Virtual Reality Point Cloud Annotation

Anton Franzluebbers
University of Georgia
Athens, Georgia, USA
anton@uga.edu

Changying Li
University of Georgia
Athens, Georgia, USA
cyli@uga.edu

Andrew Paterson
University of Georgia
Athens, Georgia, USA
paterson@uga.edu

Kyle Johnsen
University of Georgia
Athens, Georgia, USA
kjohnsen@uga.edu

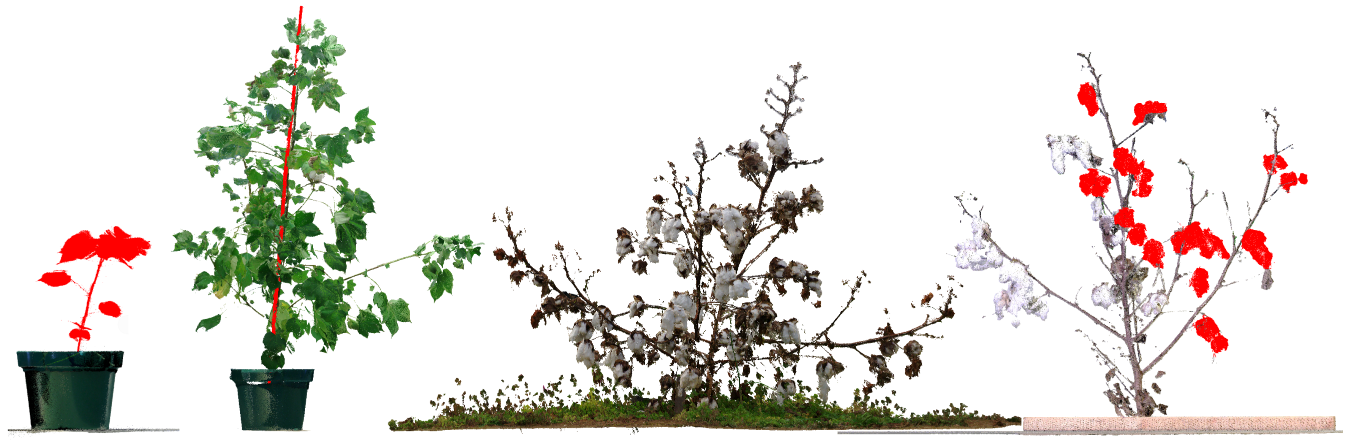


Figure 1: LIDAR scanned cotton plants visualized as point clouds in various states of annotation used for the user study. The two young cotton plants on the left were used for an annotation task, and the two mature plants on the right were used for a cotton boll counting task.

ABSTRACT

This work presents a hybrid immersive headset- and desktop-based virtual reality (VR) visualization and annotation system for point clouds, oriented towards application on laser scans of plants. The system can be used to paint regions or individual points with fine detail, while using compute shaders to address performance limitations when working with large, dense point clouds. The system can either be used with an immersive VR headset and tracked controllers, or with mouse and keyboard on a 2D monitor using the same underlying rendering systems. A within-subjects user study (N=16) was conducted to compare these interfaces for annotation and counting tasks. Results showed a strong user preference for the immersive virtual reality interface, likely as a result of perceived and actual significant differences in task performance. This was especially true for annotation tasks, where users could rapidly identify, reach and paint over target regions, reaching high levels of accuracy with minimal time, but we found nuances in the ways users approached the tasks in the two systems.

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; • **Computing methodologies** → **Point-based models**.

KEYWORDS

virtual reality, point clouds, annotation, visualization

ACM Reference Format:
Anton Franzluebbers, Changying Li, Andrew Paterson, and Kyle Johnsen. 2022. Virtual Reality Point Cloud Annotation. In *Symposium on Spatial User Interaction (SUI '22), December 1–2, 2022, Online, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3565970.3567696

Many important real-world applications obtain and process digital representations of built and natural environments. These surveying activities are vital for scientific and engineering pursuits, as they provide a means to measure physical traits of the environments and how they change over time. Commonly, scans of the environment are created using methods such as light detection and ranging (LIDAR) or photogrammetry, which output discrete spatial samples, known as point clouds. From here, a variety of automated and manual tasks ensue. Tasks involving human interaction with these point clouds, such as counting, measuring, and annotation, may be well-suited for immersive virtual reality (VR), which, through stereoscopic, head-tracked rendering, could provide a better sense of measurement depth and also improve navigation performance. Further, tracked controllers commonly used in VR may improve selection of point data. Though immersive LIDAR visualization tools exist (e.g. LIDAR Viewer [12], there is little evidence that they improve task performance over desktop-based tools (e.g. Cloud Compare or Mesh Lab). To both address and better understand this gap, we aimed to design a VR-based tool. This paper reports on that tool, including a study that addresses the critical question of how VR may improve task performance relative to desktop tools.

Beyond visualization, one of the contemporary use cases for such a tool is to provide labelled training data for automatic semantic segmentation, which allows higher level information processing. Manual annotation is required, as can be seen in public datasets with pre-annotated features, such as the DublinCity [32], Semanticposs [20], and Semantic3D [9] datasets. These are often quite large in scale, and are generally useful for self-driving vehicle applications, as this is one of the foremost uses of raw point cloud data for the present and near future. As machine learning methods become more widespread, this technique is being applied to a wider variety of fields, and the development of these new algorithms requires new ground-truth datasets, which often do not exist for application-specific annotation, counting, or classification problems. The intent of this work is not to contribute to this body of datasets directly, but to present and analyze a system that makes the development of new datasets more accurate and efficient, especially for applications that require datasets that do not exist, since their focus is too narrow or too innovative for an existing effort to have occurred.

This work was motivated by one such application, a research project aiming to automate and accelerate the measurement of agricultural products. As a part of this project, LIDAR point cloud datasets of cotton plants were produced, and then subsequently processed and annotated, either directly using a point cloud visualization toolkit, or more recently, by automatic annotation algorithms. The development and training of these algorithms via machine learning depends heavily on a large variety of point cloud data that are annotated correctly on a per-point level, or "ground-truth," through which their effectiveness could be evaluated. Given that this was seen as a tedious task using existing tools, we aimed to develop a tailored tool optimized for the process, and were particularly interested in the value of headset VR interfaces.

VR displays have immediate benefits in depth perception over traditional 2D displays through both stereoscopic display technology as well as parallax shift from motion tracking [8, 29]. The interaction affordances of scenes viewed in this manner may also change, especially when combined with 6-Degree-of-Freedom (DoF) motion controllers. For example, users may reach into the point cloud, directly selecting points, or move about the environment differently, such as by walking and turning in the real world. Enabling these actions with dense, natural point clouds is a major performance challenge. VR experiences must run at high frame rates to remain usable and to reduce simulator sickness, but incur significant rendering performance overhead due to high resolution stereoscopic rendering. To mitigate this, our application uses a continuous level of detail (LOD) system, with the novel variation of applying LOD emphasis on the hand positions, facilitating annotation work. We also addressed performance issues encountered when selecting points, and by doing this, enabled fluid, high frame rate point cloud interactions in immersive VR.

