MAVR: Multi-functional Point Cloud Annotations Using Virtual Reality

Xiao Zhang, Zhanhong Huang, and Xinming Huang

***Abstract*— Learning-based point cloud perception methods rely on labeled data for training data-driven models, necessitating the development of precise and efficient tools for point cloud annotations. In this paper, we propose MAVR, a multi-functional annotation framework based on virtual reality (VR) technology, capable of accurately labeling point cloud data for diverse applications, including part segmentation and object detection. We begin by evaluating the user interface (UI) efficiency through interactive efficiency analysis. Subsequently, a comprehensive three-step process is introduced, which consists of pre-processing, point selection, and post-tagging. For 3D object part segmentation and scene perception, we propose two distinct tagging pipelines. Our experimental results on various datasets validate the effectiveness of MAVR in accurately annotating point clouds from different data sources within an immersive workspace.**

I. Introduction

Point cloud is a unique data structure that provides a sparse representation of objects and their surroundings in 3D space. The accurate depth information in a point cloud can be integrated with intensity, norm, RGB, etc., providing new opportunities and challenges for research in 3D computer vision beyond the traditional image-based mechanism. Point clouds are generated from different sources. For instance, the stereo camera or laser fused camera can generate a dense point cloud with color. The LiDAR sensor can provide a sparse cloud with accurate depth at the centimeter level [1]. Besides the direct representation of objects, point cloud can also be used as a skeletal structure to support stereo models or as a mesh attached to rendering materials to bring vivid 3D visual effects.

In robotic and autonomous driving systems, point cloud serves as an important data source from sensors. Many recent works have been published to target high-level point cloud perception, such as semantic segmentation, panoptic segmentation, moving object segmentation, object detection, and object tracking. For object-oritented tasks, people use bounding boxes with tags to label objects in 3D space or in 2D bird-eye-view(BEV). In KITTI object detection task [2], perception system is required to identify cars, pedestrians, and cyclists in the scene. Point-oriented tasks require annotations of each point. For instance, semantic segmentation results in SemmanticKITTI [1] have point-by-point tags for 28 different classes. This raises a key question - how to correctly, accurately, and efficiently annotate labels in point cloud?

*This work was supported by the US NSF Grant 2006738 and by The MathWorks. The authors are with the Department of Electrical & Computer Engineering and Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA 01609, USA. {xzhang25,zhuang5,xhuang}@wpi.edu

The first role of point cloud annotation is to label the ground truth. The performance of a perception task always needs the ”true” labels to assess the model prediction accuracy. The metrics for systematic evaluation vary from task to task, but the need of labeled data is indispensable. On the other hand, machine learning based approaches have gained popularity in recent years, which calls for new labeling methods that can produce a large mount of labeled point cloud data efficiently and accurately.

Point cloud annotation can be divided into two categories: object segmentation, which separates objects from the scene, and in this problem, objects do not collide in 3D space. As the most challenging case, to track multiple objects in consecutive frames, multiple frame multiple object (MFMO) always suffers from complex environments and data noise. Single frame multiple object (SFMO) can be treated as the initial frame of MFMO, and multiple frame single object (MFSO) is the simple case of MFMO. The second category is part segmentation, single frame single object (SFSO) requires finer work to separate the components in an object where the most complicated problem is that some components are physically connected in space.

Traditional 2D image labeling is a time-consuming but intuitive work. The human label agents can instinctively recognize the objects in the image and label them using a keyboard. However, annotating the 3D point cloud is more challenging regarding object recognition, user interaction (UI) design, and sequence data adapting.

Object Recognition: Point cloud data is not as dense and information-rich as images. However, the sparsity also makes it hard to annotate. Generally, the point cloud view is not intuitive for human agents. Therefore, the recognition of objects in multi-object scenes is more complicated. In addition, the point loss problems caused by scanning occlusion will make labeling even more difficult.

User Interaction: Most UI designs are based on 2D displays with keyboard and mouse input. In 2D images, selecting objects by bounding the box requires only two actions: initializing the position and scaling the box. For 3D annotations, selecting objects with 6 degrees of freedom (DoF) is an uphill task and theoretically requires at most nine actions for each target: to place the bounding box along the X, Y, and Z axis and to adjust the rotation in pitch, yaw, and roll, as well as to adapt the scaling in all three directions.

