# SMVLift: Lifting Semantic Segmentation to 3D on XR Devices

Marcus Valtonen Örnhag[a], Püren Güler[b],
Anastasia Grebenyuk[c], Hiba Alqaysi[d] and Tobias Widmark
*Ericsson Research, Lund, Sweden*

Keywords: Semantic Segmentation, XR Devices.

Abstract: Creating an immersive mixed-reality experience, where virtual objects are seamlessly blending into physical environments, requires a careful integration of 3D environmental understanding with the underlying contextual semantics. State-of-the-art methods in this field often rely on large and dense 3D point clouds, which are not feasible for real-time performance in standalone XR headsets. We introduce Sparse Multi-View Lifting (SMVLift), a lightweight 3D instance segmentation method capable of running on constrained hardware, which demonstrates on par or superior performance compared to a state-of-the-art method while being significantly less computationally demanding. Lastly, we use the framework in downstream XR applications with satisfactory performance on real hardware.

## 1 INTRODUCTION

High demand for processing power, especially for tasks involving 3D computations such as dense reconstructions, creates a bottleneck for most mobile devices (Wu et al., 2020), which typically lack the necessary computational resources. This limitation is especially pronounced in resource-constrained devices like XR headsets, where traditional 3D computational methods are not feasible for real-time, on-device processing. These devices require more efficient, lightweight solutions that can achieve similar results without the overhead of heavy processing loads.

To overcome these challenges, we propose a novel approach that leverages sparse 3D data combined with 2D image-based semantic segmentation, significantly reducing the computational burden. Our solution processes 2D images for semantic segmentation and then "lifts" the extracted semantic information from multiple views into a sparse 3D point cloud, utilizing a new lightweight component we developed called *SMVLift*. This approach not only alleviates the computational demands but also retains the precision needed for XR systems, offering a more efficient and scalable solution for object detection and interaction in augmented environments. Our main contributions are threefold:

- A fast incremental algorithm (SMVLift) for lifting multi-view 2D semantics to 3D on constrained hardware,
- A thorough comparison with the state-of-the-art learning-based method,
- We showcase the applicability of the proposed method on real XR hardware for downstream tasks by placing a virtual object in a physical scene.

## 2 RELATED WORK

### 2.1 2D to 3D Semantic Segmentation

By leveraging existing advancements in 2D image analysis, these methods offer a computationally lighter alternative for segmenting 3D point clouds, and as such are well-suited for resource constrained devices.

Wang et al. (2019) introduce the Label Diffusion Lidar Segmentation (LDLS) technique, which tackles the challenge of 3D object segmentation in lidar point clouds by using the semantic segmentation results of 2D images. LDLS employs a semi-supervised learning framework, creating a graph that bridges the 2D

[a] https://orcid.org/0000-0001-8687-227X
[b] https://orcid.org/0000-0001-6254-5135
[c] https://orcid.org/0009-0002-6503-9494
[d] https://orcid.org/0000-0001-9319-1413



Örnhag, M. V., Güler, P., Grebenyuk, A., Alqaysi, H. and Widmark, T.
SMVLift: Lifting Semantic Segmentation to 3D on XR Devices.
DOI: 10.5220/0012982800003905

In *Proceedings of the 14th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2025)*, pages 555-562
ISBN: 978-989-758-730-6; ISSN: 2184-4313