In addition, we aimed to quantify the task performance gains with this system, especially compared to a desktop interface. Thus, we also co-designed such an interface that could be used for the same tasks, but was optimized for mouse-keyboard-monitor interactions. These interfaces were compared through a within-subjects user study (N=16), where participants performed annotation and counting tasks on LIDAR scans of cotton plants and provided feedback on usability. Large performance advantages were found for the VR interface on the point annotation task, especially during early phases, where precision may be less important, but this advantage became much smaller as the task neared completion. Counting performance was similar, with a small, but significant advantage for the VR interface. Surveys and written responses qualify these results, highlighting that users strongly preferred the VR interface for both tasks, mostly due to a perceived performance advantage (even though this was not always the case). Altogether, results were promising that our VR interface could accelerate, and make more enjoyable, the currently tedious tasks on point cloud datasets, especially given that there is significant room for improvement in VR hardware and interface design.

1 RELATED WORKS

1.1 Immersive LIDAR Visualization

Immersive data visualization became practical in the early 1990s with the introduction of the CAVE [6], a cube of projection displays that surround the user. CAVEs, and similar spatially immersive displays, have since been used for a variety of visualization applications in architecture, astronomy, mathematics, biology, medicine, and physics. [5, 18]. Kreylos et al. was one of the first to explore large-scale immersive point cloud visualization and interaction, resulting in the open source LIDAR Viewer application [12], which has been used extensively for geographic science visualizations [14, 26]. This system was originally designed to be used with CAVEs, which provide access to high-performance computing and support multiple users [10]. Though visualization walls have remained popular [7], recent efforts have brought LIDAR Viewer and other similar systems to personal computers and personal VR headsets [11, 13, 17, 23, 31], greatly lowering the barrier for use.

Performance has been a dominant theme in the above LIDAR visualization research. LIDAR point clouds tend to be large, greatly exceeding the boundaries of system memory, requiring so-called "out-of-core" techniques that pre-process and subdivide datasets into hierarchical levels-of-detail (LOD), loading increased LOD into memory as the camera approaches. The exceptional performance of this technique is highlighted by web-based point cloud viewers, such as Potree [23] that gracefully reduce performance to run on most hardware. However, Schütz et al. point out that it is worthwhile to explore the maximization of in-memory point cloud visualization, i.e. the maximum number of points that can be precisely visualized in real time. They presented a continuous LOD method that they classify as a "view-dependent, continuous LOD method with fidelity-based simplifications [24]." Their system creates a lower resolution version of the full point cloud based on both gaze direction and viewpoint position and is updated regularly to account for movements of the camera using a compute shader. This method was found to effectively render point clouds of up to 104M points with a GTX 1080 on an HTC VIVE Pro. The continuous LOD system was found to have more "subtle and less irritating changes of detail as users move through the scene" compared to discrete LOD methods. Another, related approach furthers the idea of fully in-memory rendering, dividing work to render the cloud over multiple frames [25, 27].

Commercial software solutions also exist for the purposes of visualizing large points clouds both on desktop displays and in VR, such as Nubigon [1] and Interviews3D [2]. Our work builds upon existing efforts, with a focus on agricultural LIDAR datasets that have vast differences in point density, owing to the need to capture fine features of plants by capturing data from multiple perspectives. We focus on in-memory visualization, as our tasks are isolated to single plants at a time, and hence levels of detail offer few advantages.

1.2 Annotation Interfaces

Annotation is the process of ascribing data beyond that contained in the original data set, and is a common goal for machine learning algorithms to complete automatically. For scanned environments, this is often assigning each point or region a label (e.g. chair, teapot). Algorithms for this tend to rely upon supervised machine learning techniques, which in turn rely upon manually annotated data as a ground-truth for training and evaluation. For example, there are several large pre-labeled points that are available to use for this kind of training, such as the TUBS Road User Dataset with 12,000 labeled scans [22], the H3D dataset with 1 million labeled objects from 27,000 scans [21], or the semantic3D.net dataset with over 4 billion points, which was created as a benchmark for other machine learning classification algorithms [9]. These datasets were each annotated using either manual or semi-automatic methods, making annotation a key, though tedious, part of the pipeline, and the interface to this has received attention from the 3DUI research community.

Wirth et al. investigated whether the advanced 3D visualization and input capabilities of an immersive VR headset would improve annotation performance [30]. Points were selected by placing and scaling boxes using the position and rotation of the 6-DoF Oculus Rift Touch controllers. The authors then evaluated the performance of the technique using a method detailed in Monica et al. [19], basing the evaluation on both time to completion and accuracy of the annotation result. The work of Monica et al. made use of manually placed control points in the middle of segment regions, and then calculated the correct segment for all the points in the cloud based on "distance" using a cost function based on position, normal, and color of the points. Wirth et al.'s VR method compared favorably in all metrics to this method of control point based selection.

Li et al. developed a similar point cloud annotation software called SUSTech POINTS [16]. The software is designed as a web application to be used on a computer monitor. The main interface is divided into several sub-views showing top, side, front, and perspective views, along with a context photo of the environment from which the point cloud was scanned. 3D annotation is performed by creating and transforming boxes around the selected points. Once a box is placed, the user has options for adjusting its 3D transform using gizmos. The box size can be adjusted automatically by scaling to tightly enclose the points it contains. Since the tool is designed for use on LIDAR data streams, which contain consecutive frames of similar data, an algorithm enables the transfer of annotations between frames. This allows the user to then manually adjust the annotation to fit the new data. Annotation efficiency was measured using both accuracy and speed as metrics.

Several other works have investigated the use of alternate input methods to improve the point cloud annotation or manipulation process. BioVR uses the Leap Motion hand tracking system to eliminate the need for tracked controllers while visualizing and manipulating similar bio-informatics datasets [31]. Bacim et al. designed a system to annotate point clouds based on the concept of repeated bisection of the data [1]. Due to the reduced input complexity of bisection, they were able to use a Leap Motion to perform the slicing. In order to change the viewpoint of the camera, a six-degree-of-freedom 3DConnexion SpacePilot Pro 3D mouse was used. The authors noted the use of progressive refinement was critical to allowing a less precise input device such as the Leap Motion to still provide sufficient annotation accuracy. Veit et al. proposed a point cloud annotation framework that makes use of a tracked smartphone for the selection of points [28]. The touchscreen makes text entry for the labels convenient. HyFinBall was developed to meld the advantages of traditional 2D inputs and 3D natural user interfaces by using multi-touch ball input devices that can be used while resting on a surface [4]. The authors tested their device by exploring LIDAR datasets.

It has been found that VR offers advantages over 2D modalities for complex 3D visualization tasks, and our work confirms these findings using using modern hardware and a new applied task [2, 3]. Surprisingly, few works have evaluated differences in task performance between different user interfaces for this type of application [15]. As such, alongside our designs for the immersive and non-immersive interfaces, we also provide evidence of how the immersive interface improves certain aspects of the annotation process.