Data sequence: For real-time perception, data is continuously collected in sequences rather than random frames. The correlation in adjacent frames must be considered during the labeling as a consecutive time series. For instance, object tracking requires the identification of each object with a

unique ID, which increases the effort and complexity of annotation.

To address these challenges. We proposed an immersive point cloud annotation tool based on a virtual reality (VR) headset, which can be applied to multiple tasks. The contributions of our work are listed as follows:

- **Immersive VR point cloud annotation design**: A point cloud annotation framework is proposed and implemented on a PC with an Oculus Quest 2 headset. The immersive virtual vision and mobile tracking headset provide users with a more intuitive and focused workflow. Data cleaning, view augmentation, and point stream play features are designed to help identify objects in complex environent.
- **Multi-functional labeling pipeline**: MAVR can handle annotations for both single-frame single-object (SFSO) and multi-frame multiple-data (MFMO) scenarios. The annotation output offers both bounding box and point-wise semantic labels.
- **Validation on practical datasets**: Two experiments are performed to demonstrate the capability of MAVR, including part segmentation and object detection annotation. The experiment result indicates our proposed immersive annotation system has better object recognition ability and can accurately place the bounding box in different scenarios.

II. Related Work

2D interface In early attempts, [3] designed a touchscreen device-based tagging tool. The fixed-size pre-select cube was set to reduce the number of candidates, making the tool only available for part selection with limited accuracy. For real-time data annotation, researchers developed semi-automatic tools to assist agents in tagging the LiDAR point cloud. LATTE [4], a sensor fusion-based on-click annotation and tracking system, avoids sophisticated 2D manipulation with the assistance of clustering and tracking algorithm, but the image R-CNN based pre-labeled process makes it hard to adopt the other datasets. A follow-up work [5] reinforced the denoising and tracking methods to elevate the accuracy. The semi-automatic strategies reduced workload, but the annotation performance is subjected to the clustering's adaptive capacity to different resolutions and object distances [6].

3D interface A 3D free-hand gesture-driven design [7] was built for part segmentation. The designer implemented a 3D mouse with a leap motion camera. Viewpoint-changing, selection of data points, and label creation were achieved through finger gesture collaboration. However, the multi-step selection only supported the segmentation by plane, which limited the utilization of tasks with simple object structure and loose accuracy constraints.

VR interface [8] proposed a point manipulation tool to select, tag, and move the target points by VR head-mounted device (HMD). The design has the advantage for the SFSO task because its hand-brush selector constrains the efficiency of point selection. [9] had a more efficient labeling method target on selecting the objects in complex LiDAR scenes. Size-scalable bounding boxes can cover the points of the things by referring to the image from the camera that shares the view field as LiDAR. However, since users adjusted the size manually, the output bounding boxes are not ideal for model training. In this case, the SFMO problem has been addressed with VR-based solutions, but the MFMO scenario is still challenging.

III. Interactive Efficiency Analysis

Virtual reality design has been studied to potentially have a better user experience by providing better visualization and intuitiveness in games and CAD designs. An immersive vision can help users concentrate on tasks, leveraging efficiency and performance. [10] indicated that users could achieve better results in VR than in a 2D monitor for the flaw detection task. [11] conducted a 3D structure design experiment that includes traditional UI and VR users. The investigation shows that VR users accomplish a design with a more realistic physical structure in a shorter time. Why does VR provide better interactive efficiency? In this section, we propose E_i to measure the efficiency of manipulation action in annotation design.

The dimension of manipulation DoM describes the number of manipulation dimensions that are required in the task or could be mapped in the interaction system:

$$DoM = \{T, R, S\} \in N \tag{1}$$

where, T, R, S represent the manipulation dimension in translation, rotation, and scale, respectively.

Given the two DoM, the target object DoM_t is related to the DoF of the annotated target according to the task assumption. DoM_h demonstrates the capability of manipulation handle, which is impacted by UI design and input device. We can calculate the maximum action number to select an object, which can be defined as:

$$\max Act = \sum^{dim} DoM_t \tag{2}$$

The maximum possible number of actions equals the sum of the target object DoM. Under this condition, box adjustment in every dimension is executed by a single operation.