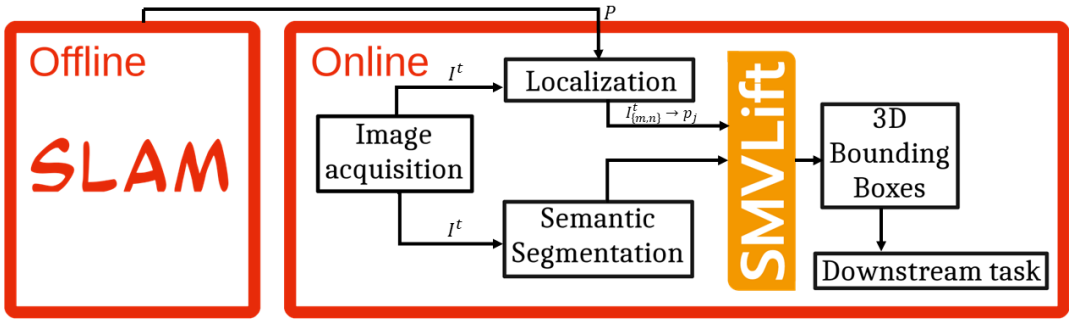


Figure 1: Illustration of general workflow. From an offline module, a map is calculated and 3D point cloud $P$ is given to the online module to extract 3D instances from the map. The online module starts with image acquisition $I^t$ at time $t$. Then "Localization" module extracts the 2D-3D correspondences $I^t_{m,n} \rightarrow p_j$, where $I^t_{m,n} \in I_t$ is a pixel in image $I_t$ at coordinates $m \in \{0,1,\ldots,W-1\}$ and $n \in \{0,1,\ldots,H-1\}$, and $p_j \in P$ is a corresponding point in a set $P$. In parallel, "Semantic Segmentation" module calculates 2D semantics for each image $I^t$ as object class $c_t$, object mask $m_t$, and class label score $s_t$. Using 2D-3D projections and 2D semantics, "SMVLift" module lifts 2D semantics to 3D instance semantics using a novel set-based instance assignment algorithm. Finally, we calculate 3D bounding boxes to be used by downstream tasks.

pixel space with the 3D point space. This graph facilitates the diffusion of labels from the 2D image segmented using Mask R-CNN (He et al., 2017) to the unlabeled 3D point cloud. The nodes of the graph consist of 2D image pixels and 3D lidar points, while the edges connect corresponding pixels and points based on their projections and 3D proximity. The initial labels from the 2D segmentation are diffused through this graph to the 3D points, utilizing the geometric relationships encoded in the graph to refine the segmentation boundaries. The process iterates until convergence, resulting in a finely segmented 3D point cloud.

Some authors fuse semantic information from multiple views, e.g., running semantic segmentation on images (RGB or RGB-D) captured from known poses and making use of multi-view geometric relationships to project the predicted semantic labels into the 3D space. Mascaro et al. (2021) presented a multi-view fusion framework for semantic scene segmentation. The framework addresses the 3D semantic segmentation challenge by using 2D semantic segmentation of multiple image views to produce a consistent and refined 3D segmentation. This approach formulates the 3D segmentation task as a label diffusion problem on a graph, leveraging multi-view data and 3D geometric properties to propagate semantic labels from 2D image space to the 3D map.

Wang et al. (2018) introduced PointSeg which uses spherical images derived from 3D lidar point clouds. The 2D spherical images are fed to a CNN that predicts point-wise semantic masks. The predicted masks are then transformed back to 3D space.

## 2.2 3D Instance Segmentation

In this category, the detection and segmentation are made directly on large and dense 3D point clouds. The semantic predictions are usually done point-wise using a deep neural network, followed by clustering the points into object instances. Due to the large size and density of the 3D point clouds required for these methods to perform well, they should be considered infeasible for device implementation.

Current state-of-the-art methods in 3D semantic segmentation mostly rely on neural networks and conduct the detection and segmentation directly on the map, i.e., these methods do not propagate class labels from the 2D images to the 3D map. Typically, pixel-wise semantic feature extraction is aggregated on extremely dense 3D point clouds and uses computationally expensive models to regularize the resulting 3D segmentation. Although these learning-based methods generally achieve better results, they require labeled 3D data for training and do not scale to XR devices.

Existing 3D instance segmentation methods, primarily bottom-up and cluster-based, struggle with closely packed objects or loosely connected large objects. To overcome these issues, ISBNet (Ngo et al., 2023) proposed a cluster-free method for 3D instance segmentation using Instance-aware Farthest Point Sampling to sample candidates and leverage the local aggregation layer to encode candidate features and box-aware dynamic convolution.

To reduce memory usage and inference time of segmenting 3D point clouds, Zhang et al. (2020) proposed to reduce the number of input points before feeding them to the segmentation model.

