Demo abstract: Live 3D Scene Capture for Virtual Teleportation

Tao Jin*, Mallesham Dasari*, Connor Smith$, Kittipat Apicharttrisorn*
Anthony Rowe*, Srinivasan Seshan*
Carnegie Mellon University*, Magic Leap$
{taojin, malleshd, kapichar, agr}@andrew.cmu.edu, srini@cs.cmu.edu, consmith@magicleap.com

ABSTRACT

It has long been a goal of immersive telepresence to capture and stream 3D spaces such that a remote viewer can watch from any location or angle within the scene. This demonstration presents `Mosaic`, a new distributed 3D scene capture system that uses *textured mesh data representation* for streaming a 3D volumetric video of a space to remote viewers. Compared to more common point cloud based methods, we show that textured mesh data requires less bandwidth and yields the same visual quality. However, textured mesh reconstruction is compute and memory intensive, mesh simplification is not easily parallelizable, and texture maps lacks spatial and temporal coherence. `Mosaic` tackles these challenges by examining each computational stage and determines how they can be efficiently distributed across multiple compute nodes to reduce overall latency, minimize bandwidth, and maintain quality. We then provide an end-to-end latency and bandwidth breakdown that can be used to target future acceleration work.

CCS CONCEPTS

• **Computing methodologies** → **Mixed / augmented reality**; **Virtual reality**; **Mesh geometry models**; **Volumetric models**.

ACM Reference Format:
Tao Jin*, Mallesham Dasari*, Connor Smith$, Kittipat Apicharttrisorn*, Anthony Rowe*, Srinivasan Seshan*, Carnegie Mellon University*, Magic Leap$, {taojin, malleshd, kapichar, agr}@andrew.cmu.edu, srini@cs.cmu.edu, consmith@magicleap.com. 2022. Demo abstract: Live 3D Scene Capture for Virtual Teleportation. In *The 20th ACM Conference on Embedded Networked Sensor Systems (SenSys '22), November 6–9, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3560905.3568086

1 INTRODUCTION

One of the core challenges of any interactive and immersive telepresence application is the ability to capture and stream 3D scenes in real-time. This requires combining high-rate sensor data from multiple depth sensors in real-time. Holoportation [3], Volograms [5], and Project Starline [2] are just a few examples of emerging 3D telepresence systems that are poised to deliver significant value for several verticals in Industry 4.0 ranging from logistics and supply chain to hospitality and healthcare. In the future, we envision a volumetric telepresence platform composed of a mixture of fixed sensors (to capture a working area) along with mobile sensors (on phones and headsets) to enhance/fill in regions of interest. In practice, this system should operate on standard networks (wired or wireless with background traffic) and across the Internet.

Existing 3D scene capture systems fall into two categories: ones that output RGB-D [4] or point cloud [1] representations and ones that output textured meshes [6]. We can draw an analogy to 2D graphics where mesh geometry is more akin to a vectorized format and point cloud data is more like a rasterized representation. The benefits of RGB-D and point cloud designs include the fact that most sensors natively output such data representation and combining sensor feeds into a larger scene is a simple, additive process which reduces the computational burden on the capture side. However, clients that consume this data are given the relatively complex task of converting large data streams into a rendered geometries. In contrast, meshes are easier to render at a client but require sensors (or capture site) to perform necessary geometry creation.

A significant trade-off between these approaches is that RGB-D and point cloud data are relatively inefficient and use significant bandwidth to capture real-world scenes in high-quality. This is due to two reasons: (1) mesh representations naturally capture and compactly represent the often planar features found in indoor spaces at much lower bitrates, and (2) mesh geometry makes it easy to decouple geometry and texture resolution for efficient 3D representation. For these reasons, `Mosaic` advocates for adopting textured meshes as an intermediate data structure for streaming.

However, streaming textured meshes faces significant challenges. Existing systems [3] rely on centralized 3D scene reconstruction and, as a result, require extremely high data rates (e.g., Holoportation requires 1-2 Gbps bandwidth for each scene [3]) or take significant time (seconds or minutes) to compress using standard 3D compression techniques. Existing designs also scale poorly with the size of a scene due to the high compute and GPU memory demands of doing full scene reconstruction on a centralized node.

2 SYSTEM OVERVIEW

We design `Mosaic` to leverage compute nodes deployed close to cameras to perform partial scene reconstruction. A merging server (deployed within the capture environment) then pulls together the partial reconstructions and merges them into a single scene description. The resulting system eliminates the compute and memory bottlenecks present in the centralized design since operations are performed on smaller components of the overall scene. Fig. 1 shows a detailed system block diagram, Fig. 2 shows system latency breakdown, and Fig. 3 shows a conference room test setup.

