More Agile in 3D: Making 3D Annotation More Intuitive with VR

Emily Steiner, Codey Sun, Liyuan Zhu
Stanford University

Abstract

The manual annotation of 3D data poses a significant bottleneck for advancing 3D computer vision tasks, due to the complexity of 3D data structures, reliance on 2D interfaces for 3D visualization, and intricate scene geometries. This paper introduces an intuitive and immersive virtual reality (VR) application designed to enhance the efficiency and quality of 3D visual data annotation. By integrating a Meta Quest VR headset with AGILE3D, a state-of-the-art interactive multi-object 3D segmentation model, our system allows users to interact with 3D scenes directly in a VR environment, overcoming the limitations of traditional 2D graphical user interfaces (GUIs). We implement a client-server architecture that facilitates real-time communication between the VR client and the segmentation backend. Additionally, our system introduces novel scene editing capabilities, enabling users to detach and reposition segmented objects, which can significantly expedite data augmentation and scene customization. A user study comparing our VR simulation with a conventional 2D GUI indicates a strong preference for the VR interface, with improved intuition of interaction, enhanced accuracy for occluded objects, and high perceived value for post-segmentation editing. This work demonstrates the potential of VR to make 3D data annotation more agile, intuitive, and efficient.

1. Introduction

Data collection and user annotation remain a time-consuming and limiting bottleneck for many 3D computer vision tasks. Due to the lack of data diversity and high quality annotations, 3D applications such as 3D Visual Question Answering (VQA), 3D scene generation, and 3D scene understanding [4, 11, 6] lack the zero-shot capabilities–the ability to generalize to unseen tasks and classes – of their 2D foundation model counterparts [10, 8]. 3D scans, unlike photographs and videos, are not organically collected or captioned through people's natural interaction with the internet and must be manually collected, curated and annotated. This annotation process is intricate as 3D data structures are not standardized, the annotators must use 2D windows to visualize 3D structures, and scenes often have large, complex geometries and occlusions.

This project addresses these challenges by introducing an intuitive and immersive Virtual Reality (VR) application designed to significantly enhance 3D visual data annotation. We integrate a Meta Quest VR headset with AGILE3D [15], a state-of-the-art interactive multi-object 3D segmentation model. Our system enables users to directly interact with 3D scenes within a VR environment, thus overcoming the limitations of traditional 2D Graphical User Interfaces (GUIs) for 3D interaction. Beyond streamlining interactive annotation, our system introduces novel scene editing capabilities. Users can easily detach and reposition segmented objects within the 3D environment. This additional functionality will provide further data on changed 3D scenes without the need to revisit and recollect scans of the original scene. The editing process will also allow for personalizing a 3D space without manual effort, which has applications in architecture, design, and furnishings. Through this work, we demonstrate how VR can transform 3D data annotation into a more agile, intuitive, and efficient process, thereby contributing to the development of next-generation 3D computer vision applications.

2. Related Work

3D data annotation has long been recognized as a critical bottleneck for training foundation models for 3D perception and generation tasks [5, 14]. Datasets like ScanNet [5], Matterport3D [3], S3DIS [1], SemanticKITTI [2], ScanNet++ [14] have provided invaluable resources for tasks like 3D object detection, semantic segmentation, and instance segmentation, the process of generating these annotations remains laborious and error-prone. Annotators typically interact with 3D meshes or point clouds through 2D projection interfaces—clicking and dragging to delineate object boundaries on screen—which often leads to imprecise masks, particularly in regions with occlusions or sparse point density. Moreover, accurately labeling complex indoor scenes (as in ScanNet [5] or Matterport3D [3]) frequently requires switching between multiple views or even leveraging time-synchronized RGB-D streams, further increasing annotation time and cognitive load. Consequently,

the scarcity of high-quality, densely annotated 3D datasets continues to impede the development of robust foundation models capable of generalizing to new environments and downstream applications.