### 2.3 Spatial Computing on XR Devices

XR devices leverage spatial perception to support various tasks; however, the limited computational power available on such devices makes it a challenging task to realize. Many have considered offloading computationally expensive tasks to remote edge devices or cloud services, e.g., GPU servers, allowing for real-time processing (Heo et al., 2023). In Wu et al. (2020), the Microsoft HoloLens is used to recognize objects in an environment with object detection performed on 2D images. This data is subsequently mapped to a reconstructed 3D space.

In Hau et al. (2022), the authors proposed a method for 3D semantic mapping using a network of smart edge sensors for object pose estimation and refinement. The proposed method distributes a multi-view 3D semantic mapping system over multiple smart edge sensors to include object-level information for downstream tasks. According to their experiments, semantic tasks such as object detection and tracking can be executed at an update rate of 1 Hz while object pose estimation and refinement are performed online and in real-time. They conclude that the overall latency in such applications is highly dependent on the amount of processing needed to display the result.

In challenging network conditions, it is not feasible to rely on offloading and certain spatial perception tasks should be performed on device in order to guarantee a satisfactory user experience. However, it is not possible to run state-of-the-art 3D semantic tasks on device due to hardware limitations. This requires a different data flow to be realized in practice, e.g., by working with significantly sparser 3D data.

## 3 SYSTEM OVERVIEW

We propose to use object detection on multiple 2D images and lift these to a sparse point cloud. In our framework, we assume that the 3D reconstruction of the scene is computed offline, e.g., using available SLAM frameworks, and can be downloaded from the cloud or edge to the XR device. In real-time, images are captured on device and are parsed by a semantic segmentation algorithm in parallel to a localization framework. These are later merged in our proposed component, called Sparse Multi-View Lifting (SMVLift), and from the semantically enhanced point cloud we extract 3D bounding boxes, see Figure 1.

Assume a 3D map of an environment represented by a sparse point cloud $P \in \mathbb{R}^{3\times N}$ created through image views $I = \{I_t\}$ where $t \in \{0,...,T\}$. In the online module, we run multi-view object detection in 2D images $I$ and map these tentative labels to the corresponding points from the obtained 2D-3D correspondences using the localization module of the pipeline, see Figure 2. Then through a novel set-based instance assignment algorithm, we cluster the 3D points based on the semantic content of instances. After post-processing of instances to filter out noisy data, the final 3D bounding boxes can be extracted.


Figure 2: 2D-3D correspondences between images (with instance semantic masks visible) and point cloud from the *Lounge 2* of the S3DIS dataset. We will use a heavily downsampled point cloud, only keeping 0.3 % of the points, in order for us to work on XR devices.

The current framework is assuming a static scene; however, for future applications, this can be extended to dynamic environments.

### 3.1 Aggregating Multi-View Semantic Masks to Sparse 3D Points

An image $I^t \in \mathbb{R}^{W\times H}$ at time $t$ is fed to a 2D instance segmentation method which outputs class predictions $c(I^t_{m,n}) \in \mathbb{N}^N$, mask confidences $m(I^t_{m,n}) \in \mathbb{R}^{N\times W\times H}$ and label scores $s(I^t_{m,n}) \in \mathbb{R}^N$ for each image pixel $I^t_{m,n}$ at coordinates $m \in \{0,1,\ldots,W-1\}$ and $n \in \{0,1,\ldots,H-1\}$, assuming $N$ objects are found. Given an image $I^t$, there is a set of 2D-3D point correspondences, and for each detected object the 3D point indices belonging to the view are collected together with the label as a *local instance* $L^t_i$ for $i \in \{0,\ldots,N-1\}$ as a part of a preprocessing step. This can efficiently be implemented on GPU.