The minimum action number is subject to the DoM_h. In the most aggressive design, the designer lets the user control all available handle input dimensions simultaneously without dimension mapping concerns. In this case, the least number of operations equals to:

$$\min Act = \lceil \frac{\sum^{dim} DoM_t}{\sum^{dim} DoM_h} \rceil \tag{3}$$

The interactive efficiency of an interaction design as:

$$E_i = \frac{Act}{\sum^{dim} DoM_t} = \frac{\sum^{step} Act}{\max Act} \tag{4}$$

E_i represents the average action needed to determine a single target object dimension. A higher E_i value means

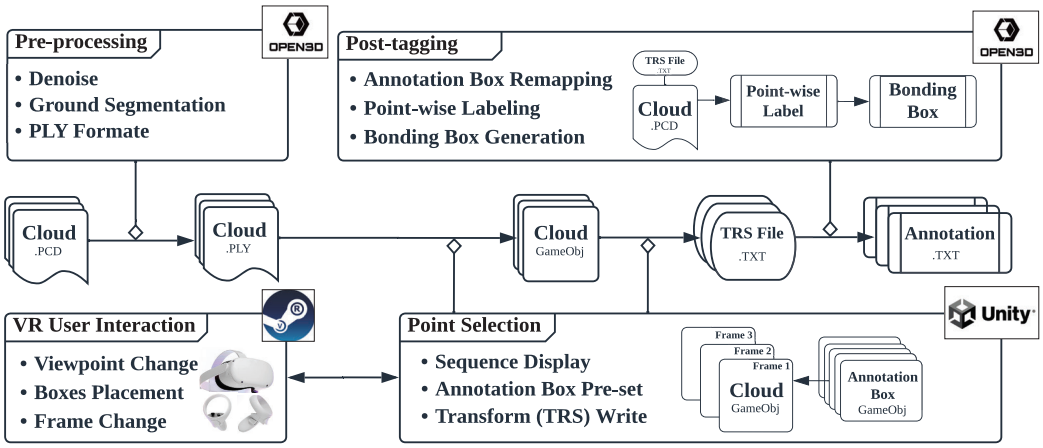

Fig. 1. MAVR Workflow

more actions are needed to determine a single dimension for the target object, and the corresponding UI and workflow step design are less efficient. Ideally, the design of the annotation workflow should approach a lower efficiency value while keeping an intuitive manipulation action to achieve the best interactive performance.

TABLE I

INTERACTIVE EFFICIENCY EXAMPLE

Handle { DoM_h } \Task {DoM_t}	Image $\{2,0,2\}$	Image-R $\{2,1,2\}$	3D-Cube $\{3,0,0\}$	Cloud-BEV $\{3,1,3\}$	3DCloud $\{3,3,3\}$
Mouse $\{2_t\}$	0.5 (2s)	0.6 (3s)	0.67 (2s)	0.71 (3s)	0.67 (3s)
Leap Motion Camera $\{3_t, 3_r\}$	**0.25** (s)	0.4 (2s)	**0.33** (s)	0.28 (2s)	**0.22** (2s)
VR Controller $\{3_t, 3_r, 2_k\}$	**0.25** (s)	**0.2** (s)	**0.33** (s)	**0.14** (s)	**0.22** (2s)

The subscript t indicates the current dimension input is completed by handle translation, r means it is completed by handle rotation, and k means it is triggered by a joystick handle to adjust the input.

Table I displays efficiency for various annotation tasks and UI setups. A single DoM handle can't map to multiple target dimensions simultaneously due to action complexity. The first two columns show that for tasks like 2D image detection, advanced UI devices don't always boost efficiency because of redundent DoMs. Devices like Leap Motion and VR controllers require at least one step for image labeling. Efficiency doesn't merely hinge on target DoM quantity. While placing a 3D cube seems simpler than image object selection based on x, y, and scale, the mouse excels in image labeling. In image tasks, mouse movements align closely with bounding box perspectives. In 3D tasks, setting the z position after x and y requires an additional step.

From an E_i standpoint, UI handles with more freedom often have superior efficiency. However, pricier devices with added DoMs aren't always crucial. For tasks like non-rotating image detection and 3D cube positioning, a mouse and keyboard suffice. For point cloud tagging, VR shines, particularly in BEV 3D LiDAR detection.