Open-vocabulary 3D Segmentation. Open-vocabulary semantic segmentation refers to the task of assigning meaningful labels to image or 3D scene regions using an unrestricted, user-defined set of textual categories without requiring dedicated training examples for each category. This new field utilizes the zero-shot capabilities of 2D vision-language models [10, 7]. OpenScene [9] distills 2D image features onto 3D point clouds, building a point cloud representation co-embedded with text and image pixels in CLIP space. OpenMask3D [13] extends open-vocabulary scene-understanding to instance-level by multi-view feature aggregation. Segment3D [6] proposes a novel approach for fine-grained class-agnostic 3D point cloud segmentation without manually annotated labels. Great progress has been seen in open-vocabulary 3D scene understanding, however, results from these methods are still far from perfect to serve as the ground truth training data, thus requiring further human annotation or refinements.

Interactive Segmentation. Recent works AGILE3D [15] and Easy3D [12] have explored the 3D interative segmentation for more efficient and high-quality 3D semantic annotation. Easy3D [12] proposes a simple voxel-based encoder with implicit click fusion, significantly improving generalization and efficiency across diverse datasets. AGILE3D [15], employs attention mechanisms to simultaneously segment multiple objects using spatial-temporal user clicks, achieving state-of-the-art performance with fewer interactions and faster inference. Both approaches substantially enhance segmentation accuracy and user interaction effectiveness. A downside of AGILE3D GUI is the interface shown in Figure 1, requiring an annotator to navigate a potentially complex 3D scene with a 2D window and keyboard navigation. In addition, the annotator must switch to fill in text prompt boxes to provide a semantic annotation for each segmentation, potentially disrupting the segmentation workflow.

3. Method

This section details the architecture and operational procedures of the interactive segmentation system, adapted to Virtual Reality (VR) headset control. Our method, outlined in Figure 2 consists of two main components: 1) the backend AGILE3D server and 2) the front-end Unity application. The project is tested on a Meta Quest device, as a VR controller is necessary for user interaction with the scene.

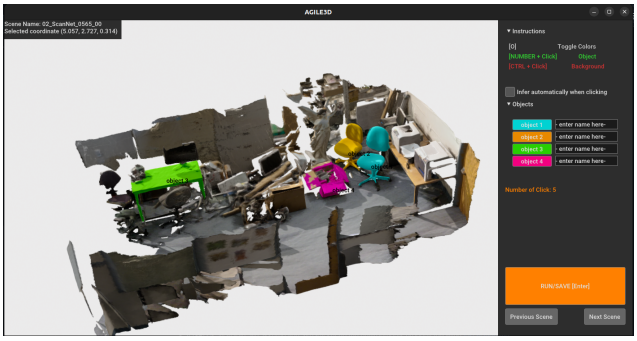

Figure 1. **2D Interface of AGILE3D.** The desktop interface allows annotators to toggle color schemes, label objects and background via numbered clicks, monitor the current click count, and run or save the refined segmentation, with controls for navigating between scenes.

3.1. Backend Model

The core of our system is an existing cutting-edge interactive 3D point cloud segmentation model, AGILE3D [15]. AGILE3D is an attention-guided interactive framework for multi-object segmentation in 3D point clouds. Figure 3 shows the original model architecture. Rather than segmenting objects one at a time, it encodes user clicks as spatial-temporal queries. It also allows positive and negative (background) clicks on any object to influence all other object masks jointly (click sharing and holistic reasoning). A lightweight transformer-based decoder refines these click queries and per-point features via an attention module, fuses per-click logits into region-specific predictions, and enforces global consistency so each point is assigned exactly one label. The predicted mask is then mapped back to the original point cloud resolution.

