3D Gaussian Splatting based VR Video Chat

JASON ZHANG, University of Washington

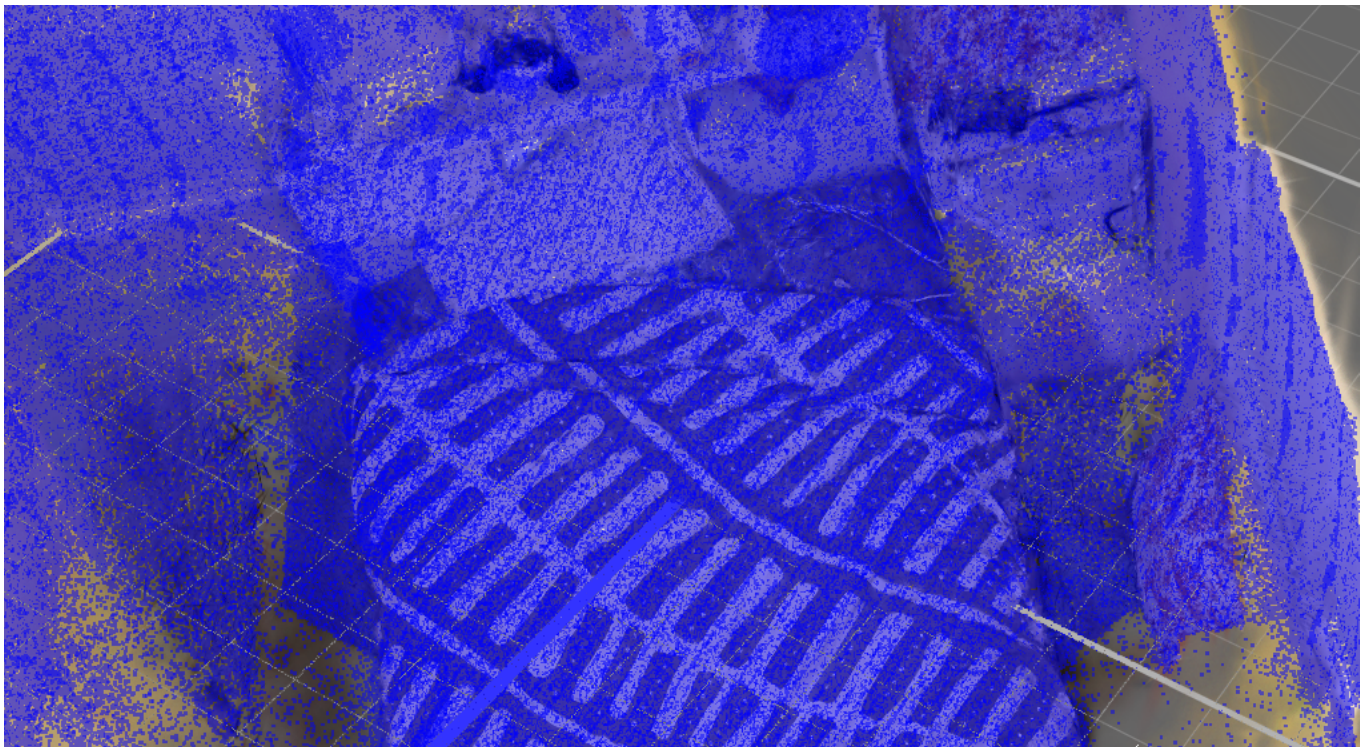

Fig. 1. With 3D Gaussian Splatting, users can generate a 3D scene from just a few images. Through Unity, these scene can be viewed using VR headsets, creating an immersive and interactive video chat experience. This is the point cloud of the scene constructed from 6 images