IV. THE ANNOTATION PROCESS

The pipeline of MAVR shows as Figure 1. The annotation procedure has three main steps. Pre-processing cleans the noise and ground from the raw dataset. In point selection, a sequence of point frames is loaded in Unity. For every frame, the VR annotation user utilizes the pre-set annotation box to select the points of the identified object and then writes the related transform attributes of boxes and raw point cloud to the TRS file. Finally, the post-tagging creates different types of labels by remapping the annotation box to the raw point cloud space.

A. Pre-processing

Pre-processing, including ground removal and denoise, is to increase the clarity of objects to the annotation users. Ground removal is often a necessary step. The idea for object detection or segmentation is to separate the objects from each other. While the ground is naturally connected to all the objects, it is always an obstacle for further processing. In this work, we implement the RANSAC plane fitting to remove the ground points if applicable.

Denoise is a practical method of data cleaning. The noise can prevent the annotation agent from figuring out the objects as well as selecting them. In the point cloud, the noise can be defined as an outlier with fewer neighbors. This paper utilizes the k-nearest neighbors (kNN) search-based denoise methods. The distance and neighbor number threshold depend on the point cloud data source and resolution.

In addition, the Unity point cloud data package only supports the PLY so far, so we need to reformat the point cloud data to PLY if it is not.

B. Point selection and interaction design

Data import and preset: Firstly, we use the PCX package to load the PLY point cloud files to the Unity scene. Then, the user can scale the point cloud to a fittable size for their work. From our experience, scaling down helps label the object points in the environment, and scaling up works better in part segmentation. The scale rate depends on the workspace size in the real world and the user preference. We can also preset the class and size of the annotation box (1-by-1-by-1 cube located at the origin) based on the assumption of the known target label classes.

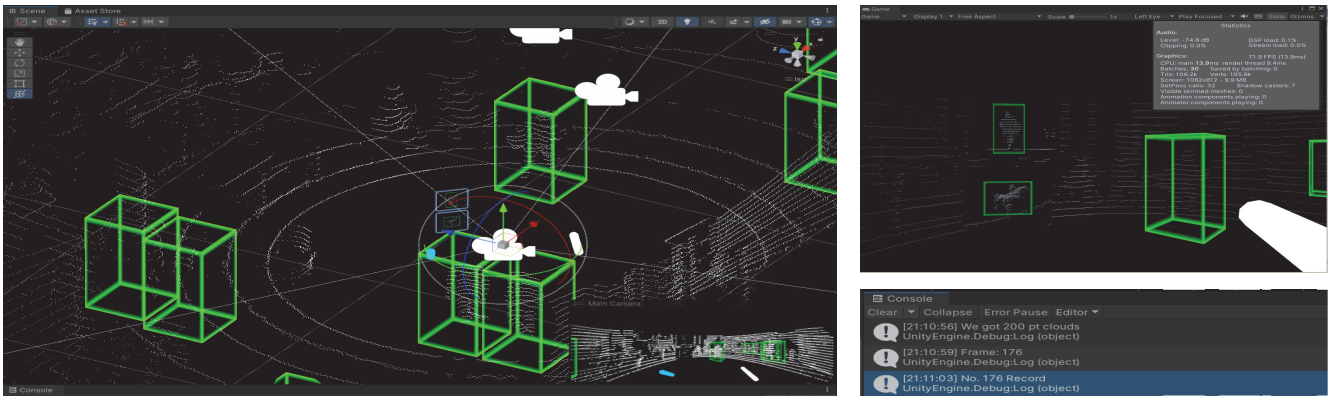

Fig. 2. MAVR User Interface

Left shows the workspace scene from Unity on Desktop. The highlighted part indicates the VR user. **Top Right** displays the VR user view. The handle emits white light means an annotation box is selected. **Bottom Right** Console on desktop can print the frame number, record, and other debug messages.

Viewpoint manipulation design significantly impacts the experience and performance of 3D point cloud UI. In traditional 2D design, users need to switch to viewpoint mode and adjust to the desired pose with several actions by the control handle. In this design, we follow the classic VR plan. The viewpoint movement is mapped to rotation and translation tracked by VR headset. It will benefit the system with a more intuitive interaction experience for human users. People can view the point cloud as the object is right in front. They can walk around and observe the point clusters from any angle they want in the immersive virtual environment.