Upon initialization, it loads a pre-trained deep learning model capable of processing 3D point cloud data. By precomputing backbone features from a voxelized point cloud input and running only the decoder at each iteration, AGILE3D achieves fast inference. It reduces the number of user interactions required for high-quality, mutually exclusive instance segmentations. In the original AGILE3D implementation, a user can interact with the 3D scene through a provided Graphical User Interface (GUI). The system continuously updates and visualizes the segmentation results, records interaction metrics (e.g., mIoU if ground truth labels are available), and saves the generated masks and click data. Scene management is handled by methods which initialize a new scene by loading point cloud data, colors, and labels, quantizing the coordinates, and computing backbone features using MinkowskiEngine. This method allows for sequential loading of scenes, switching scenes with user commands, and ensuring a continuous annotation workflow. This project has extended this functionality by implement-

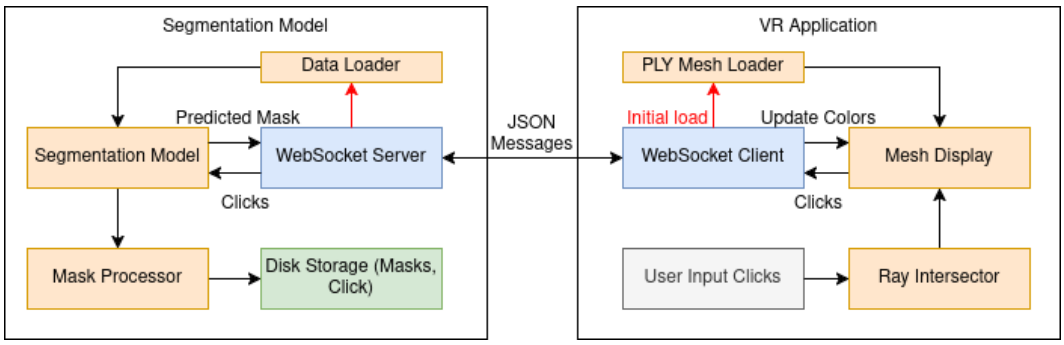

Figure 2. Conceptual overview of the system architecture. The interactive segmentation model processes point cloud data, with user input is received via a WebSocket server from a VR headset, replacing the local GUI.

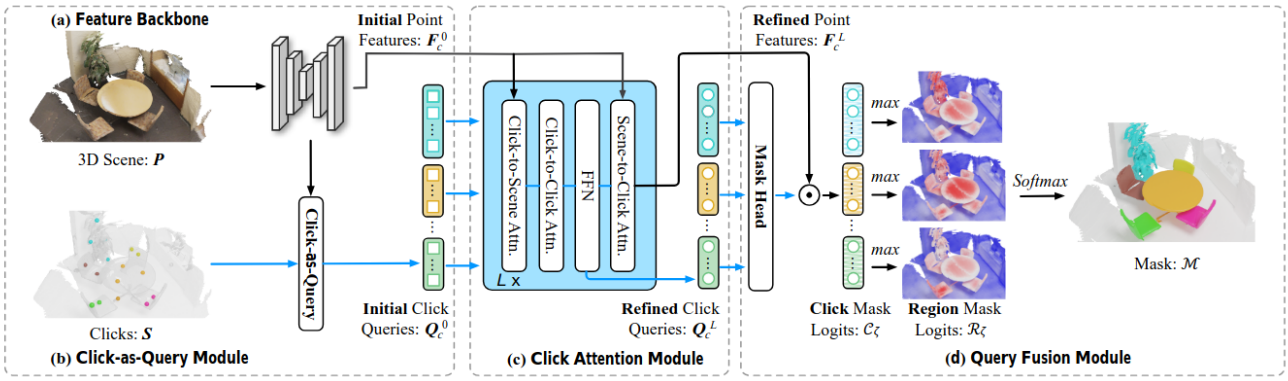

Figure 3. **AGILE3D Model Architecture[15]**. A light transformer network uses a 3D point cloud with precomputed features and user clicks to predict and iteratively refine the semantic instance segmentation.

ing a server-client communication layer to enable an external VR headset to serve as the primary interaction device. This modification allows a more immersive and intuitive user experience for annotating 3D data.