2 SYSTEM DESIGN

As mentioned in Section 1, our system was designed for visualizing point cloud datasets acquired through LIDAR scanning of cotton plants, with the primary task of annotating point clouds at the individual point level (as opposed to bounding regions). The raw data were obtained with a Faro Scanner Focus 3D, a rotating LIDAR system that provides colored point clouds. As multiple scans had to be taken to account for occlusion, these were first merged into a

[1] https://www.nubigon.com/
[2] https://www.3dinteractive.de/products/interviews3d/

single, high density, point cloud (see Figure 1) using the CloudCompare software. The data were then exported into the Point Cloud Library's point cloud data (PCD) file format, which is an emerging standard and offers significant interoperability between point cloud tools. From here, they can be processed by our application.

Our application[3] was designed entirely within the Unity3D game engine (version 2020.3 LTS), which supports a wide range of operating systems and hardware. However, it does not natively load PCD files, which required a custom importer and exporter to be built. To minimize time consuming steps in the pipeline between CloudCompare and Unity, PCD files are parsed and loaded directly into memory and then briefly processed to determine their extents such that their size can be automatically normalized. We use a compute shader for this processing, which takes less than 10 ms for point clouds of several million points on an NVidia Titan V graphics processing unity (GPU). All data is loaded into a compute buffer on the GPU, with 20 bytes stored per point; 12 bytes are used by the 3 single-precision floating point numbers for the xyz position of the point, and the remaining bytes are used to store color, assigned layer, and a reference, or "correct" layer, which was only used for the user study to give immediate feedback on completion percentage. Without this last reference value, the data could be packed into 16 bytes.

2.1 Point Cloud Rendering and Annotating

Immediately upon loading our first point clouds, which are extremely dense relative to their real-world scale (a cotton plant is roughly a 1-meter cube, with scans being in some cases over 7 million points), we found significant performance issues when attempting to visualize them in VR at interactive frame rates. Modern GPUs are capable of processing and rendering millions, even hundreds of millions, of triangles per frame at interactive rates. However, doing so while all triangles are visible on screen (i.e., they are not culled and are shaded), with significant overdraw (triangles are not sorted), and while processing them for annotation tasks, severely limits performance. This is compounded by the high resolution and frame rate requirements of VR systems. These problems do not typically occur with lower-density clouds such as environmental scans, as an LOD system can automatically reduce the number of in-memory points being rendered based on camera distance. Doing so when all points are reachable and clearly visible introduces quality issues, as details pop in and out.

In order to maintain high visual quality and detail while remaining performant with the high requirements of a VR-enabled application, our system implemented a continuous LOD similar to that presented in the work by Schütz et al. [24]. However one additional feature was implemented to improve the utility of the downsampling technique when dealing with precise input. While the original work changed density based on view direction and head gaze, our system uses the hand positions to smoothly increase the point cloud render density in the target area. This allowed the area of focus to be guaranteed to have the maximum density, while limiting the total number of points to a modest point budget. To achieve this, all of the points are added to a compute buffer. A

[3]Source is available at https://github.com/velaboratory/DataFoldvr-Virtual-Reality-Point-Cloud-Annotation

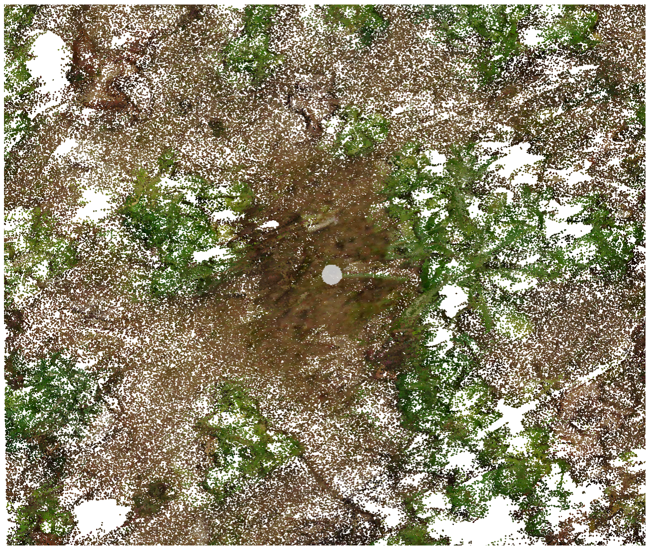


Figure 2: An example of the continuous LOD system used to render the point clouds. This system ensures that the full detail present in the source data is always visible in the area near the cursor/tracked hands, which happens to be in the center of the image here. The points in this image are reduced in size to exaggerate the low density areas and emphasize the effect.

compute shader is dispatched on the GPU every frame to select the appropriate subset of points to add to an AppendBuffer. This output buffer is then rendered using Unity's DrawProceduralIndirectNow method, avoiding the expensive operation of transferring data between the CPU and GPU memory. From here, the number of triangles rendered was variable (between 2 and 12), as produced by a geometry shader, ranging from a circular shape to a quad, depending on distance from the camera, based on Keijiro Takahashi's PCX example (https://github.com/keijiro/Pcx). The effect can be seen in Figure 2.

Beyond visualization, performance was a major concern when developing realtime annotation, and again compute shaders were used heavily to speed up the process. CPU-based implementations may use indexed data structures to retrieve close points to the cursor, and subsequently annotate them. However, this data is also needed on the GPU for visualization, and in our case, the high density of the point clouds would result in large transfers between the CPU and GPU. Instead, we inverted the problem for GPU implementation. For every frame that annotation is indicated over a designated region, a compute shader performs a distance calculation on the GPU for every point. If the point is within the region, the byte corresponding to the current layer index is changed to match the desired layer. During the rendering process, this layer index is used by the fragment shader to draw the points in either original color, a distinct solid color, or to hide the point. This allows the operation to proceed in parallel entirely on the GPU, greatly accelerating the process.

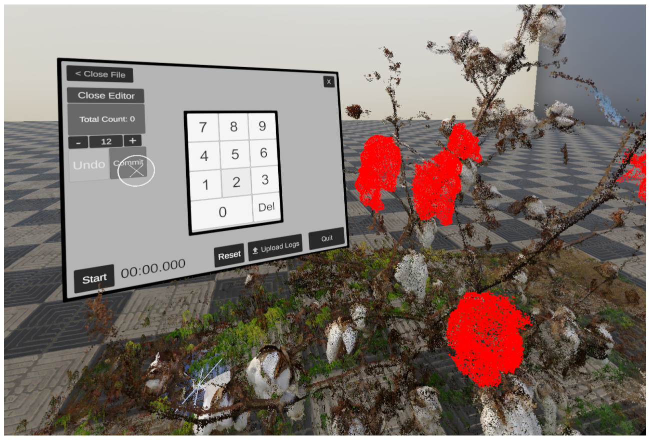


Figure 3: Screenshot of the counting task in the *VR* condition. The tablet interface is visible, on which participants entered their subtotals when counting groups of bolls.

2.2 User Interfaces

Two interfaces were designed for the study, which we abbreviate as our "*VR*" and "*2D*" interface going forward. The *VR* interface was designed for use with a stereoscopic VR headset and two 6-DoF controllers. The *2D* interface was designed for a single-monitor, keyboard and mouse setup. These interfaces could be swapped in real time, via a keyboard switch or by clicking on a button. When in *VR* mode, the monitor would show a mirror of the VR-view. Both interfaces could be in "counting" or "annotation" modes, depending on the task, which slightly varied the interface to show relevant feedback to users. In the study, these were switched by the experimenter, but would otherwise be switchable by the user, meaning that both *2D* and *VR* could be used in practice, though not simultaneously in the current implementation.