### 3.2 Set-Based Instance Assignment

Without applying tracking to semantic segmentation masks, the order in which the masks are labeled will be different between views, e.g., an object assigned instance number $M$ in one view will be assigned instance $N$ in another, and there is no clear way of mapping the two together. The primary reason to not use tracking is that we want to aggregate views from multiple locations of a room, without having to track im-

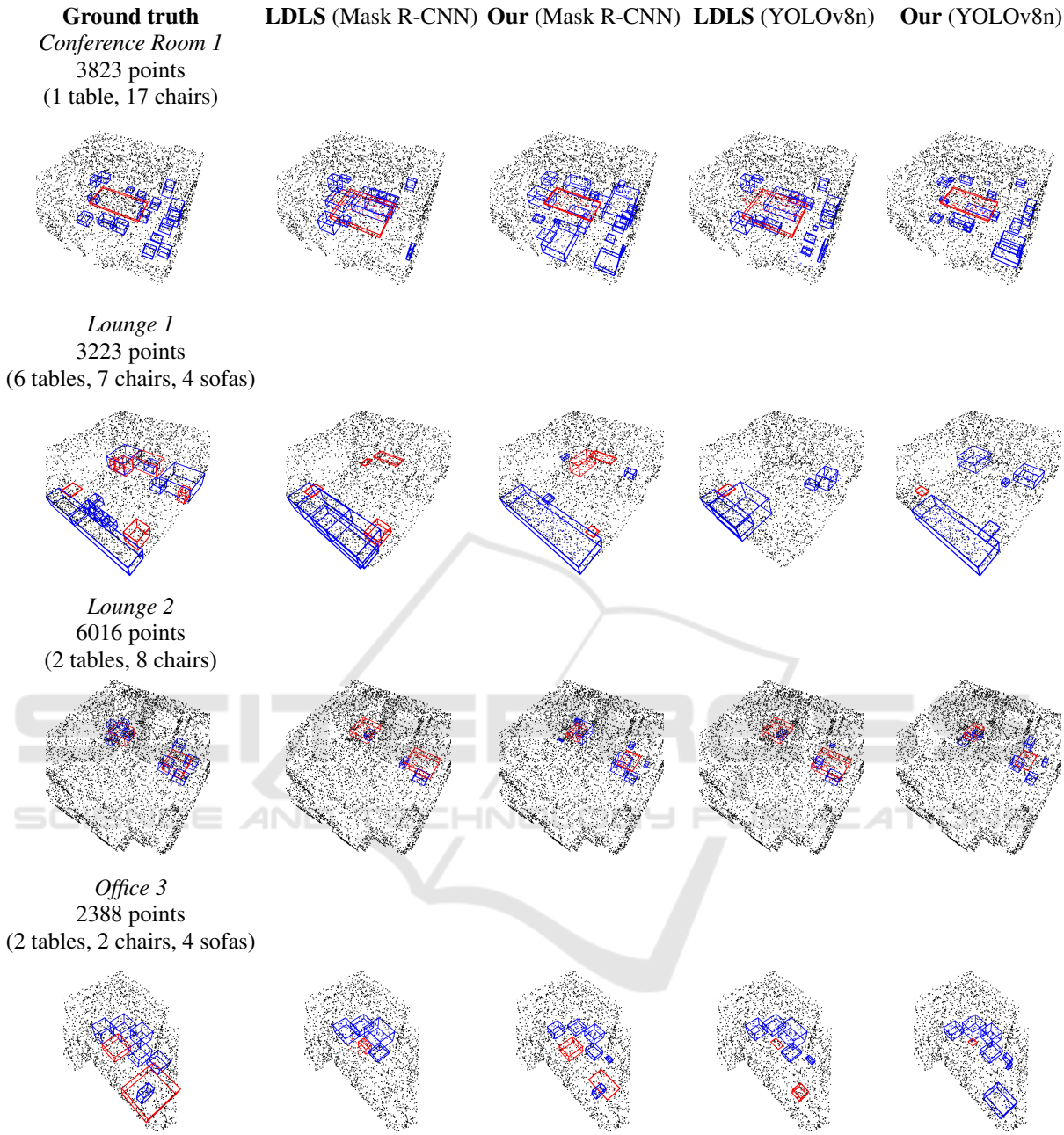


Figure 3: Experiment on a heavily downsampled point clouds of S3DIS. Bounding boxes for the two different classes are colored as follows: tables (red) and "sittable" objects (chairs, armchairs and sofas) (blue).