Conventional video chat applications are only capable of providing a basic visual and audio connection, displaying users on a screen and transmitting voice through device speaking. They lack interactivity and immersive features that replicate the natural dynamics of in person conversations. In an effort to create a more natural video chat experience, companies like Google and Apple have developed their own solution to address these limitations of traditional video calls. Google introduced the Project Starline, while Apple's integrated VR/AR FaceTime features into Apple Vision Pro, both aiming to enhance immersion and interactivity in virtual communication. However, both approaches require specialized hardware like stereo cameras and depth sensors for Project Starline, and the Apple Vision Pro for Apple's solution. With the emergence of 3D Gaussian Splatting, it seems like we are entering an era where 3D scenes can be reconstructed with just a few images taken by everyday cameras. In this project, I aim to integrate 3D Gaussian Splatting generated scenes with VR view models, allowing users to navigate through immersive 3D environments. I also aim to make these scenes dynamically update in real-time, reflecting changes captured by streaming cameras. Compared to prior solutions, this appproach offers a more accessible alternative to hardware depenedent immersive communication methods. I was able to achieve streaming offline (preconstructed) 3D Gaussian Splatting models in VR using Unity. However, real-time generation of consecutive frames remains a significant challenge, as the generation of each scene takes tens of seconds to minutes. Future alternatives could involve optimizing the training workflow of the 3D Gaussian Splatting, or implement a hybrid approach. One potential solution is to generate an initial 3D scene and use Computer Vision based tracking techniques like YOLO to update object coordinates, reconstructing movements without regenerating the full scene. Which a new reconstruction of the scene through 3D Gaussian Splatting would only be needed when detecting the entry of a new object into the scene, which is promising in significantly improving real time performance.

Author's address: Jason Zhang, jasonz04@cs.washington.edu, University of Washington.

1 INTRODUCTION

Modern video chat application typically only requires a device with a screen, camera, microphone, and basic network connectivity. However, while these applications have made remote communication widely accessible, the experience they provide lacks immersion and interactivity, and essentially fails to replicate the natural dynamic of in person conversations. Conventional video calls often present a flat, two dimensional representation of the participants on a screen, lacking depth cues, spatial awareness, and interactivity inherent in face to face interactions. As result, these limitations often lead to reduced engagement, difficulty in perceiving nonverbal cues, and an over all less immersive experience.

To address these shortcomings, companies like Google and Apple have introduced advanced solutions aimed to enhance immersive remote communications. Google's Project Starline utilizes sets of stereo cameras, depth sensors, and light field displays to create a lifelike 3D video chat experience, allowing users to see one another with depth ques without using any headsets. However, this approach relies on specialized hardware and requires users to remain seated in front of the display and camera throughout the conversation. Additionally, users are only able to view their conversation partner from a fixed perspective, limiting their ability to move around and explore the other person's space, which is a crucial aspect of natural, in person conversations. On the other hand, Apple's solution with Apple Vision Pro integrates VR/AR FaceTime, offering similar capabilities while providing a more spatially aware virtual reality environment. While both of these approaches significantly improved immersion, they both rely on specialized hardware, making them expensive and less accessible to average consumers.

With the emergence of 3D Gaussian Splatting, a novel approach to real-time 3D scene reconstruction, new possibilities arise for immersive video communication without requiring high-end hardware. Gaussian Splatting enables 3D scene reconstruction using only a few standard RGB images, eliminating the need for depth sensors or LiDAR. compared to Google's and Apple's hardware dependent solution, this approach has the potential to offer a more accessible alternative by allowing immersive video chats to function with multi-camera setups using conventional devices.

In this project, I explore the interaction of 3D Gaussian Splatting with VR view models, enabling users to navigate immersive 3D environment. I successfully implemented offline streaming of pre-constructed 3D Gaussian Splatting models in Unity, demonstrating the feasibility of viewing and interacting with 3D scenes in VR. However, as real time generation of consecutive frames remain a significant challenge, the real time streaming feature, crucial to video call experience, is currently infeasible.

