997. DOI.

[37] K. C. Zheng, S. B. Kang, M. F. Cohen, and R. Szeliski. Layered depth panoramas. In *CVPR*, 2007. DOI.

[38] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph.*, 37(4):65, 2018. DOI.