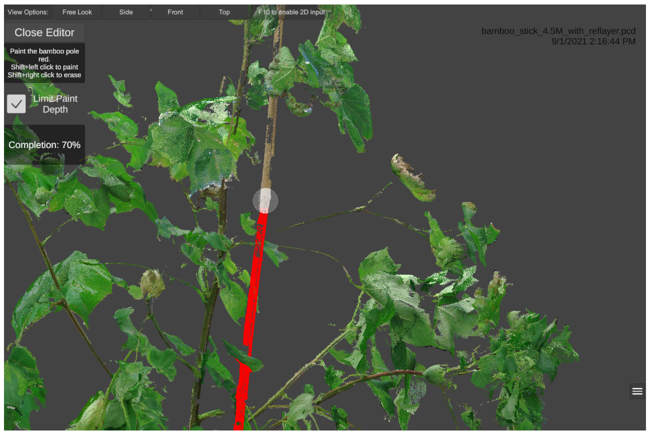
The *VR* condition interface is depicted in Figure 3. The user was spawned into a virtual room large enough to easily accommodate the point cloud, which was placed in the center according to its bounds. A virtual tablet was used to show task-relevant UI elements, which was most prominent in the counting task. All controls were identical on both controllers, eliminating the need for dominant hand preferences and simplifying the learning experience. The virtual tablet is called up by users by pressing the B or Y button on the Oculus Touch controller, allowing them to use the opposite controller to make selections (depicted by a 2D cursor that appeared on the tablet when the controller was held near it). In the annotation task, the completion percentage was shown near the lower part of the controller.

In order to assign points to a particular label, a painting metaphor was used. In the *VR* condition, this was done by visualizing a sphere around each of the users' hands, and allowing them to mark all the points within the sphere for every frame that the index trigger on that controller was held. To show the marked points, the color of the points was changed from the original scan color to a solid color. The user was able to select a layer using a color picker, though in the user study only a single color was used, and a secondary controller button (A or X) was used to erase previously painted points and return them to the original color. The spheres could be scaled by moving either controller thumbstick on its Y axis to allow for more broad or precise painting actions.

While the point clouds were relatively small in real-world scale, and thus walkable within a small space, a VR locomotion system was added to make the system usable in a seated position, and the study was performed seated to match the desktop interface. Rather than transforming the point cloud, as other systems have done [12], we chose to keep the transform between the virtual room and point cloud fixed and instead moved the user's position with a combination of pulling motions (either hand) and use of the thumbstick for rotations. When the hand grip trigger was held down, the world was pulled to match the controller position, with a small amount of momentum if the grip was released while moving. Smooth rotation (at a relatively high rate of 200 degrees/second) was activated by the X axis on the thumbstick. When combining smooth rotation and world grabbing, the origin of the rotation was moved to the hand position. In our study, participants did not show any significant difficulty getting used to the controls, and more experienced VR users were able to use this system to paint with one hand and move with the other hand simultaneously in one smooth motion. In addition, we did not have any reports of nausea or discomfort from users.

The *2D* condition interface is shown in Figure 4. In contrast to the *VR* condition interface, the user is shown only the point cloud, rather than a virtual room, as their environment is the real room. The on-screen interface, though custom designed, mimics the features found in other annotation systems, including the ability to choose from a variety of aligned orthographic views of the cloud, or to "free look" with the mouse. The mouse operated in two modes, one for navigation and the other for annotation. In navigation mode, the left mouse button was used to select, middle to pan the orbit point in view space, scroll-wheel to zoom, and right to orbit around that point. In annotation mode, which was accessed by holding the shift key, the left mouse button would annotate selected points, the right would remove the annotation, and the scroll-wheel would change the size of the selection region.

On a traditional computer display with mouse interaction, the selection of a specific sphere of influence is more difficult. Initially, the system was designed to paint all points overlayed by the 2D circular cursor, essentially creating an infinitely long cylinder of influence. While this difference between the *2D* and *VR* conditions could be considered fair because it arose as a result of a fundamental difference between the two systems, an additional feature was added to the *2D* condition to achieve a more similar spherical influence effect. To achieve this, a compute shader calculated and returned the points within a smaller circle 1/5 the size of the current cursor size to the CPU. The resulting points were then iterated on the CPU to determine the point closest to the camera. This point was then used by the annotation shader such that the depth of selection was limited to the diameter of the circular region (essentially placing a depth-limited cylinder at the cursor point). This approach prevented background points from being selected accidentally, as seen in Figure 5. We also included a toggle button for this behavior, as the infinite cylinder could also be useful, though the majority of participants left the feature on (the default). Before pressing the



Paint the bamboo pole red.
Shift+left click to paint
Shift+right click to erase

Completion: 70%

bamboo_stick_4.5M_with_reflayer.pcd
9/1/2021 2:16:44 PM

Figure 4: Screenshot of the annotation task in the *2D* condition. The semi-transparent circular cursor can be adjusted in size.

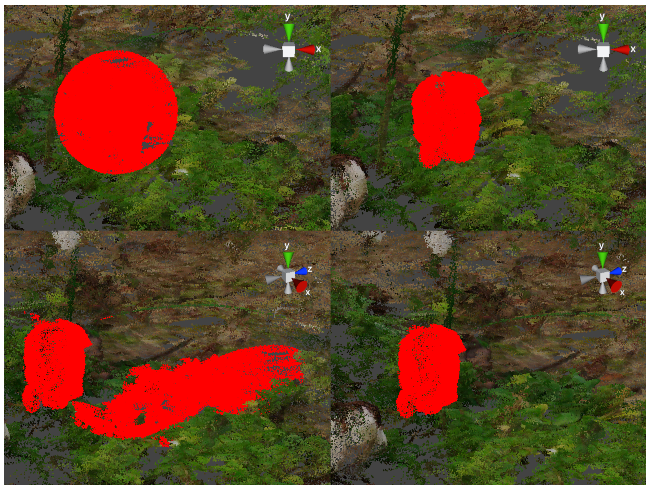


Figure 5: An example of the depth limiting algorithm used in the *2D* system. The top images show the initial painted view. The bottom two images show the same cotton boll from a rotated perspective after the paint operation. The left two images have the depth limit turned off, and the right two have the feature turned on.

button that performed the paint operation, the points were highlighted by increasing the brightness of the original color, as seen in Figure 6. This was especially useful for the *2D* condition with the depth limit feature enabled, since not all points under the cursor would be painted, though the highlight effect was visible in both conditions.