Table 1: Comparison on the S3DIS dataset. Values in **bold** indicate the better-performing method for each metric and room.

| | Mask R-CNN | | | | | | YOLOv8n | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | | Recall | | IoU | | Precision | | Recall | | IoU | |
| | LDLS | Our | LDLS | Our | LDLS | Our | LDLS | Our | LDLS | Our | LDLS | Our |
| Conference Room 1 | 0.62 | **0.68** | 0.57 | **0.73** | 0.16 | **0.38** | 0.69 | **0.82** | 0.68 | **0.73** | 0.21 | **0.52** |
| Lounge 1 | **0.60** | 0.17 | **0.41** | 0.34 | **0.11** | **0.11** | **0.62** | 0.60 | 0.21 | **0.32** | 0.05 | **0.08** |
| Lounge 2 | **0.73** | 0.66 | 0.55 | **0.62** | **0.29** | 0.17 | 0.78 | **0.84** | 0.57 | **0.60** | **0.29** | 0.25 |
| Office 3 | 0.88 | **0.90** | 0.11 | **0.49** | 0.18 | **0.20** | 0.84 | **0.92** | **0.38** | **0.38** | 0.21 | **0.24** |

ages in between, i.e., the localization should work even though the images are not taken within a short time frame of each other. To this end, an efficient way of finding corresponding instances across views is necessary. We will outline our novel approach next.

For each view at time $t$, local instances $L_i^t$ are created. The next step is to merge these to *global instances* $G_i$ which are independent of the view. The first set of local instances are directly mapped to global instances, i.e., $G_i = L_i^0$; however, subsequent masks are merged based on a set-based instance assignment heuristic, outlined in Algorithm 1. In essence, when a new set of local instances $\{L_i^t\}_{i=1,\dots,N_t}$ are found by the semantic segmenation algorithm, each local instance $L_i^t$ is assigned to a corresponding global instance. We select the global instance which has the highest number of common elements, i.e., $G_k \coloneqq G_k \cup L_*^t$, where

$$L_*^t = \underset{i=1,\dots,N_t}{\operatorname{argmax}} |G_k \cap L_i^t| \,. \tag{1}$$

If there are no common elements, a new global instance is created. Furthermore, to avoid merging of instances due to noisy segmentation masks we demand that the intersection between the instances is proportionally large with respect to the cardinality of the local and global sets considered. However, we only do this if the class labels are the same. If instead, the class labels are different, we assume these come from noise and remove them from the global instance.

### 3.3 Post-Processing

After creating the global instances, we apply post-processing to remove noisy points, by treating each global instance as a sub-point cloud. We apply DBSCAN to cluster the sub-point cloud and proceed by removing points that are further away from their neighbors compared to the average of the cluster. For smaller sub-point clouds, we found experimentally that one may directly use statistical filtering on the entire sub-point cloud without clustering. The bounding boxes are then computed as the convex hull of the 3D points, and the final label is taken to be the most common label from the local instances used to create the sub-point cloud.

## 4 EXPERIMENTS

We compare our method to LDLS (Wang et al., 2019), a state-of-the-art method in the category utilizing 2D image segmentation. We would like to emphasize that LDLS is not a feasible option for XR devices as it is too computationally demanding, which we will discuss further in Section 4.3. Furthermore, it should be noted that LDLS only handles single-view information, i.e., it does not aggregate data over multiple views.

**Algorithm 1:** Set-based instance assignment.

```
Match local instances L_i^t to global
  instances G_k (1);
if No match then
    Create new global instance from local
      instance;
else
    if Same label between local and global
      instance then
        Merge instances if |G_k ∩ L_*^t| is sufficiently
          large compared to |L_*^t| and |G_k|;
    else
        Remove common elements G_k := G_k \ L_*^t;
    end
end
Extract 3D bounding boxes from the global
  instances, see Section 3.3;
```

The goal of the comparisons is not only to show that we can get similar and sometimes better performance, but to emphasize that our method scales well to current XR devices. Three datasets were used in the experiments: S3DIS (Armeni et al., 2017), a synthetically generated dataset, and a real-world dataset collected using a Varjo XR-3 headset, with a scenario of placing an avatar in a physical environment. Furthermore, we show that it is possible to run our algorithm on an embedded device (Jetson Xavier NX).