Despite these limitations, my findings indicate that Gaussian Splatting has the potential to improve immersive communication if future research focuses on optimizing training workflows, reducing reconstruction times, and possibly integrating computer vision techniques for motion tracking to levitate the need of reconstructing the whole scene. By addressing these challenges, Gaussian Splatting could become a viable alternative to hardware heavy solutions like Project Starline and Apple Vision Pro, making high quality, real time 3D video communication more practical and accessible.

1.1 Contributions

Our primary technical contributions are:

- Developed a VR based streaming system for pre-constructed 3D Gaussian Splatting models using Unity, enabling immersive 3D scene exploration within a virtual reality environment.
- Identified and analyzed the challenges of real-time 3D Gaussian Splatting scene generation, highlighting the significant computational bottleneck that makes continuous scene updates infeasible.
- Demonstrated feasibility of using few images from everyday cameras for 3D scene reconstruction, eliminating the need for specialized hardware like stereo cameras or LiDAR
- Proposed a hybrid approach for improving real-time performance, integrating computer vision-based tracking techniques (e.g., YOLO) to update object positions dynamically, reducing the need for full scene reconstruction and paving the way for more efficient real-time updates..

2 RELATED WORK

2.1 Other Products

Feature	**Google Project Starline**	**Apple Vision Pro**	**3D Gaussian Splatting**
Hardware Requirement	Stereo cameras, depth sensors, light-field display	Vision Pro headset	RGB cameras (multi-view setup)
Immersion Level	Moderate	Low	Moderate
Real-Time Interactivity	Fixed viewpoint, limited movement	Fully interactive	Limited (requires faster scene updates)
Consumer Accessibility	Low (high cost, large setup)	Medium (expensive, but portable)	High (if optimized for real-time updates)

Table 1. Comparison of Hardware-Based Immersive Communication Solutions

Several companies have developed hardware-dependent solution to enhance depth perception and immersion in video communication. Google's Project Starline proposed the idea of using stereo cameras and depth sensors to generate a volumetric 3D representation of users and projecting them onto a gigantic light field display to provide a more realistic scenes of depth without using a VR headset. On the other hand, Apple's approach requires the user to purchase the Apple Vision Pro, which contains a few camera and sensors to recreate you facial feature in FaceTime. Finally, Gaussian Splatting requires the least amount of specialized hardware, and only requires a few RGB cameras but they have to be synchronized correctly. Comparing immersion of each product, Google Starline only creates face to face interaction, requiring the user to stay in the same position throughout the conversation and have limited views. Vision Pro's approach doesn't have any 3D constructions but rather just have other's face displayed in the headset. 3D Gaussian Splatting setup allowed users to explore fully constructed 3D scene in Unity with a VR head set, possibly creating the most immersive experience.

2.2 Other Rendering techniques and scene construction

Technique	**NeRFs**	**3D Gaussian Splatting**
Rendering Speed	Slow (seconds per frame)	Fast (real-time rendering)
Scene Construction Time	Slow (minutes per scene)	Moderate (tens of seconds)
Hardware Dependency	Requires GPU acceleration	Requires GPU but no depth sensors
Interactivity	Limited (precomputed views)	Moderate (viewpoint movement)

Table 2. Comparison of Neural Rendering Techniques

Compared to traditional rendering techniques like NeRFs, which often suffer from slow rendering times, taking several seconds per frame, and slow scene construction times, requiring minutes per scene, 3D Gaussian Splatting offers a significant advantage. While scene construction time remains similar, once the scene is generated, Gaussian Splatting enables real-time rendering, allowing users to freely move around the 3D scene without performance bottlenecks. This makes it a more practical choice for applications requiring interactive movements.

3 METHOD AND IMPLEMENTATION DETAILS

3.1 Hardware

All the implementation are done with a PC with NVIDIA RTX 3080 (10GB VRAM), Intel i9-12900KF, 32GB DDR5 4800MHz, and Samsung 990 EVO PRO 1TB running on Windows 10. The headset used was Meta Quest 2. All pictures for scene reconstruction were taken with iPhone 13 Pro at 12MP 4032x3024 resolution.