3 STUDY DESIGN

Our study was designed to evaluate and compare the two interfaces for two tasks that are commonly performed on point clouds: precise annotation and feature counting. A within-subjects, crossover design was used, such that each participant used each interface once for each type of task, resulting in a sequence of four tasks. The order of *VR-2D* was alternated in order to account for the learning effect, so that half of the participants used order *VR-2D-2D-VR* and half used *2D-VR-VR-2D*, with the first two instances being an annotation task, and the last two being the counting task. The order of the annotation and counting tasks was fixed, as there was no intention to compare between the two tasks and point clouds. In addition, we decided to use the same point cloud for completing the *VR* and *2D* versions of each task for every participant, as it would be difficult to find different point clouds with similar difficulty, avoiding a source of bias or variation in the timing between *VR* and *2D*. Each task consisted of a training phase and a testing phase, as described below.

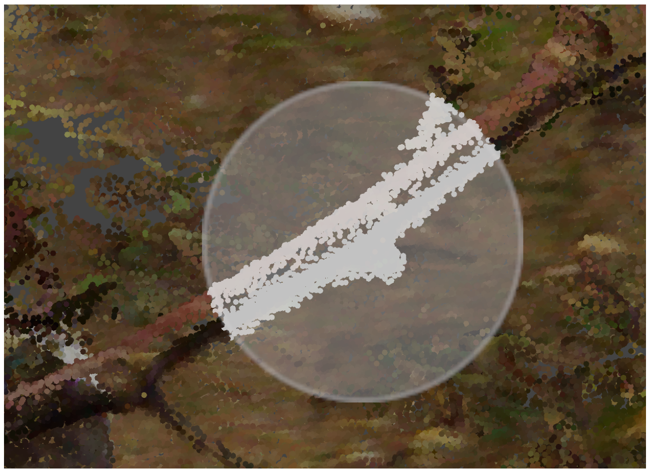


Figure 6: The highlight effect previewing the points that will be selected by a potential paint operation in the *2D* interface. In this image, the depth limit feature is preventing the background points from being selected. The *VR* condition also has this highlight effect for points within the hand spheres.

3.1 Tasks

In the **annotation task**, participants were instructed to precisely paint a portion of the point cloud according to the part of the object that it represents. In the training phase, the point cloud was a scan of a relatively small cotton plant in a pot. The task was to mark the points that belonged to the plant itself, excluding the pot, soil, and ground. The evaluation phase was a larger potted cotton plant with a bamboo pole used as a support. The task was to mark the points belonging only to the bamboo pole, which was colored significantly lighter than the rest of the plant or the plant's stem. As the plant was growing around the pole in all directions, this task was significantly more difficult, and impossible to perform from a single perspective, though the pole was nearly straight. The task was completed by achieving a 98 percent score. The score was determined by comparing the present annotation to a ground-truth annotation. As the ground-truth was also approximate, 98 percent was chosen empirically as being consistently achievable. The percentage indicated was the F_1 score (see Equations 1-3), which requires eliminating false positives (annotating incorrect points) and false negatives (not annotating the correct points), such that the user could not simply select the entire point cloud to achieve a high score.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \quad (1)$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (2)$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (3)$$

In the **counting task**, participants were asked to count cotton bolls that were scattered throughout the plant. This task would be challenging to perform in the field without picking the cotton. The training and testing tasks were similar, except that the testing task was a larger plant with more bolls. During the task, participants could mark already counted bolls with the painting interface. In addition, participants were given an interface to serve as a counter, and if they desired, to "commit" that count, which would erase the currently painted areas akin to picking the cotton. The task was completed when the participant indicated that they had counted all bolls.

3.2 Hypotheses

The following hypotheses were formulated based on prior work and pilot testing, which suggested that the *VR* interface would enable superior task performance and result in higher levels of user satisfaction.

(1) **Participants will be faster overall with the *VR* system in the counting task.** We expected *VR* to allow for more rapid identification of bolls, quicker marking of counted bolls, and easier navigation, resulting in faster performance.
(2) **Boll counts will be higher and more consistent when counting with the *VR* system.** The effects of stereoscopic rendering and smoother parallax motion present in the *VR* condition will result in fewer missed bolls due to occlusion and a more precise count.
(3) **Participants will report higher accuracy and perceived efficiency using the *VR* system.** Similar to the previous hypothesis, the *VR* condition will present information in a more intuitive format that will be perceived as better for these tasks.
(4) **Participants will prefer to use the *VR* system.** Influences such as the novelty effect and the more intuitive nature of the *VR* system will overwhelm any negative effects such as comfort caused by wearing the headset.

3.3 Study Population and Environment

Following human-subjects review approval, 16 participants were recruited from the local university population and lab groups to participate in the study by direct invitation and word-of-mouth. The study took place over the course of 3 weeks, and was performed within our research laboratory. For the *VR* condition, an Oculus Quest 2 headset (1832x1920 resolution) using the wired Oculus Link feature set to 90Hz and default 2448×2688 per eye rendering resolution was connected to a PC with a i9-7980XE CPU and an NVidia Titan V GPU. As described earlier, the application frame rate was verified to have remained at a constant 90 frames per second throughout the study, and the Oculus Link cable data rate was set to the maximum of 500 Mbps to avoid visible compression artifacts. Our study used a point budget of 3 million for point clouds of up to 7.8 million points, as seen in Table 1, though none of the participants remarked that they noticed the LOD system at all. This budget was empirically chosen because it offered sufficient visual quality for the selected points and fell well within the testing computer's performance limits, even while recording the screen, rendering to multiple monitors, and using the Oculus Link encoder. For the *2D* condition, a standard mouse and mechanical keyboard were used with a 32 inch 2560x1440 monitor at 60 Hz. We chose to retain the 3 million point budget for this condition to be consistent with the possible information available within the *VR* condition at any given time, although no LOD optimization was strictly necessary to maintain 60 frames per second.

3.4 Procedure and Measures

After getting informed consent, participants filled out a background survey with questions about demographics and prior experience with VR and point cloud data. As described earlier, the study consisted of two task-types, annotation and counting, which were repeated in the *VR* and *2D* conditions. Each task consisted of a training session as well as a timed session on a larger and more complex point cloud. During the training session, the participant was instructed by the researcher on how to perform the task and how to use the controls for both viewing and interacting with the point cloud. While the training task was timed, the participants were encouraged to take their time and get used to the controls. Once the participant completed the task on the training point cloud, they were moved on to the larger point cloud of the same task. They were instructed to complete the task as quickly as possible and were given minimal external instruction during the activity. After completing the task in VR, participants performed the exact same task using the same point clouds using the *2D* interface before switching to the second task in the opposite order. During both the *2D* and *VR* training periods, time was only logged while the participant, and not the researcher, was actively operating the controls themselves. We excluded the time when the researcher was demonstrating the controls. In both conditions, the participant was only moved on to the main task after completing the task and reporting that they were comfortable with the controls and task. Brief questionnaires were given between every task, asking for any sources of difficulties after each of the four tasks and relative preference scores between the *VR* and *2D* cases after both the annotating and counting groups of tasks.

Table 1: Point cloud sizes used in the experiment

Task	Point Cloud Size
Annotation Training	3,844,639
Annotation	4,485,971
Counting Training	2,609,825
Counting	7,867,195

4 RESULTS

All participants were able to complete all tasks successfully, though one participant was asked to restart the first 30 seconds of one of the tasks due to a software crash. The average age was 28 yrs (SD=9.26). Of the 16 participants, 12 reported male, 3 reported female, and 1 declined to identify. There was a mix of very experienced and inexperienced VR users, but the majority (9/16) were very comfortable with 3D video games. Completion of the study took between 30 minutes and 1 hour for all participants.

