Interactive Point Cloud Visualization in Virtual Reality

Kade Mathers
Kmathers@ualberta.ca

Jince Liu
Jince@ualberta.ca

Ken Su
Zhengya5@ualberta.ca

Yiteng Cao
Yiteng9@ualberta.ca

Erturk Ocak
Ocak@ualberta.ca

1 Introduction

Our project will focus on a possible solution to remote forest inventory work. Forestry research and tree inventory collection currently operates by sending people to physically inspect and collect data in the field. This can be dangerous, expensive and takes a long time to accomplish. With the Government of Canada updating the Beyond Visual Line Of Sight (BVLOS) rules in November 2025 so that BVLOS missions are now possible, many possibilities open up for the remote sensing industry. Using UAV LiDAR systems we can collect high density 3D point cloud data from up to 49 km away. This long distance data collection could allow us to have a virtual presence in the forest in areas that may otherwise be inaccessible or very expensive to have a physical presence in. Our project aims to use a LiDAR point cloud captured by the Zenmuse L2 to allow the user to have a virtual presence in that forest area. To demonstrate the visualization, a LiDAR point cloud captured by a DJI Matrice 300

RTK equipped with an L2 LiDAR scanner is used along with a TLS scan by a RIEGL VZ-400i. The acquired point clouds are processed and rendered in the Unity game engine, with deployment targeting the Meta Quest 3 virtual reality (VR) headset. The technical objective is to evaluate the feasibility of rendering real-world point clouds in a standalone VR system while maintaining acceptable performance and visual quality. We also have a simple pipeline using unity prefabs so it is easy to set up and run with new data Unity is used as the visualization platform due to its strong XR support, while the Meta

Quest 3 provides an accessible standalone VR environment. User feedback suggests that immersive point cloud visualization improves spatial perception and understanding of complex environments. The results demonstrate both the potential and current limitations of point cloud rendering on standalone VR devices, motivating future work in performance optimization and interaction design.

2 Implementation

2.1 Point Clouds

For our project we used two collected point clouds. One was collected with a UAV L2 LiDAR system, and another was collected with a Terrestrial Laser Scanning (TLS) System. The UAV scan is easily rendered as it is much less dense, and most of the density is in the crown which is farther away from the player and can have lower levels of detail. The TLS cloud is much denser and harder to render, and most of the points are close to the player. The TLS cloud requires much more extensive optimization to run, but it is viewable with our techniques at a high framerate at the cost of details and render distance.

2.2 Shaders

Our implementation consisted mostly of a shader rendering system for the point cloud so that is was viewable in a VR stereo headset with a high framerate. Our shader reads in each point from a ASCII

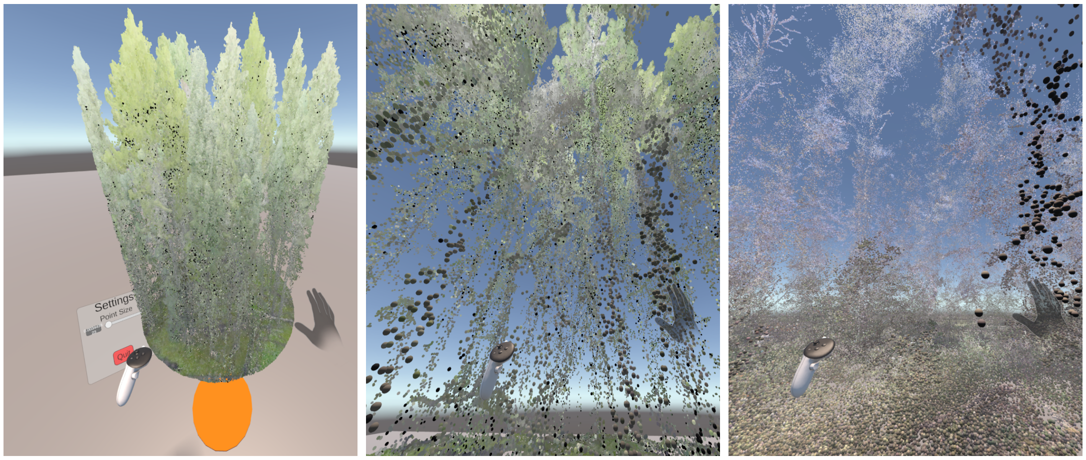

Figure 1: Left: The 1:25 scale rendering of the UAV point cloud. Middle: The full scale rendering of the UAV point cloud. Right: The Full scale rendering of the TLS point cloud. All images are shown in the left eye of the headset

".ply" file and then stores its location and color in a unity vector list. The shader then uses the camera location to render a camera facing circle that is colored and shaded to appear as a sphere with the location and color of the point in the point cloud. The shader uses a few techniques to speed up the rendering though so that the approximately 1.5 million points can be shown at a comfortable framerate. The first technique is that it uses flat circles instead of spheres to simulate points. These circles take much less time and resources to render and look identical to sphere in almost all scenarios. Another technique used by the shader is distance based levels of detail for the sphere. The farther away the sphere is, the less shading it is given, and after a certain variable distance the circles are unshaded, and then after another variable distance the points are no longer rendered so that far away and not visible points are not taking up resources. This implementation is relatively simple but is held back by the data loading method. Large point clouds will require more ram to hold the millions of points in memory, and the points also take a long time to load. This could likely be sped up with parallelization, but that is not implemented here. For each point cloud it is most effective to tweak the render distance, and the distance to unshaded points to increase the framerate and optimize visuals.

2.3 Interaction

The most important part of a VR experience is interaction with your environment. We have implemented very simple interactions with our project as it is mostly focused on the viewing aspect rather than interaction directly. We have basic movement and the ability to use a menu to resize points, resize the point cloud as a whole, and also adjust rendering distances and location of the point cloud. This enables the user to do simple manipulation of the point cloud and view it in interesting ways.

3 User Reviews

Users generally reported some confusion initially as they are unfamiliar with what the point cloud structure is, but after looking around for a bit they began to understand how it represented the forest structure. Some complaints included that the UAV point cloud is a bit sparse at lower levels, but is still easily understood. Players generally enjoyed looking at the small scale point cloud first as with the points closer together with a high framerate it was easy to understand exactly what they were seeing before diving into the full scale version. The TLS point cloud had the most issues with visibility and framerate, and because of this we may need further optimization before using point clouds with more than 2 million points. The reduction in shading and render distance generally made it harder to visualize and resulted in a poorer experience. Overall users enjoyed seeing the data, and knowing it was a real place they were looking at rather than a fictional model seemed to add novelty to the experience. Small scale UAV point clouds seem to result in the best user experience so far.

4 conclusion and future objectives

Our project shows an excellent proof of concept for the short time working on it, and proves the feasibility of VR point cloud rendering. Very few implementations of this kind of pipeline exist, and the benefits to teaching and/or basic visualization and data collection for the industry are being overstepped. This concept is applicable well beyond the scope of forestry, and could be used to allow users to experience any distant area that can be scanned with point cloud imagery.

Continued work on the shader system for rendering would greatly improve our results. Methods such as better LOD's for distant points, and culling of points out of view are just some simple additions that may greatly improve performance. Once performance is increased it would be beneficial to improve the visuals with better blending of points so that the structures are more clear. For interaction, a better menu system along with more interactive measuring tools would provide a greate benefit to our project. We would like to add a measuring tape, the ability to place markers, and the ability to fly up to the canopy and save measurements taken in the program to an exportable csv. These additions would make the program a useful and interactive product for teaching and collecting data.