3.2 Software

The 3D Gaussian Splatting reconstructions in this project were based on the GitHub repository *"InstantSplat: Sparse-view SfM-free Gaussian Splatting in Seconds* by Jonathan Stephens. This codebase was used to process multi-view images and generate 3D Gaussian Splatting representations, allowing for faster scene reconstruction compared to original codebase published with the paper 3D Gaussian Splatting for Real-Time Radiance Field Rendering.

The Unity integration was implemented using the GitHub repository *"Gaussian Splatting Playground in Unity"* by Aras Pranckevičius. This provided the necessary tools to render Gaussian Splatting scenes in Unity, enabling users to move within precomputed 3D environments.

The project was developed using Unity 2022.3, which provided support for VR rendering, along with the Oculus Integration SDK for handling user movement and scene navigation on the Meta Quest 2. This specific Unity version is chosen as the Gaussian Splatting Playground in Unity is originally written in this version. To avoid any compatibility issues, I decided to stay with Unity 2022.3.

3.3 Implementation steps

Below are steps after having all software downloaded and completed setup.

First we have to capture 6 images (specified in InstantSplat) using any camera (desirably using the same camera at the same resolution). Then we run the InstantSplat application, where we specify a input directory (with the 5 images) and a output directory which is where the Splat files would be saved to. We select the corresponding number of views, and desired training iteration (1000 iteration in this implementation). After we start the training process, it should take around 1-2 minutes, and then a mp4 video and a point cloud file would be saved to the specified directory.

After obtaining the point cloud file (.ply), we can run the Gaussian Splatting Playground in Unity, and then create a Gaussian Splat Asset using built in toolkits of this CodeBase using the point cloud file, which then we would just have to drag created asset in to the "Asset variable" of the Gaussian Splat Renderer script and we will have the Gaussian Splatting running in Unity.

For the purpose of this project, I constructed point clouds from consecutive sets of images captured and have turn them into point cloud files. Then I wrote a Unity script for it to alternate in an order of the assets I created as the "game" runs. I also implemented some Unity script control which connects the Oculus Quest 2's joystick to the movements of the main camera, and i also limited it's vertical position to stay always above 0.

This implementation doesn't require blurring out the back ground where even simple flat images are realistically constructed and well incorporated into the scene.

4 EVALUATION OF RESULTS

I was able to accomplish part of the original goal of implementing the system on Meta (Oculus) Quest 2. Specifically, I was able to pre-construct multiple 3D scenes with Gaussian Splatting and have them exported to Unity. Within Unity, I implemented a system that alternates between the scenes (to simulate 3D video playback) while user can move around with joy sticks on the Quest 2 controllers. To ensure a seamless transition between scenes, all scenes were constructed with aligned axes, so when alternating between scenes, we just have to keep track of the current user position, and they will be in relatively the same location visually, maintaining a consistent immersive experience.

My implementation allows users to move around the scene with around 60 Frames Per Second(FPS) while streaming from a PC (with RTX 3080), and around 30 FPS when running on Meta Quest 2. However, several limitations exists in the current implementation. One major limitation is that, due to only using 5 cameras for scene capture, the scene constructed have quite a lot of occluded areas, which caused dark portions and graphical distortions in such area where less image information is available. Another key limitation is due to the high computation cost of reconstructing 3D scenes. Because of this, I was only able to have "offline" reconstruction where scenes have to be pre-constructed, which rather than live video calling, this is more of looking at a 3D scene as a video, which users cannot interact with dynamically updated content.

5 DISCUSSION OF BENEFITS AND LIMITATIONS

3D Gaussian Splatting has demonstrated its ability to produce high-quality 3D reconstructions from images. However, like any technique, it comes with its own set of advantages and challenges. In this discussion, we explore the key benefits, such as its efficiency in generating detailed scenes, as well as its limitations, including computational demands and real-time performance constraints.