Using a Chi-square test for independence, no significant differences were found between the two groups assigned to different condition orders. In addition, none of these background variables (shown in Figure 7) correlated strongly with the performance variables, so they were not used as covariables in subsequent analyses. No motion sickness, vision problems, or discomfort were reported during the study.

Three relative preference questions were asked for each of the two tasks. Responses were recorded on a numerical scale with a response of 1 representing strong preference for the *VR* condition, and 7 representing strong preference for the *2D* condition. The questions and distributions are shown in Figure 8. Using a one-sample two-tailed t-test centered at neutral, all of the results showed significant preference for the *VR* condition at the p<.001 level, except for the question asking about relative speed/efficiency for the counting task, which was significant with p=.03.

The main objective evaluation metric for both the annotation and counting tasks was completion time. In the annotation task, participants were asked to continue the annotation until they reached 98% completion, but intermediate times to 80% and 90% were evaluated as well. These results are shown in Figure 9. Generally, there was a learning effect across subsequent trials for within subjects comparisons, but a paired two-tailed t-test between all the first- and second-attempt completion times found no significant difference. In the annotation task using a paired two-tailed t-test, the *VR* condition was found to be significantly faster to 80% (p=.002) and 90% (p=.001) completion, but no significant differences were found at the 98% completion threshold. For 80% completion, times were 55% faster in the *VR* condition (M=43.5, SD=31.7) than in the *2D* condition (M=97.7, SD=69.8), and for 90% completion, they were 51% faster in the *VR* condition (M=62.3, SD=46.1) than the *2D* condition (M=126.8, SD=79.4). No significant differences between the *VR* and *2D* conditions were found in completion times for the training session for either of the tasks using paired two-tailed t-tests.

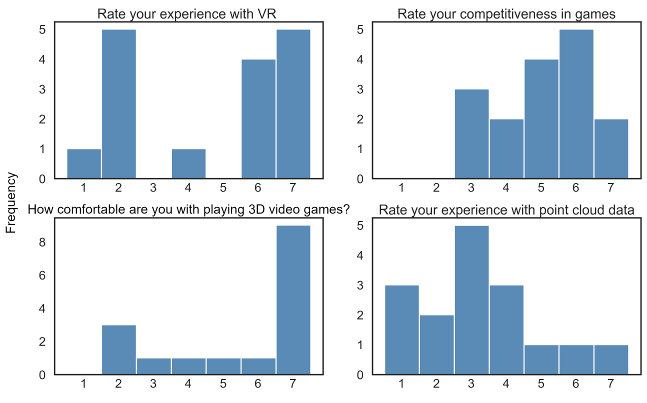


Figure 7: Background questionnaire results.

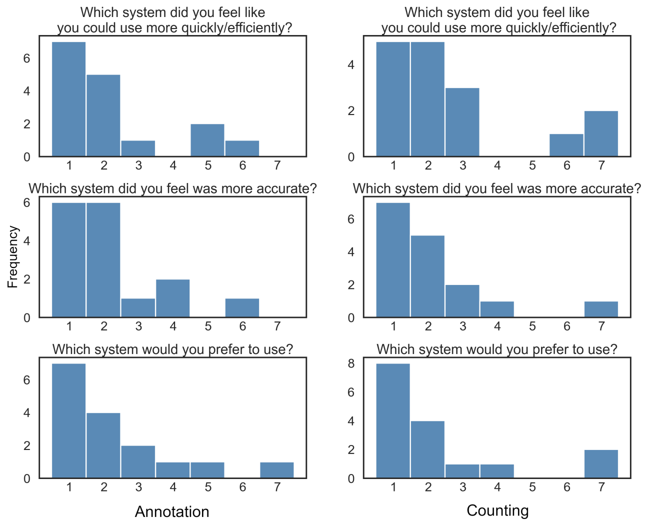


Figure 8: Relative preference between the *VR* and *2D* conditions. A score of 1 represents preference for the *VR* system, and a score of 7 represents preference for the *2D* system.

For the counting task, times were found to be 16% faster on average in the *VR* condition, which was found to be significant (p=.022) using a paired two-tailed t-test. While no true count was calculated to compare accuracy, as such a count would be inherently biased toward one of the conditions depending on how it was calculated, the average counts between the *VR* and *2D* conditions showed a trend towards higher counts in the *VR* condition with p=.051. Participants also performed significantly fewer commit operations in the *VR* condition (p=.008), meaning they kept higher numbers in working memory before entering their subtotals and restarting the count. The results for these measures in the counting task are shown in Figure 10.

4.1 User Feedback

Free response user feedback was requested after each of the four tasks primarily to discover any usability issues that would not otherwise be captured. In the annotation task, the most common response was the difficulty in visually finding the last few percent needed to achieve 98% completion, with 9 out of 16 users having a similar comment during their first attempt at the task. Two users reported difficulty as a result of depth estimation, and three users found it difficulty to annotate due to occlusion problems with the rest of the point cloud. These reports only occurred in the *2D* condition. No other significant usability issues were reported by participants. After all parts of the study were completed, general feedback was requested. One participant noted that "Precision with keyboard and mouse was a little better, but visual perception was much easier in VR." Another stated, "For the first task I preferred the 2D much more than the second task, where I preferred the VR.

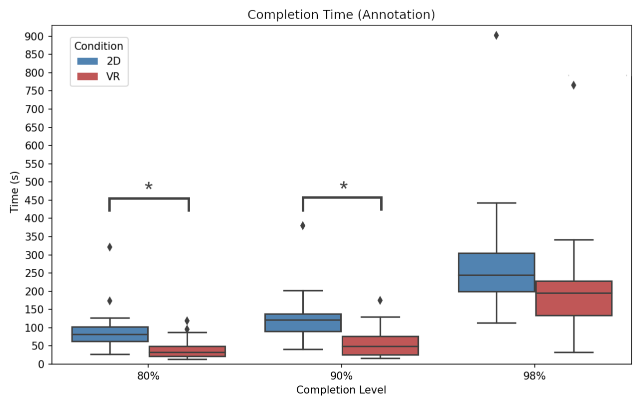


Figure 9: Completion times for the annotation task at the 3 target completion levels. Significant differences occurred between the *VR* and *2D* conditions at the 80% and 90% levels using a paired t-test.

I believe that has a lot do with the movement and accuracy needed. ... For the second task the 2d was harder because there was a lot more to pay attention to."

5 DISCUSSION

The results largely supported our hypotheses, though interesting nuances arose that were not expected.

H1: Participants will be faster overall with the *VR* system in the counting task. Confirmed. Completion time for the counting task was significantly faster in VR, though we were surprised that the magnitude of the difference was not higher. It should be noted (see H3) that participants also tended to find more cotton bolls in *VR*, likely increasing the amount of time required slightly due to the extra work involved.