3.2. Server-Client Architecture

A back-end server hosts an instance of the AGILE3D model modified to accept point prompts via a WebSocket, which listens for connections from VR clients and manages the bidirectional communication. The *InteractiveSegmentationVR* acts as an intermediary, translating VR interactions into calls to the backend's segmentation model's methods and sending model outputs (updated masks and colors) back to the VR headset for rendering. This client-server architecture effectively separates the computationally intensive segmentation model from the immersive user interface, allowing for real-time interaction in a virtual reality (VR) environment. The design of this architecture considers the different requirements for implementation in various environments. The backend model operates on a Linux-based system, while the Meta Quest requires a Windows PC equipped with a GPU.

3.2.1 Server Initialization and Operation

The *InteractiveSegmentationVR* server is initialized with an instance of the *UserInteractiveSegmentationModel*, along with a specified host and port. It runs on a separate thread, managing a command queue for asynchronous communication between the segmentation model's logic and the VR client. Due to the large scale of the point cloud scene, the geometry and original colour information are preloaded onto the device connected to the VR headset rather than sent via WebSocket. As a result, when the server is started, initial scene information (e.g. name identifier) is sent to connected clients, which will locate the scene locally to ensure annotation is scene consistent.

3.2.2 Communication Protocol

The communication between the server and the VR client is JSON-based, with specific message types for different interactions described in Table 1.

Message Type	Description
`scene_loaded`	Confirms that the VR headset has loaded the requested scene geometry.
`click`	Sent from the VR headset to the server, containing the `point_index` of the clicked point (or its `position` as a fallback), `click_type` (e.g., "object" or "background"), and `object_id`.
`run_segmentation`	Triggered by the VR client to request a segmentation update based on current clicks.
`next_scene/previous_scene`	Commands to navigate through the dataset scenes.
`toggle_colors`	Toggles between the original point cloud and semantic segmentation colors.
`create_object/switch_object`	Manages creating new segmentation objects or switching between existing ones.
`send_scene_info`	Sent from the server to the VR client to inform about the current scene name and point count.
`send_mask`	Sends the computed segmentation mask to the VR client for visualization.
`send_colors`	Transmits updated point colors to the VR client.
`send_objects`	Informs the VR client about the available and current objects for segmentation.
`session_complete`	Indicates the end of the segmentation session.

Table 1. JSON-based message types for server-VR client communication

3.2.3 Click Processing and Model Interaction

Upon receiving a *click* message from the VR headset, the *InteractiveSegmentationVR* server processes the *point_index* to identify the corresponding point in the quantized coordinates. It maintains internal records of positive and negative clicks for each object, storing their index, position and time in dictionaries. If *auto_infer* is enabled, a segmentation run is automatically triggered after each click. The server then communicates the click feedback to the VR headset, allowing immediate visual response. The information is sent with quantized coordinates and chunked to account for Websocket message size limits. The *find_nearest_point* utility is used as a fallback to identify the closest point if the VR client does not provide an exact p̆oint_index.

3.2.4 Test Client

The communication protocol was verified by creating a simple test client. The client performs eight tests to confirm the the process of (1) server heartbeat, (2) requesting and loading consistent scenes, (3) updating chunked color information, (4) creating test objects from VR client, (5) performing an object and background click, (6) running the segmentation model, (7) toggling visualization colors, and (8) enabling and performing clicks with auto-inference.

3.3. Unity Application

The Unity application allows users to segment and edit scenes easily through a VR interface. A walkthrough of the scene editing process is shown in Fig. 5. The Unity application connects with the AGILE3D server to synchronize scene meshes, point prompts, and masks. After an initial handshake, the client reads the currently loaded scene mesh from the server. The user can traverse this scene naturally within VR. The Meta Quest controllers are used to further interact with the scene. A complete overview of the user controls is shown in Fig. 4. When the server processes the point prompts and predicts a mask, the application visualizes the masks by coloring the scene. A simple Phong shader is implemented to allow for basic stereo rendering of meshes with vertex colors.

