Casual Real-World VR using Light Fields

Yusuke Tomoto
Fyusion Inc.

Srinivas Rao
Fyusion Inc.

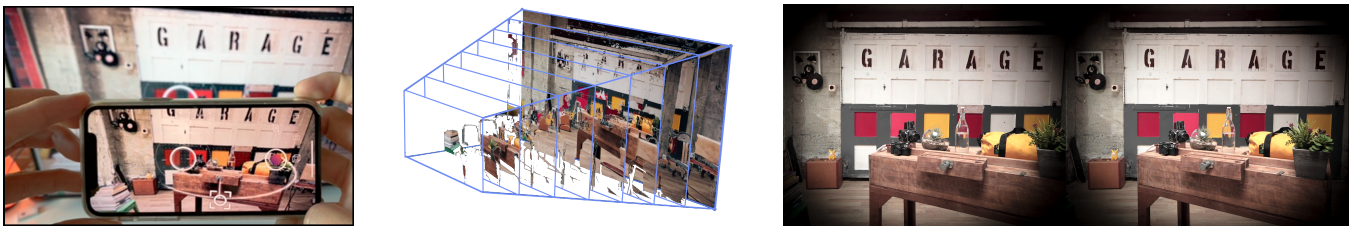
Tobias Bertel
University of Bath

Krunal Chande
Fyusion Inc.

Christian Richardt
University of Bath

Stefan Holzer
Fyusion Inc.

Rodrigo Ortiz-Cayon
Fyusion Inc.



(a) Guided Capture **(b) Multiplane Image (MPI)** **(c) Rendering in VR**

Figure 1: Overview of our system: (a) An AR app guides through the casual capturing process. (b) A neural network promotes a subset of input viewpoints to multiplane images, from which we extract high-quality geometry per view for faster rendering. (c) Our scene representation can be rendered in real-time on desktop and VR. We provide results for several datasets.

ABSTRACT

Virtual reality (VR) would benefit from more end-to-end systems centered around a casual capturing procedure, high-quality visual results, and representations that are viewable on multiple platforms. We present an end-to-end system that is designed for casual creation of real-world VR content, using a smartphone. We use an AR app to casually capture a linear light field of a real-world object by recording a video sweep around the object. We predict multiplane images for a subset of input viewpoints, from which we extract high-quality textured geometry that are used for real-time image-based rendering suitable for VR. The round-trip time of our system, from guided capture to interactive display, is typically 1–2 minutes per scene. See the submission video for a walkthrough and results.

CCS CONCEPTS

• **Computing methodologies** → **Rendering**; **Computational photography**; **3D imaging**; **Neural networks**.

KEYWORDS

Virtual reality, VR photography, view synthesis, multiplane images

ACM Reference Format:
Yusuke Tomoto, Srinivas Rao, Tobias Bertel, Krunal Chande, Christian Richardt, Stefan Holzer, and Rodrigo Ortiz-Cayon. 2020. Casual Real-World VR using Light Fields. In *SIGGRAPH Asia 2020 (SA '20 Posters), December 04-13, 2020.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3415264.3425452

SA '20 Posters, December 04-13, 2020, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8113-0/20/11.
https://doi.org/10.1145/3415264.3425452

1 INTRODUCTION

Providing real-world VR experiences requires capture, reconstruction and representation of real scenes such that novel viewpoints can be rendered in real-time. State-of-the-art visual results for methods assuming *casual* capture stages are achieved by either using explicit geometry, e.g. per-view depth maps obtained from a smartphone's dual camera [Hedman and Kopf 2018], or learned representations trained solely on posed images [Mildenhall et al. 2019].

Light fields emerged as a *plenoptic* approach for rendering 3D scenes without the need of detailed modeling [Levoy and Hanrahan 1996]. Light fields directly encode the visual appearance of a scene as seen from a densely sampled set of input views. Davis et al. [2012] developed the first *casual* end-to-end approach based on light fields that does not require any special hardware setup, such as a gantry or a camera rig. The method relies on *hundreds* of images to produce high-quality results. Hedman and Kopf [2018] produce high-quality results, but using the dual camera has two important limiting consequences: the capturing time increases, and processing the depth maps into a multilayer mesh representation is sophisticated and time-consuming. Mildenhall et al. [2019] use multiplane images (MPIs; Zhou et al. [2018]) to create a light field representation consisting of several *local* light fields associated to the input viewpoints, which are blended together *back-to-front* to synthesize novel viewpoints. The size of an MPI and its rendering time both limit the representation from being used in VR.

We present a fast and casual end-to-end system for creating real-world VR experiences while focusing on *user-friendly* capturing paths, specifically 1D video sweeps. We predict MPIs for a subset of input viewpoints from which we extract high-quality depth maps that are used for a high-quality image-based rendering algorithm suitable for VR. We demonstrate our interactive capture process and show results of captured objects and scenes in the supplemental video.

2 SYSTEM OVERVIEW

Our system consists of three stages: capture, processing and rendering. For capturing, we use a custom app on an iPhone 11, which guides the user with augmented reality markers. The processing is performed on a remote server with an NVIDIA TITAN RTX GPU, which consumes the captured images and produces a scene representation. The rendering runs on a VR-ready laptop with an Intel i7 CPU, 16 GB RAM and an NVIDIA GeForce RTX 2070 GPU. We built two versions of the final viewer: an OpenGL-based viewer for directly visualizing results on a desktop or laptop, and a Unity-based viewer for VR rendering using an Oculus Rift S headset.