H2: Boll counts will be higher and more consistent when counting with the *VR* system. Not confirmed. A trend towards higher boll counts was found in the *VR* condition, but the difference was not significant. Unexpectedly, the variance of the counts between the two conditions was also similar (in fact, count variance was higher in VR). This is an encouraging result, since the purpose of a system like this is for ground-truth gathering, and significant differences between the two conditions would cast doubt onto the reliability of either of the systems. Absolute accuracy of either of the conditions can not be calculated, as there were ambiguities in what parts of the plant should count as a boll. Participants were told to use their own judgement when determining whether to count a half-opened bud as a boll or not, but were asked to be consistent between the *VR* and *2D* conditions with these decisions.

H3: Participants will report higher accuracy and perceived efficiency using the *VR* system Confirmed. Both 7-point scale questions asking about perceived accuracy as well as speed/efficiency resulted in significant preference for the *VR* condition, even for the annotation task where the difference was not significant. Preference questions were presented as a reflection of the task as a whole, and participants were not aware of the 80% and 90% thresholds for timing used in the evaluation. The reason for this discrepancy in

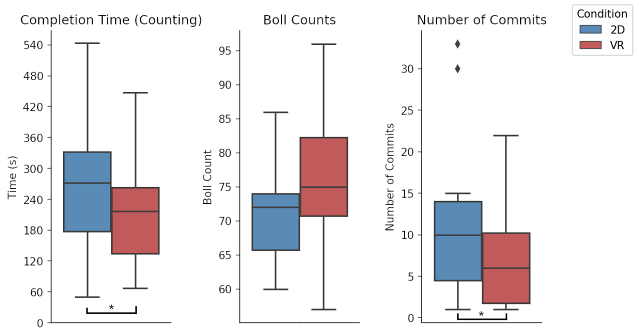


Figure 10: Output measures from the counting task. Significant differences were found between the *VR*/*2D* conditions for completion time and number of commits using a paired t-test.

the perceived and actual speed in the annotation task may suggest an altered sense of time caused by a more engaging and novel experience in the *VR* system, rather than the more familiar desktop interface. This effect may also have had an influence on the results of the final hypothesis.

H4: Participants will prefer to use the *VR* system. Confirmed. Both the annotation and counting tasks showed significant preference for the *VR* system. This was further verified by participant feedback.

Other findings: Though we did not form any hypotheses about cognitive load of the task, the commit interface of the counting task offered a way to delve into this aspect. We found that a significantly lower number of commit actions were performed in the *VR* condition than the *2D* condition during the counting task. While not entirely unexpected based on our pilot usage of the system, this showed a willingness of the participants to keep larger counts in working memory compared to the *2D* condition. There are several effects that could have contributed to this result. It is possible that the *VR* condition allowed for a more intuitive and instinctive painting operation, reducing the mental workload of the painting and leaving more room for the actual counting process. Another possible explanation for this effect is the reduction of clear continuity breaks in the *VR* system. In the *2D* condition, every time the camera perspective is changed, the user is presented with a visually different scene with few corresponding elements to the previous image, but in the *VR* condition, perspective is changed more gradually through head movement and locomotion by hand movement. Combined with the effect the stereoscopic rendering has on perception of the scene as a 3D object rather than a projected 2D image, the *VR* condition is a more continuous experience, while the *2D* system can conceptually be broken into smaller segments of panning movement separated by orbit interactions. These breaks may have provided a better opportunity for entering the subtotals. The last potential reason for the behavior could be increased friction in entering the total using the virtual keypad in VR, though none of the participants reported this as a source of difficulty.

We also investigated the completion times for the annotation task. In an attempt to characterize the completion time curve between the two conditions, completion times were compared at a few checkpoints, including 98%, 90%, and 80%. 98% represents the maximum easily achievable completion percentage as seen in informal pilot trials during development, and 80 and 90% were chosen as regular intervals below this point. Comparing checkpoints much lower than this completion percentage was not expected to provide useful data in the completion of the task, since a single rough painting stroke could already achieve 50-70% F1 score in tests. The majority of participants did not use the same method as predicted by the authors in the *2D* condition. The originally conceptualized method involved painting roughly down the length of the point cloud to cover the target bamboo stem, then rotating the camera by 90 degrees and erasing any areas that were previously not visible and were false positives. However, this method assumed that participants would disable the depth limit feature, which they generally did not do. With depth limiting turned on, the completion trajectory was more linear. In order to get to 90% completion, only a general concept of the structure of the bamboo stem was needed in VR to paint broadly through the center of the point cloud. However, in order to achieve the full 98% dictated by the completion guideline, the task switched to more of a searching problem. The greater angular resolution of the 32" 1440p monitor may have contributed to better performance in the *2D* condition during this part of the task. In addition, it is possible that visual search for small details was inhibited by stereo convergence, which requires not only searching a 2D visual space, but to converge to multiple depth planes.

6 CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

This work designed and evaluated the potential of an immersive VR system for point cloud visualization and annotation of agricultural LIDAR datasets. The system enabled users to annotate, at the individual point level, large clouds at interactive frame rates in immersive VR with the help of a continuous LOD system that emphasized details at controller locations. We also designed a desktop interface for non-immersed use and comparison in a user study. Our study findings suggest that the immersive VR interface would increase both user performance and user satisfaction, though its efficacy likely diminished during latter stages of completion, requiring more detailed search and manipulation.

Our study had several limitations. First, participants had a relatively high overall level of expertise with VR and 3D modeling programs, game playing and some with experience using point cloud tools. This recruitment was intentionally designed to avoid issues that we may have faced with a lack of familiarity with such tools within the general population, which may have created a bias in our outcomes. This is, perhaps, why we did not find any evidence of nausea from using the VR locomotion interface, though in practice, more comfortable techniques such as real-walking could be used. The gender disparity that arose from this recruitment also limits the generalizability of this work to the general population. Another limitation was that we did not give users multiple attempts with each Task-Interface combination, aside from the training-testing phases of each task. This compromise was made due to the already long study duration and worry about fatigue. It is unclear which interface is fastest to learn, and if additional training would reduce the difference in performance. Finally, the study compared two hardware interfaces that differed in many ways (stereoscopy, resolution, field of view, controller DoF, etc.), and hence we cannot ascribe the differences in outcomes to particular features. Instead, we optimized the application interface to the available input and output devices, in an attempt to minimize the difference in user performance between the two systems. A lingering issue with this approach is that there is no effective limit to this optimization process. In all likelihood, both interfaces could continue to be improved for the task, though VR interfaces may have more room to grow. For example, we could reduce hand jitter during painting operations in VR, by increasing the control-display ratio, or by adding interactions that allow the user to scale the point cloud dynamically to focus on a specific region at a more comfortable size.

In fact, we see the greatest future value in combining the two interfaces. The current application is able to switch between modes in the middle of a task, and could be made to function simultaneously (two users) to accelerate the work. Already, mixed reality displays are emerging that allow for effective use of planar displays and 3D content. The benefits of both systems could be used to complement each other, and deeper understanding of their corresponding strengths could result in higher quality user experiences in the future.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1934481.

REFERENCES

[1] Felipe Bacim, Mahdi Nabiyouni, and Doug A Bowman. 2014. Slice-n-Swipe: A free-hand gesture user interface for 3D point cloud annotation. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 185–186.