View augmentation During the development, we found that the VR user can observe and find the objects very efficiently. However, locating the objects exactly is not as easy as we expected, even in the immersive platform. After people place the annotation box on the object's point cluster roughly, they need to walk around to fine-tune the position, aligning the border of the annotation box to the edge of the point cluster. To overcome the problem, we designed the projection camera sets to help people adjust the accurate position. The camera sets display the top and side view of the bounding box and is attached to the side of the main camera vision. As shown in Figure 2 top right, when confirming the point cluster is inside the annotation box in all views, the object points selection is finished.

Controller mapping Figure 3 presents the Oculus Quest 2 controller we utilize. The controller has original DoM $\{3_t, 3_r\}$ with four buttons/triggers and one joystick. A good design never puts up all available resources as the first goal but tries to think as a user to make work more straightforward. In our design, we map the scaling and placing functions to the two controllers separately to clarify the division of work. For the placing controller (right), the user can drag and select the boxes by holding the front trigger (Figure 2 shows that when a box is selected, the controller will flash white). The side trigger holds the box delete function. Transforming and rotating the controller directly maps to the annotation box when selected. The joystick is set to slide the box on the BEV plane for a more stable and tiny adjustment. The scale controller (left) takes charge of box scaling in three space directions. The frame changing can also be completed by left triggers. Finally, when work on the current frame is completed, the user presses Button B or Button A to record the RTS information of both the point cloud and boxes for post-tagging.

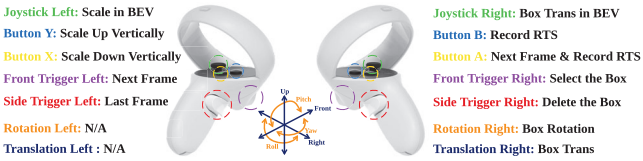

Fig. 3. VR Controller Mapping

Frame change and TRS file When the frame changes, the boxes of the previous frame will remain to reduce the placement work. Meanwhile, the TRS file for the frame is recorded. The file collects the class, unique object ID, translation, rotation, and scale related to the global X, Y, and Z reference. Each game object in the workspace has TRS values that determine its global position, pose, and size.

C. Post-tagging

From previous work completed by Unity, we know the point cluster of the object is within the annotation box. In the post-tagging step, we make the actual annotation by RTS file. More information can be referred to in Alg. 1.

Algorithm 1 post-tagging

Input : Original Point cloud. P
The TF and scale information. $TRS_P(1, 11)$
The TF and scale information of n boxes. $TRS_A(n, 11)$
Merging the annotation boxes belonging to the same class. $merge\ Boolean$
Output: Labeled point cloud. P
Bounding box vertices. $bBox$

$aBox \leftarrow unitBox()$
$bBox \leftarrow \{1, n\}$
$M_P = matGen(TRS_,[3:11])$
$P_t = pctransform(P, M_P)$
for $i = 1 : n$ **do**
  $M_{aBox} = matGen(TRS_A(i, [3:11]))$
  $aBox_T = pctransform(abox, M_{aBox})$
  $idx = checkBox(P_T, aBox_T)$
  $bBox(i) = boxGen(P, idx)$
  $class = TRS_A(i, 1)$
  $ID = TRS_A(i, 2)$
  if $merge$ **then**
    $label = class$
  else
    $label = ID$
  end
  $labelUpdate(P, idx, label)$
end
return *P, bBox*

First, the annotation box is initialized as a unit box (a unit cube centered at the origin). The rotation matrix M is obtained from $TRS\{1_{class}, 1_{ID}, 3_{trans}, 3_{rot}, 3_{scale}\}$ computed by $matGen()$. Then the point cloud and the box are remapped back to the annotation space by $pctransform$. After extracting the point indexes from the annotation box by $checkBox$, we can use $boxGen$ to generate bounding boxes according to the requirements and label the point clouds according to the merging strategy.

The different tagging strategies of object partitioning and segmentation are realized by label merge and priority update. Segmentation tasks have no box collision and class merge, the object can be labeled by the unique box ID directly. But object partitioning often needs a merge. As the chair has four supports, four boxes of supports need to be labeled as the same class. (Semantic object segmentation can be considered the same case) Since boxes have difficulty handling 3D shapes with curves, overlapping may occur at the connecting surface to avoid label missing. When repeat labeling occurs, strategically, the boxes with a smaller shape have priority.