Scene Representation. We represent the captured scene as a set of multiplane images (MPIs). An MPI consists of D parallel planes with RGBα textures [Zhou et al. 2018], as illustrated in Figure 1b. MPIs can be considered as light fields with local partial geometry. This representation has recently become popular in the computer vision and graphics communities, as MPIs are particularly well-suited for a deep-learning pipelines [Flynn et al. 2019; Mildenhall et al. 2019; Srinivasan et al. 2019]. In practice, we use $D = 64$ layers.

2.1 Capture

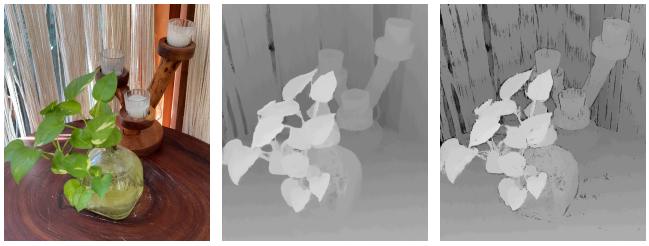
We developed a custom iPhone app that guides the user along a ring-like trajectory around an object of interest using augmented reality markers. To keep our pipeline tractable in terms of bandwidth and processing time, we sparsely sample input frames from a video following the sampling theory presented by Mildenhall et al. [2019], who proposed a bound for sampling views of a scene while still reliably reconstructing the desired MPIs. We propose to capture 1D video sweeps (ring-like trajectory) instead of a 2D grid of viewpoints to keep the process itself as user-friendly and casual as possible.

2.2 Processing

Our processing consists of three main steps that convert the sampled input views to RGBD images for efficient rendering: (1) we compute camera poses using our own custom structure-from-motion technique, (2) we predict an MPI for each sampled image using a 3D CNN, (3) we compute depth maps from the predicted MPIs, which are used in our rendering stage. For our SfM system, we make assumptions about the capture trajectory which are used for initialization and as priors to the bundle-adjustment problem. We set the number of keypoints, iterations and other optimization parameters needed to achieve acceptable results while keeping the system as fast as possible. For predicting MPIs, we use Mildenhall et al.'s pretrained network, which was mainly trained on synthetic data, and fine-tune it on 100 real training examples to increase the quality of MPIs for real scenes (see Figure 2). We compute depth maps from MPIs using a weighted summation of the depth of each layer, weighted by the MPI's per-pixel α values.

2.3 Rendering

MPIs render views by re-projecting its layers into the novel view and compositing them from back-to-front. If we want to render views by blending contributions of $N = 8$ neighboring views (4 for each eye), one single stereo-frame would require at least $N \times D = 512$ render passes, which severely limits the rendering performance. Instead, we use the depth maps extracted from the MPIs to create a per-view geometry and render individual contributions from the neighboring input views. For a given posed input view, we create a dense triangulated mesh assigning a vertex to the center of each pixel and back-projecting its depth to world space. For memory efficiency, the geometry of each view is created at rendering time. We reuse a vertex buffer object with vertices on a 2D grid while depth is uploaded to a vertex buffer on demand. This rendering approach produces rendering artifacts because some triangles will be excessively stretched at depth discontinuities. To reduce this effect during blending, we penalize the contribution of those regions with the inverse of the area stretching factor (the amount that a pixel gets stretched under re-projection). We compute the pixel area stretching factor as the determinant of the Jacobian matrix of texture coordinate. We also weight contributions by proximity to the novel view, penalizing contributions from far-away input views. We show a variety of results in our supplemental video.



(a) RGB frame **(b) Mildenhall et al.** **(c) Fine-tuned model**

Figure 2: Fine-tuning Mildenhall et al.'s MPI prediction network (b) increases the accuracy of our depth maps (c).

3 CONCLUSION

We demonstrated a fast and reliable end-to-end system for casually creating and displaying real-world VR experiences. We based our work on two tightly connected principles: *guided capturing* of *local light fields*. The guided capturing determines the efficiency of our system and the representation gives us theoretical guarantees that the final representation will perform sufficiently well, i.e. providing sufficiently good MPIs for the given datasets for further generation of per-view depth maps. The main limiting factors are all inherited from rendering with explicit geometry, e.g. we see stretched triangles from the mesh reconstruction if we move too far from the input viewpoints. Note that our pipeline can be adjusted to the requirements of different target platforms, e.g. by decreasing image or depth map resolution to speed up transmission and rendering, thus making interaction with our system more appealing.

REFERENCES

Abe Davis, Marc Levoy, and Frédo Durand. 2012. Unstructured Light Fields. *Computer Graphics Forum* 31, 2 (May 2012), 305–314.

John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. 2019. DeepView: View synthesis with learned gradient descent. In *CVPR*.

Peter Hedman and Johannes Kopf. 2018. Instant 3D Photography. *ACM Trans. Graph.* 37, 4 (July 2018), 101:1–12.

Marc Levoy and Pat Hanrahan. 1996. Light Field Rendering. In *SIGGRAPH*.

Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. 2019. Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines. *ACM Trans. Graph.* 38, 4, Article 29 (July 2019), 14 pages.

Pratul P. Srinivasan, Richard Tucker, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. 2019. Pushing the Boundaries of View Extrapolation with Multiplane Images. In *CVPR*.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. 2018. Stereo Magnification: Learning View Synthesis using Multiplane Images. *ACM Trans. Graph.* 37, 4 (Aug. 2018), 65:1–12.