[2] Doug A Bowman and Ryan P McMahan. 2007. Virtual reality: how much immersion is enough? *Computer* 40, 7 (2007), 36–43.

[3] Doug A Bowman, Ryan P McMahan, and Eric D Ragan. 2012. Questioning naturalism in 3D user interfaces. *Commun. ACM* 55, 9 (2012), 78–88.

[4] Isaac Cho, Xiaoyu Wang, and Zachary J Wartell. 2014. HyFinBall: a two-handed, hybrid 2D/3D desktop VR interface for multi-dimensional visualization. In *Visualization and Data Analysis 2014*, Vol. 9017. SPIE, 150–163.

[5] Carolina Cruz-Neira, Jason Leigh, Michael Papka, Craig Barnes, Steven M Cohen, Sumit Das, Roger Engelmann, Randy Hudson, Trina Roy, Lewis Siegel, et al. 1993. Scientists in wonderland: A report on visualization applications in the CAVE virtual reality environment. In *Proceedings of 1993 IEEE research properties in virtual reality symposium*. IEEE, 59–66.

[6] Carolina Cruz-Neira, Daniel J Sandin, Thomas A DeFanti, Robert V Kenyon, and John C Hart. 1992. The CAVE: audio visual experience automatic virtual environment. *Commun. ACM* 35, 6 (1992), 64–73.

[7] Ronan Gaugne, Quentin Petit, Jean-Baptiste Barreau, and Valérie Gouranton. 2019. Interactive and Immersive Tools for Point Clouds in Archaeology. In *ICAT-EGVE 2019-International Conference on Artificial Reality and Telexistence-Eurographics Symposium on Virtual Environments*. 1–8.

[8] Kenny Gruchalla. 2004. Immersive well-path editing: investigating the added value of immersion. In *IEEE Virtual Reality 2004*. IEEE, 157–164.

[9] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. 2017. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847* (2017).

[10] Rajiv Khadka, Nikhil Shetty, Eric T Whiting, and Amy Banic. 2016. Evaluation of collaborative actions to inform design of a remote interactive collaboration framework for immersive data visualizations. In *International symposium on visual computing*. Springer, 472–481.

[11] Boštjan Kovač and Borut Žalik. 2010. Visualization of LIDAR datasets using point-based rendering technique. *Computers & Geosciences* 36, 11 (2010), 1443–1450.

[12] Oliver Kreylos, Gerald W Bawden, and Louise H Kellogg. 2008. Immersive visualization and analysis of LiDAR data. In *International Symposium on Visual Computing*. Springer, 846–855.

[13] Oliver Kreylos and Louise H Kellogg. 2017. Immersive Visual Data Analysis For Geoscience Using Commodity VR Hardware. In *AGU Fall Meeting Abstracts*, Vol. 2017. IN32B–04.

[14] Oliver Kreylos, Michael Oskin, Eric Cowgill, Peter Gold, Austin Elliott, and Louise Kellogg. 2013. Point-based computing on scanned terrain with LidarViewer. *Geosphere* 9, 3 (2013), 546–556.

[15] Bireswar Laha, Kriti Sensharma, James D Schiffbauer, and Doug A Bowman. 2012. Effects of immersion on visual analysis of volume data. *IEEE transactions on visualization and computer graphics* 18, 4 (2012), 597–606.

[16] E Li, Shuaijun Wang, Chengyang Li, Dachuan Li, Xiangbin Wu, and Qi Hao. 2020. SUSTech POINTS: A Portable 3D Point Cloud Interactive Annotation Platform System. In *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1108–1115.

[17] Oscar Martinez-Rubi, Stefan Verhoeven, Maarten Van Meersbergen, Peter Van Oosterom, R GonÁalves, Theo Tijssen, et al. 2015. Taming the beast: Free and open-source massive point cloud web visualization. In *Capturing Reality Forum 2015, 23-25 November 2015, Salzburg, Austria*. The Servey Association.

[18] Ryan P McMahan, Doug A Bowman, David J Zielinski, and Rachael B Brady. 2012. Evaluating display fidelity and interaction fidelity in a virtual reality game. *IEEE transactions on visualization and computer graphics* 18, 4 (2012), 626–633.

[19] Riccardo Monica, Jacopo Aleotti, Michael Zillich, and Markus Vincze. 2017. Multi-label point cloud annotation by selection of sparse control points. In *2017 International Conference on 3D Vision (3DV)*. IEEE, 301–308.

[20] Yancheng Pan, Biao Gao, Jilin Mei, Sibo Geng, Chengkun Li, and Huijing Zhao. 2020. Semanticposs: A point cloud dataset with large quantity of dynamic instances. In *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 687–693.

[21] Abhishek Patil, Srikanth Malla, Haiming Gang, and Yi-Ting Chen. 2019. The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 9552–9557.

[22] Christopher Plachetka, Jens Rieken, and Markus Maurer. 2018. The TUBS Road User Dataset: A New LiDAR Dataset and its Application to CNN-based Road User Classification for Automated Vehicles. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2623–2630.

[23] Markus Schütz. 2015. *Potree: Rendering large point clouds in web browsers*. Ph. D. Dissertation. Wien.

[24] Markus Schutz, Katharina Krosl, and Michael Wimmer. 2019. Real-Time Continuous Level of Detail Rendering of Point Clouds. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE. https://doi.org/10.1109/vr.2019.8798284

[25] Markus Schütz, Gottfried Mandlburger, Johannes Otepka, and Michael Wimmer. 2020. Progressive real-time rendering of one billion points without hierarchical acceleration structures. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 51–64.

[26] William R Sherman, Gary L Kinsland, Christoph W Borst, Eric Whiting, Jurgen P Schulze, Philip Weber, Albert YM Lin, Aashish Chaudhary, Simon Su, and Daniel S Coming. 2014. 47 Immersive Visualization for the Geological Sciences. (2014).

[27] Ross Tredinnick, Markus Broecker, and Kevin Ponto. 2016. Progressive feedback point cloud rendering for virtual reality display. In *2016 IEEE Virtual Reality (VR)*. IEEE, 301–302.

[28] Manuel Veit and Antonio Capobianco. 2014. Go'Then'Tag: A 3-D point cloud annotation technique. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 193–194.

[29] Colin Ware, Kevin Arthur, and Kellogg S Booth. 1993. Fish tank virtual reality. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*. 37–42.

[30] Florian Wirth, Jannik Quehl, Jeffrey Ota, and Christoph Stiller. 2019. Pointatme: efficient 3d point cloud labeling in virtual reality. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1693–1698.

[31] Jimmy F. Zhang, Alex R. Paciorkowski, Paul A. Craig, and Feng Cui. 2019. BioVR: a platform for virtual reality assisted biological data integration and visualization. *BMC Bioinformatics* 20, 1 (feb 2019). https://doi.org/10.1186/s12859-019-2666-z

[32] SM Zolanvari, Susana Ruano, Aakanksha Rana, Alan Cummins, Rogerio Eduardo da Silva, Morteza Rahbar, and Aljosa Smolic. 2019. DublinCity: Annotated LiDAR point cloud and its applications. *arXiv preprint arXiv:1909.03613* (2019).