V. Experiment

A. Setup

Experimental design. To evaluate the capability of the MAVR annotation tool from multiple perspectives, two experiments based on different data sources were conducted to verify the part labeling accuracy in SFSO [12] and the MFMO multi-target and the tracking labeling capability without image reference [13].

Device Setup Oculus Quest 2 VR set and PC with Nvidia 2080Ti GPU + Intel 12700k CPU.

B. Objects Segmentation in LiDAR (MFMO)

In [13], the researchers built an indoor LiDAR dataset for person detection. They collected and annotated the dataset manually and used it to train machine learning models. This collection is one of the few published individual datasets with object-level labels. Based on their LiDAR data only, we re-perform the people tagging to validate the MFMO annotation ability.

TABLE II
AVERAGE ANNOTATION STATISTICAL RESULT FOR ONE FRAME

	People	Group*	Total
L-CAS	1.7	1.2	4.3
Ours	7.5	-	7.5

*Group is estimated to contain 2.2 people in the 500 frames experiment

Table. II presents the statistical result. The headcount shows that MAVR can annotate around 3.2 more people on average per frame than the ground truth given by L-CAS. Meanwhile, it does not mix multiple people into the group.

As shown in Figure 4 left, the performance of L-CAS's annotation is limited by the candidates provided by the clustering. Object missing and under-segment issues frequently occur, seriously affecting the annotation and the following detection model. The figure from the right shows the annotation result provided by MARV. Our labels avoid the objects merging and missing problems. The object ID can be easily tracked as the tool keeps the annotation boxes from the pre-frame to help user labeling.

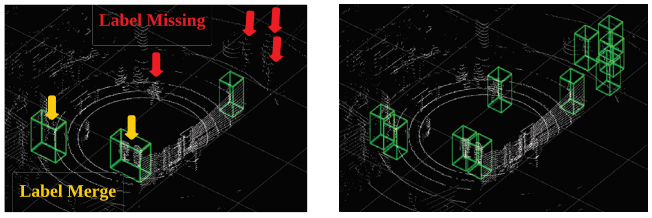

Fig. 4. MFMO Annotation Result
Left figure shows the "ground truth" provided by the L-CAS dataset, the **right** figure shows our annotation result

In the experiment, we found that people at the edges are hard to detect because the sparsity of LiDAR points increases as the distance increases. L-CAS implemented a clustering-assisted approach, letting people manually select candidates from a pool of clusters. However, because it is difficult for the clustering algorithm to adapt to various sparsity and object distances, the accuracy of the candidate pool can not be guaranteed. MAVR framework can solve this problem better: it is difficult for the human eye to distinguish the objects with several points at the edges of a frame, but by inheriting the boxes from the preceding frames, the tracking method significantly reduces the object missing issue.

C. Part Segmentation of 3D Shape (SFSO)

The data with partial segmentation labels are published as a subset of ShapeNet [12]. The point cloud of 3D shapes like guitars, lamps, and chairs is tagged with two to six components. In this experiment, we labeled several typical objects point by point and compared the annotation

result with ground truth to verify the label accuracy of our framework.

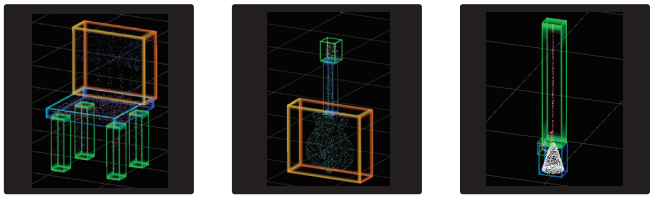

Fig. 5. SFSO point selection
Figures above show the box placements to segment the parts of chair, guitar and lamp.

Figure. 5 shows the result of box placements during point selection. The different colored bounding boxes represent different partial categories on one type of object. The color priority is set to green being the highest, blue the second, and orange the third. When points are repeatedly labeled, higher-priority boxes are set. It is aimed at selecting more precise structures for the best labeling result.

TABLE III
IOU OF PART SEGMENTATION FOR FIVE CLASSIC CLASSES

	Chair	Knife	Pistol	Guitar	Lamp
Ours	94.4	97.7	94.3	95.4	95.2

The experiment is completed by the same annotation agent and each category contains 50 samples.

As shown in the Table. III, the overall performance passes 90 in the selected categories, which shows the ability of proposed tools can achieve a high annotation accuracy in such a fine-grained partial segmentation task.