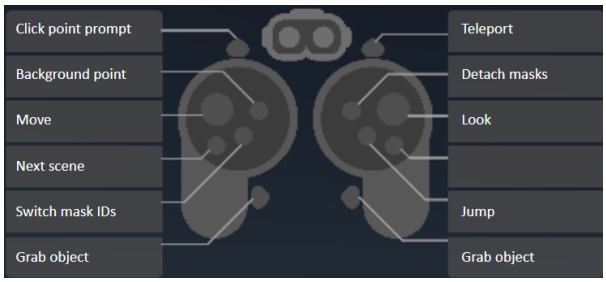

Figure 4. An overview of the Unity application user VR controls

3.3.1 Scene Processing

The Unity project facilitates the loading, manipulation, and interactive labeling of 3D mesh data, primarily in the PLY file format. *BinPLYMeshLoader* parses binary PLY files and converts them into Unity 'Mesh' objects. This includes processing PLY header information to extract the mesh properties (e.g., vertex count, face count, presence of color or normal data) and then efficiently reading the binary body to extract vertex positions and triangle indices. The loader ensures consistent orientation and positioning within the Unity scene by automatically centring rotating ($-90°$ rotation around the X-axis), and shifting the mesh's lowest point to $Y = 0$, to account for different standard coordinate systems between computer vision and VR fields.

For the user demo, we use ScanNet/ScanNet++ indoor

scene reconstructions [5, 14]. Users may also scan and import their scenes using a consumer smartphone reconstructed in PLY format. Despite being trained on ScanNet/ScanNet++ indoor scenes, AGILE3D shows strong generalization abilities when evaluated on S3DIS [1] and the outdoor KITTI dataset [2] with a significant domain shift [15] as a result supporting custom scenes.

3.3.2 Segmentation Interface

In addition to movement throughout the scene using the headset and the interactive segmentation, the backend model requires user input in the form of positive and negative vertex clicks, which are collected with the Meta Quest controller. The *RayInteractor* enables interactive vertex-level labeling of the loaded 3D models using an XR Ray Interactor, commonly found in Virtual Reality (VR) setups. When a raycast from the VR controller intersects a mesh, the system identifies the closest vertex to the hit point. The application visualizes a ray shooting from the left-hand controller for better user control and interaction. Upon pressing the left trigger, a point prompt is placed at the intersection of this ray and the scene mesh. The left controller is used as the primary annotation, and in particular, the left trigger was chosen as a natural location for the main positive click functionality. A small colored point will appear to visualize the selection. This point prompt is sent to the AGILE3D server to perform mask prediction.

The user can press the primary button to cycle between different object identities, visualized as different colors; additionally, by holding down the secondary button, point prompts will be sent as "background" click prompts to constrain over-segmentation. The Unity application manages a mask for each interactable object, storing the assigned label index for each vertex. This mask can be updated incrementally through messages from the backend server, allowing for dynamic color updates based on incoming data. The system also supports resetting colors to the original state and sending interaction events (vertex index and label) for external processing. Users can press the menu button to cycle between the available demo scenes. The system is fast and responsive, with sub-second latency between user input and response visualization.

For semantic annotation, the proposed interaction involves the user pressing the right menu button to enable dictation while using the left controller to assign the spoken semantic name to a mask. Due to time constraints, the Unity application currently does not include semantic annotation. Using voice dictation will streamline the annotation workflow, allowing users to continue annotating without having to pause to type in classifications. Additionally, this approach will support open-ended semantic labels, which aligns with recent efforts to diversify annotation for open-vocabulary scene understanding tasks.

3.3.3 Scene Modification Interface

This project not only adapts the original AGILE3D GUI for VR but also introduces functionality for scene modification. By enabling quick transitions to scene editing within the same application, we aim to accelerate the data annotation pipeline through efficient virtual data augmentation. Users will be able to rearrange objects within the same scene or add and remove items as needed. This implementation addresses the real-world demand for easy and rapid data augmentation. Additionally, it could serve as a user-friendly application, allowing individuals to virtually segment a scan of their personal space and modify, rearrange, or exchange furniture with minimal effort.