5.1 Benefits

5.1.1 High-Fidelity Immersion. The primary benefit of Gaussian Splatting is its ability to generate photorealistic rendering with realistic lighting, texture, and depth cues, making it ideal for VR applications. Unlike traditioanl mesh based rendering, which requires complex geometry processing, Gaussian Splatting sllows for continuous, smooth representation of 3D environments without the need for explicit polygonal structures.

5.1.2 Efficient Rendering. While scene construction remains computationally intensive, once trained, Gaussian Splatting can be rendered in real time. This allows users to freely move within a pre-constructed 3D scene without performance bottlenecks, which makes it an ideal solution for applications requiring interactive features, such as virtual meeting, remote collaborations, or even immersive media experiences.

5.1.3 No Need for specialized hardware. A key breakthrough of Gaussian Splatting is that it eliminates the needs specialized depth sensors or LiDAR to generate 3D scene. Compared to Google's and Apple's approaches, Gaussian Splatting offers a more accessible alternative for immersive video chat, making it more feasible for general consumers. However, in order to stream the footage, it still requires multiple cameras to accurately reconstruct a high quality 3D scene.

5.2 Limitations

5.2.1 Computational Bottleneck. The training process for each new scene is computationally expensive, often requiring tens of seconds to minutes per scene. This limitation is particularly evident when constructing scenes from only six images, which often results in lower image quality. To improve the quality of Guassian Splatting reconstructions, more images need to be included in the training process. However, increasing the number of input images further extends computation time, making real-time scene generation even more challenging.

5.2.2 Limited real-time interactivity. Gaussian Splatting is primarily optimized for static scene representation. Unlike traditional 3D rendering techniques that allow for dymica objects movement and continuous updates, Gaussian Splatting struggles with real-time modifications, making it less suitable for highly interactive environments where objects and users frequently change positions. In my original implementation, representation of movements of objects requires a full reconstruction whenever significant changes occur in the scene. Unlike mesh-based approaches that allow for incremental updates by modifying vertices or textures, Gaussian Splatting lacks the ability to update for only certain parts of the scene without regenerating the whole model.

5.2.3 Dependence on Multi-Camera Input. Unlike conventioanl 2D veideo chat applications, which require only a single camera, Gaussian Splatting needs multiple viewpoints to accurately capture depth, structure, and lighting conditions of a scene. The reliance on multi-camera setup introduces challenges in terms of accessibility and complexity, making it somewhat less piratical for consumers and also hard to implement in real time settings. Gaussian Splatting reconstruct 3D scenes by learning from multiple 2D images, which is only a single camera is used, the system lacks necessary viewpoints, leading to incomplete reconstructions, depth ambiguities, and poor quality in occluded areas. Another challenge with multi-camera setup is that it complicates real time scene acquisition. Capturing live 3D scene requires synchronized multi camera setup, where each camera must be calibrated and positioned correctly to provide consistent image input. Any misalignment, differences, in exposure, or time synchronization errors can lead to artifacts in the final in the final 3D reconstruction.

6 FUTURE WORK

Immediate extension to this work includes optimizing the training workflow of 3D Gaussian Splatting, which remains a significant bottleneck in achieving real time scene reconstruction. While 3D Gaussian Splatting excels in rendering speed, enabling users to move around within the scene, it still takes a really significant time to training the model. This limitation makes it impractical for continuous updates, preventing truly dynamic and interactive experience. If the training process can be optimized to just a few seconds, this approach would become significantly more viable for real-time immersive communication. This direction of future work can require leveraging hardware acceleration techniques, such as parallel processing GPU (as currently implemented in CUDA for NVIDIA GPUs) to speed up training iterations. Another possible approach to improving real time 3D Gaussian Splatting workflow is to integrate a auxiliary pre-trained model that stores and processes common scenery details, reducing the computational burden of training each new scene from scratch. instead of requiring full training for every fame, this auxiliary model could provide pre-learned features, allowing the system to generate new scenes with minimal additional computation.