D. Discussion

From the above sections, we prove the MAVR's annotation accuracy and capability to resolve the lack of reference data. During the part segmentation, we suppose the design can provide a higher level of accuracy, but two obstacles prevent us from achieving it. One is that the ground truth of ShapeNet has noise: some outlier points exist on the component surface. Then, our annotation box only supports cubes. Though we can scale the size, the curve shapes are still tough to handle in some cases. In future work, supporting more annotation boxes (such as spheres) can be one solution. Through the MFMO case, we found that the first frame annotation will be the most time-consuming. After the initialization, tracking the objects one by one can be the most efficient pattern for human agents.

VI. Conclusion and Future Work

This paper proposes MAVR, a novel multi-functional point cloud annotation virtual reality framework. The pre-processing helps the user clean up the point cloud outliers for better observation. Through the Steam VR + Unity workflow, users can intuitively place the annotation boxes on the objects or parts within several steps. Single object annotation can even be finished in one click in the sequential frames after initialization. The post-tagging provides both bounding boxes and point-wise labels for various annotation jobs. Finally, according to the experiments, the three stages pipeline can handle the point cloud annotation tasks that vary from SFSO to MFMO accurately and has the capability to take the challenging raw data without extra reference.

This work aims to prove the potential and strength of a VR-based annotation framework in dealing with various point cloud tasks accurately and efficiently in 3D interactions. We understand that manual work is still hard to handle a significant amount of data required to feed into training the deep learning model. In this case, the tracking techniques implemented in [4] [5] are helpful in reducing labor work. Our future work will focus on the automation for labeling objects in adjacent frames, and manual work is only needed in the fine-tuning phase to increase accuracy. In that case, the structure will benefit from both an immersive manual and auto-tracking mechanism.

References

[1] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, and C. Stachniss, "Towards 3d lidar-based semantic scene understanding of 3d point cloud sequences: The semantickitti dataset," *The International Journal of Robotics Research*, p. 02783649211006735, 2021.

[2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[3] M. Veit and A. Capobianco, "Go'then'tag: A 3-d point cloud annotation technique," in *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 2014, pp. 193–194.

[4] B. Wang, V. Wu, B. Wu, and K. Keutzer, "Latte: accelerating lidar point cloud annotation via sensor fusion, one-click annotation, and tracking," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 265–272.

[5] H. A. Arief, M. Arief, G. Zhang, Z. Liu, M. Bhat, U. G. Indahl, H. Tveite, and D. Zhao, "Sane: smart annotation and evaluation tools for point cloud data," *IEEE Access*, vol. 8, pp. 131 848–131 858, 2020.

[6] Y. Zhao, X. Zhang, and X. Huang, "A technical survey and evaluation of traditional point cloud clustering methods for lidar panoptic segmentation," *arXiv preprint arXiv:2108.09522*, 2021.

[7] F. Bacim, M. Nabiyouni, and D. A. Bowman, "Slice-n-swipe: A free-hand gesture user interface for 3d point cloud annotation," in *2014 IEEE Symposium on 3D User Interfaces (3DUI)*, 2014, pp. 185–186.

[8] D. Garrido, R. Rodrigues, A. Augusto Sousa, J. Jacob, and D. Castro Silva, "Point cloud interaction and manipulation in virtual reality," in *2021 5th International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, ser. AIVR 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 15–20. [Online]. Available: https://doi.org/10.1145/3480433.3480437

[9] F. Wirth, J. Quehl, J. Ota, and C. Stiller, "Pointatme: efficient 3d point cloud labeling in virtual reality," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1693–1698.

[10] J. Wolfartsberger, "Analyzing the potential of virtual reality for engineering design review," *Automation in Construction*, vol. 104, pp. 27–37, 2019.

[11] S. M. Feeman, L. B. Wright, and J. L. Salmon, "Exploration and evaluation of cad modeling in virtual reality," *Computer-Aided Design and Applications*, vol. 15, no. 6, pp. 892–904, 2018. [Online]. Available: https://doi.org/10.1080/16864360.2018.1462570

[12] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[13] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for 3d lidar-based human detection: experimental analysis of point cloud clustering and classification methods," *Auton. Robots*, vol. 44, pp. 147–164, 2020.