Once the user is satisfied with the masks, the user can press the right-hand secondary button to detach the segmentation masks. Each colored mask/object will become detached from the scene as its sub-mesh. The user can then grab and move objects as desired using the grip buttons, the standard for grabbing, moving, rotating, and resizing objects on the Meta Quest controller.

4. Evaluation and Results

This project's central hypothesis is that an immersive VR experience, rather than a 2D GUI, will improve the user experience of interactive segmentation. The user experience encompasses several factors, including ease, speed, quality, and comfort.

4.1. User Study

A user study evaluated the perceived user experience and potential benefits of immersive virtual reality (VR) for 3D annotation tasks. This study compared a simulated VR environment with a traditional 2D graphical user interface (GUI) for performing 3D segmentation tasks, utilizing the same AGILE3D backend. A total of 10 participants, who were familiar with VR and the requirements of 3D annotation, took part in the survey. The study focused on two primary interfaces for interacting with 3D scenes:

VR Simulator: Participants could interact with 3D scenes in a simulated VR environment presented through a custom Unity application on a desktop computer. This application was designed to mimic a full VR headset's navigation and interaction paradigms. However, the application could not be deployed on a Meta Quest due to hardware limitations, which constrained our user experience study. The purpose of this setup was to provide a controlled environment to assess core interaction concepts without requiring

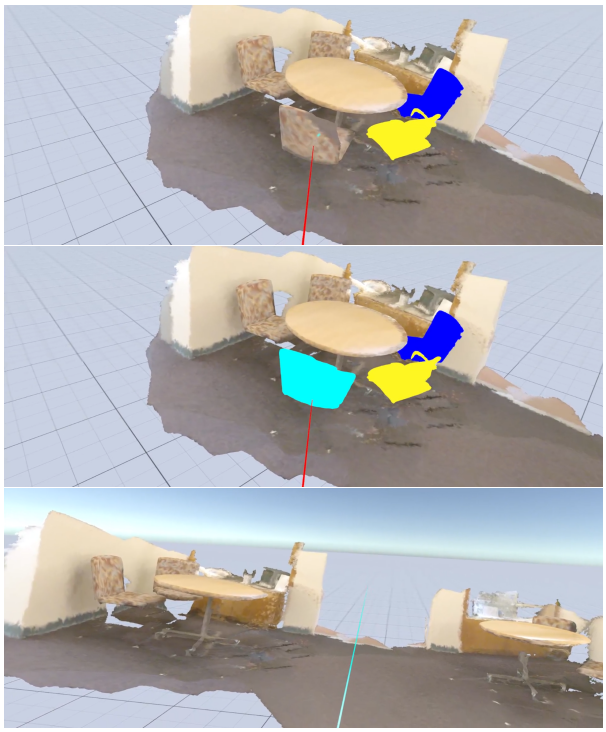

Figure 5. **VR-AGILE3D.** A walkthrough of the scene editing process on a ScanNet mesh. (Top) A point prompt appears at the ray intersection with the mesh. (Middle) Mask predictions of the prompt are visualized, with each color representing a unique object instance. (Bottom) The edited scene after the masked chairs are detached and then removed.

an extensive full VR headset, allowing us to collect feedback on potential benefits and user experiences.

2D GUI: For comparison, participants also performed tasks using the conventional 2D GUI of AGILE3D, shown in Figure 1, which employed standard mouse-and-keyboard controls for navigating and clicking.