Another promising direction is the idea of a hybrid approach, which leverages Computer Vision-based tracking algorithms (e.g., YOLO) alongside Gaussian Splatting to reduce computational bottlenecks while enabling more dynamic scene interaction. Since multiple cameras are used to stream video footage to an application for processing, these same inputs can be used to simultaneously track object movement in real-time, enhancing the responsiveness of the reconstructed scene. The proposed method involves grouping the Gaussian Splatting point clouds into distinct objects after the initial 3D scene construction. Once segmented, a computer vision tracking algorithm can determine how much each object moves across consecutive frames. By incorporating data from multiple cameras, a 3D motion vector can be computed for each object, which can then be applied to the corresponding Gaussian Splatting point group to simulate real world movement. This approach allows us to be disburdened from the computation bottleneck of Gaussian Splatting, but interact with the scene more like VR game with like movement tracking. However, this approach still requires reconstruction of the scene when a new object is introduced to the scene, which we would either have to reconstruct the whole scene from scratch again, or we can just construct an artifact of the object and have it integrated in to the scene. Also, since we are using three dimensional motion vectors constructed from CV based tracking algorithms, it could be hard to recreate facial expressions as tracking of facial features often requires more detailed frames and more precise tracking systems.

Additionally, beyond VR based implementations, future research could explore how Gaussian Splatting can be adapted for projection based systems. This break through would eliminate the need for VR headsets, which can create a more immersive and natural communication experience. Unlike VR, where users are confined to wearing headsets, a projection based approach could enable holographic style displays, allowing participants to see and interact with each other in 3D space without any wearable devices. Moreover, this approach also addresses a common challenge in scene reconstruction, where users wearing headsets often appear with their devices captured in the generated 3D model, obscuring their facial expressions. By removing the need for VR headsets, Gaussian Splatting powers projections could provide a more authentic representation of participants, enhancing the sense of realism in virtual communication.

7 CONCLUSION

This work explores the integration of 3D Gaussian Splatting with VR view models, demonstrating its potential as an accessible alternative to hardware-dependent immersive communications methods. By implementing offline streaming of pre-constructed 3D Gaussian Splatting models in Unity, we demonstrated the feasibility and challenges of real time 3D scene reconstruction. While Gaussian Splatting provides a promising approach for immersive video communication, the computational cost of real time scene updates remains a major limitation. To address these challenges, we could explore the possibilities of hybrid solutions, such as combination with computer vision based tracking techniques for object movement updates, or significantly improve the training speed of Gaussian Splatting. This work reinforces the idea that 3D reconstructions do not have to be limited to hardware setups, and if further research on improving the work flow of 3D Gaussian Splatting construction or other ways to hack around the problem, Gaussian Splatting could forever change how people view virtual meetings or remote collaboration, with its imagery quality and ability to move around within scenes.

ACKNOWLEDGMENTS

Special thanks to Dr. Douglas Lanman, Evan Zhao, Shaan Charttrath, and John Akers for invaluable feedback and Hardware/Software support. We also recognize and have a special thanks to Aras Pranckevičius for code base of the prototype of implementing Gaussian Splatting in Unity which made this project possible.

REFERENCES

REFERENCES

[1] J. Stephens, *InstantSplat: Sparse-view SfM-free Gaussian Splatting in Seconds*, GitHub Repository, 2023. Available: https://github.com/jonstephens85/InstantSplat_Windows
[2] A. Pranckevičius, *Gaussian Splatting Playground in Unity*, GitHub Repository, 2023. Available: https://github.com/aras-p/UnityGaussianSplatting
[3] J. Kerbl, T. Leimkühler, T. Müller, A. Keller, *3D Gaussian Splatting for Real-Time Radiance Field Rendering*, arXiv preprint, 2023. Available: https://arxiv.org/abs/2308.04079
[4] Google, *Project Starline*, Available: https://starline.google/
[5] Apple, *Apple Vision Pro Guide*, Available: https://support.apple.com/guide/apple-vision-pro/tan1e660fd7d/visionos