2.1 Distributed 3D Scene Capture

In order to alleviate the GPU memory and latency bottlenecks, `Mosaic` splits the capture pipeline across many compute nodes

SenSys '22, November 6–9, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9886-2/22/11…$15.00
https://doi.org/10.1145/3560905.3568086

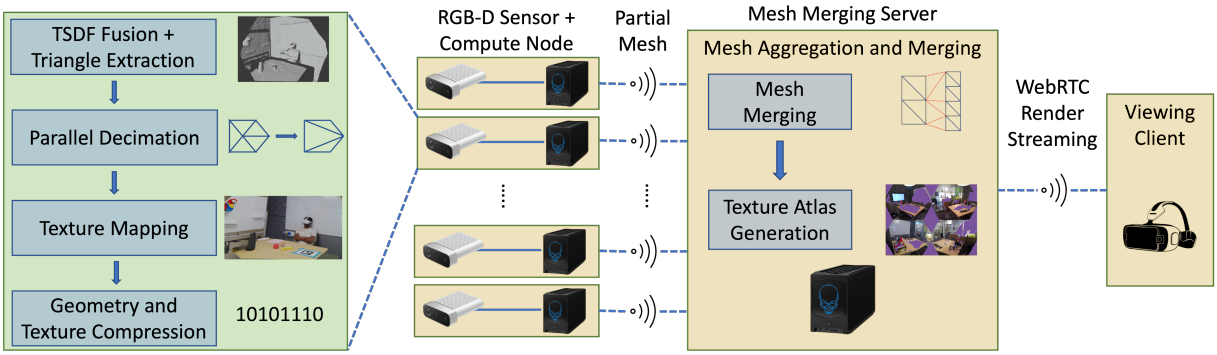

Figure 1: `Mosaic`'s end-to-end scene capture pipeline.

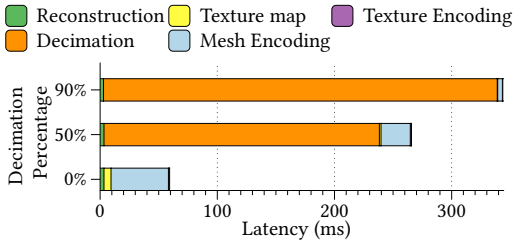

Figure 2: Breakdown of scene capture latency

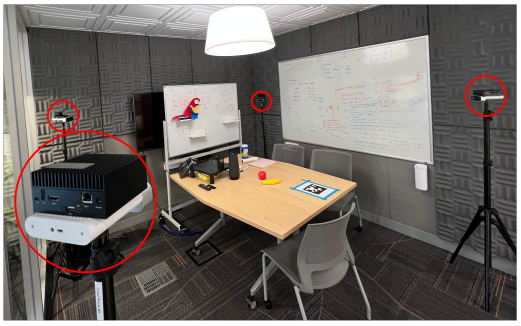

Figure 3: `Mosaic`'s test setup. 4 Azure Kinect cameras with compute nodes are circled in red.

where each node performs a per-camera scene reconstruction (i.e., TSDF volume integration) as well as mesh decimation. The system merges these per-camera streams into a single 3D model on separate compute nodes. A compute node and its GPU has sufficient resources to support a single camera scene and the node can reconstruct its scene and decimate with minimal latency.

2.2 Mesh Merging

Once the per-camera reconstructions are available at the merging server, `Mosaic` merges all the partial reconstructions into a single model. It is necessary to remove the cross-camera overlapping regions for bandwidth efficiency. In theory, it is straightforward to merge different mesh models (albeit assuming that they are from the same coordinate system) by simply concatenating the two data structures and remove duplicate triangles. However, in practice, the camera pose as well as the cameras' depth readings are not precise. Hence it is difficult to observe perfect duplicates when merging even if they are exactly the same geometry. We preserve only the visible mesh layers and remove the redundant regions through (1) connecting the the mesh boundaries by nearest neighbor search and merge, (2) removing hidden mesh layers through raycasting.

2.3 Texturing and Atlas Generation

Once the partial mesh reconstructions from each camera view are merged, our next step is to create a texture map and an atlas for the mesh with the RGB frames from multiple overlapping camera textures. The goal of this process is twofold: (1) creating a 3D to 2D texture map, so that a rendering client can use the map to project the texture onto the geometry during rendering. (2) creating a texture atlas by removing the camera views that are either not in depth camera field of view or overlapping with other camera views.

3 DEMONSTRATION

Our demonstration takes the form of four Microsoft Azure Kinect DK cameras with co-located computers (Intel NUC Extreme) distributed around a 3x3 meter space mounted on tripods. A background scene and users in the center are digitized in real-time and transmitted to a 3D viewer through a wireless network link. The real-time captured 3D scene is shown on a computer screen; it can also be viewed in AR/VR headsets on a browser. In addition, we show a breakdown of end-to-end latency and bandwidth on a external monitor.

We target to show the following: (1) Real-time capturing, processing, and streaming of volumetric content in the form of textured mesh representation. (2) Live free-viewport viewing of captured scene with wirelessly connected AR/VR headset and computer. (3) Live statistics of end-to-end system latency and bandwidth.

REFERENCES

[1] Han et.al. Vivo: Visibility-aware mobile volumetric video streaming. In *MobiCom*, pages 1–13, 2020.
[2] Lawrence et.al. Project starline: A high-fidelity telepresence system. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 40(6), 2021.
[3] Orts-Escolano et.al. Holoportation: Virtual 3d teleportation in real-time. In *UIST*, pages 741–754, 2016.
[4] Record3D. Record3D Website. https://record3d.app/. Online. Accessed: Sep 2022.
[5] Vologrames. Vologrames. https://www.vologrames.com/. Online. Accessed: Sep 2022.
[6] Patrick Stotko, Stefan Krumpen, Matthias B Hullin, Michael Weinmann, and Reinhard Klein. Slamcast: Large-scale, real-time 3d reconstruction and streaming for immersive multi-client live telepresence. *IEEE transactions on visualization and computer graphics*, 25(5):2102–2112, 2019.