4.1.1 Survey Design

The survey comprised several questions designed to capture feedback on the user experience, intuition, perceived accuracy, and potential utility. The survey questions included:

- **Preference for Full VR Headset:** Participants rated their likelihood of preferring a full VR headset for 3D segmentation tasks compared to the GUI on a screen, on a scale from 1 (Strongly Prefer GUI) to 5 (Strongly Prefer VR).
- **Intuition of Interaction:** Participants evaluated the perceived intuitiveness of interacting with 3D scenes and placing segmentation clicks in a real VR headset versus the GUI, on a scale from 1 (Much Harder in Full VR) to 5 (Much Easier in Full VR).
- **Accuracy for Occluded Objects:** Participants assessed the extent to which a full VR headset would improve their ability to accurately segment occluded objects (e.g., furniture partially hidden by other items) compared to using the GUI, on a scale from 1 (Significantly Worse in Full VR) to 5 (Significantly Better in Full VR).
- **Value of Post-Segmentation Editing:** Participants rated the perceived value of immediately editing the scene by relocating segmented objects (e.g., moving a segmented chair to a different position) in a full VR headset for tasks like data augmentation, architectural design, or scene prototyping, on a scale from 1 (Not at all Valuable) to 5 (Extremely Valuable).
- **VR Fatigue:** Participants were asked about their experience with VR fatigue when using a Meta Quest headset, with response options being *Yes*, *No*, or *Have not used a Meta Quest*.

The survey highlighted two case studies on potential benefits: annotation of challenging objects (e.g., partially occluded) and the additional benefit of scene editing for annotation or personalization within the same application. It also touched on the uncontrollable risk facing VR applications: VR fatigue. If users experience strong VR fatigue or sickness, the immersive experience will not offer any net benefit.

4.1.2 Results

The user study revealed a generally positive perception of VR for 3D annotation tasks. Participants preferred VR interaction and recognized its potential benefits for tasks like occluded object segmentation and scene editing. Participants rated the intuition of interaction (Mean=4.56) and the value of post-segmentation editing (Mean=4.56) as exceptionally high, suggesting these as substantial potential benefits of a VR interface. The perceived improvement in accuracy for occluded objects (Mean=4.44) was also highly regarded. Table 2 shows the mean and standard deviation of the survey questions; for all questions, larger numbers indicate preference and positive responses for the VR implementation.

The survey also found that most participants experienced VR fatigue when using the Meta Quest headset, as seen in Figure 6. This poses a critical concern for the feasibility of long-term VR-based annotation tasks.

The qualitative results are illustrated in Figure 5, which displays the outcomes of click masking along with the enhanced modification capabilities. The effectiveness of the

Table 2. Average Scores of Likert Scale Questions (1-5 Scale)

Question	Mean	Std. Dev.
Preference for Full VR Headset	3.78	1.09
Intuition of Interaction	4.56	0.73
Accuracy for Occluded Objects	4.44	0.70
Value of Post-Segmentation Editing	4.56	0.50

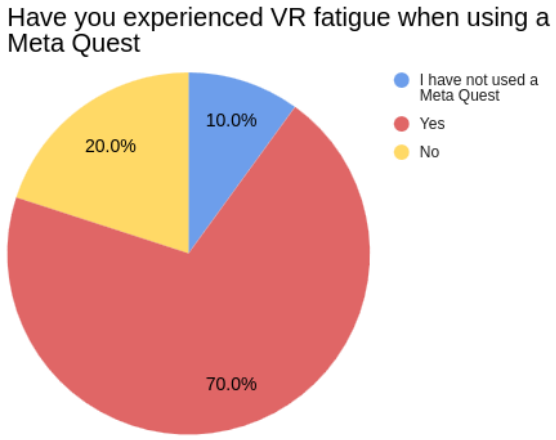

Figure 6. Distribution of Responses to VR Fatigue Question.

click masking remains consistent in the VR application compared to the 2D graphical user interface (GUI). Additionally, the lower figure demonstrates that simply crouching in VR allows users to achieve a new viewpoint more easily than adjusting with the keyboard. This new perspective reveals better vantage points for masking partially occluded objects, such as the legs of a table.

4.2. Runtime

An additional risk facing the VR application is that the WebSocket server will introduce a delay, compromising the interactive segmentation tool's real-time aspect. The latency was measured from when a user initiates a click to when the updated mesh is displayed. The original GUI has a latency of 0.13s, while the VR server has a latency of 0.2s. The 0.07 second (54% increase) latency is qualitatively negligible from the user's point of view.

4.3. Limitations & Future Work

The main limitation faced by this project was the lack of full VR user experience testing due to time constraints and the lack of a Windows desktop with a dedicated graphics card. This is a constrained evaluation of the user experience. Provided this hardware and more time, a complete user study to compare the experiences more thoroughly would be beneficial. This could include side-by-side comparison, including the resulting accuracy of the final masks, the time required for a whole scene on annotation, user perceived ease and comfort, and reports on whether VR fatigue was experienced using the VR application. The backend model's accuracy primarily limits the annotation tool's accuracy; however, some variation might occur based on user click control with each interface. With complete deployment, overall accuracy could be tested by getting users to annotate entire scenes with each method. In addition, the backend model may become bottlenecked by the model's generalization ability when tested on custom data. With complete testing, more iterations and feedback could be leveraged to determine each feature's most intuitive VR controller buttons.

Additional future work would include implementing voice dictation for semantic labelling of the segmented masks. This is a critical component of the annotation pipeline, and shifting from typed labels to hands-free annotation can potentially improve the workflow.

5. Conclusion

This project developed an immersive VR application for intuitive 3D data annotation and scene editing, addressing bottlenecks in 3D computer vision data collection. By integrating the Meta Quest VR headset with the AGILE3D segmentation model, we created a system that significantly enhances 3D annotation through natural VR interactions, improving accuracy and efficiency, especially for complex and occluded objects. A key contribution is the novel scene modification functionality, allowing users to detach and relocate segmented objects within VR. This streamlines diverse 3D dataset generation through virtual data augmentation and enables new applications in architectural design and virtual furnishings. Our user study in a simulated VR environment provided compelling evidence of the approach's benefits. Participants overwhelmingly preferred the VR interaction paradigm, citing improved interaction intuition, enhanced perceived accuracy for occluded objects, and high value for post-segmentation editing capabilities. While successfully demonstrating VR-based 3D annotation's feasibility and benefits, future work should focus on evaluating the fully deployed application on the Meta Quest headsets, incorporating voice dictation for semantic labeling, and exploring improvements in the user experience for segmentation and object manipulation. More Agile in 3D represents a step towards making 3D data annotation more accessible, intuitive, and agile, accelerating progress in 3D computer vision applications.

References

[1] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference*

on computer vision and pattern recognition, pages 1534–1543, 2016.

[2] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019.

[3] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

[4] B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Sadigh, L. Guibas, and F. Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14455–14465, June 2024.

[5] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[6] R. Huang, S. Peng, A. Takmaz, F. Tombari, M. Pollefeys, S. Song, G. Huang, and F. Engelmann. Segment3d: Learning fine-grained class-agnostic 3d segmentation without manual labels. *European Conference on Computer Vision (ECCV)*, 2024.

[7] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

[8] OpenAI. Gpt-4o system card. Technical report, OpenAI, San Francisco, CA, aug 2024.

[9] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–824, 2023.

[10] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[11] B. Roessle, N. Müller, L. Porzi, S. R. Bulò, P. Kontschieder, A. Dai, and M. Nießner. L3dg: Latent 3d gaussian diffusion. In *SIGGRAPH Asia 2024 Conference Papers*, December 2024.

[12] A. Simonelli, N. Müller, and P. Kontschieder. Easy3d: A simple yet effective method for 3d interactive segmentation. *arXiv preprint arXiv:2504.11024*, 2025.

[13] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[14] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[15] Y. Yue, S. Mahadevan, J. Schult, F. Engelmann, B. Leibe, K. Schindler, and T. Kontogianni. AGILE3D: Attention Guided Interactive Multi-object 3D Segmentation. In *International Conference on Learning Representations (ICLR)*, 2024.