LDLS is not designed for multi-view scenarios and performs 3D point cloud segmentation by leveraging semantic segmentation from a single 2D image taken by an aligned camera. For a fair comparison, we let LDLS process each view independently, resulting in multiple different potential labelings of the input point cloud, and pick the best label with respect to each ground truth object from all outputs. Bounding boxes are then computed by taking the convex hull of the points for each instance.

**Evaluation Metrics.** we use commonly occurring quality metrics used in semantic segmentation to evaluate our method. Precision and recall are computed point-wise over all classes and IoU score is reported with respect to the predicted 3D axis-aligned bounding boxes that the methods output. The mean scores are averaged over all classes.

### 4.1 Experiments on S3DIS

The S3DIS dataset (Armeni et al., 2017) is used to compare our framework with LDLS. S3DIS contains six large-scale indoor areas with 271 rooms. For the

evaluation, we specifically focused on Area 3, which consists of many environments including Conference Room 1, Lounge 1, Lounge 2, and Office 3, which we have used in these experiments. These environments are chosen as they capture a variety of different constellations. Furthermore, the S3DIS dataset contains larger object classes than the ones that Mask R-CNN and YOLO are trained on and as a result, the comparison to ground truth had to be limited to "sittable objects" and tables; however, these are important objects for interaction in XR environments.

The S3DIS data contains highly dense point clouds, while our framework is constructed to work on sparse 3D data. We subsample the point clouds and retain only 0.3% of the original data by randomly selecting points from each object in each scene. We use Mask R-CNN which was originally used and trained with LDLS, and YOLOv8 (Jocher et al., 2023) as our instance segmentation algorithms of choice. We use YOLOv8n—the smallest model—as it is most likely to run on an XR device. The output is shown in Figure 3 and the corresponding results in Table 1.

In Conference Room 1, our method outperforms LDLS in all metrics for both models. In Lounge 1, LDLS demonstrates superior or comparable performance in precision and IoU using Mask R-CNN, whereas our method shows improvement in recall with YOLOv8n. The significant decrease in precision for our method with Mask R-CNN in Lounge 1 is due to challenges in accurately detecting tables. Unlike the LDLS method, which effectively captures the entire height of the bounding boxes, our approach tends to recognize only the tabletops, leading to this disparity in performance. This specific limitation in detecting the full 3D structure of tables leads to a lower precision score in this particular scenario. On the other hand, this is not crucial for XR applications, as it is often the tabletop that one uses for interaction. In Lounge 2, LDLS has higher precision and IoU with Mask R-CNN, while our method shows better recall with both models. In Office 3, our method generally outperforms LDLS, particularly in recall.

The results suggest that the performance varies significantly depending on the room and the specific metric, with our method showing notable improvements in several cases.

## 4.2 Experiments on Synthetic Dataset

To overcome the issue of having 2D object detection algorithms that have been trained on different data, we created a synthetic dataset with objects that are present in the COCO dataset, which was used in training Mask R-CNN and YOLOv8. In this experiment, we use YOLOv8x the most powerful model in the series, as it is better at finding smaller objects, which makes the comparison more interesting. In Section 4.3 we discuss the feasibility of using such a comparatively heavy algorithm on device.

Table 2: Class-wise comparison on the synthetic dataset objects. Values in **bold** indicate best performance.

| | Precision | | Recall | | IoU | |
|---|---|---|---|---|---|---|
| | LDLS | Our | LDLS | Our | LDLS | Our |
| Bottle | 0.63 | **1.00** | 0.01 | **0.37** | 0.24 | **0.34** |
| Bowl | 0.53 | **1.00** | 0.13 | **0.76** | 0.37 | **0.60** |
| Chair | 0.35 | **0.99** | 0.04 | **0.19** | 0.02 | **0.11** |
| Clock | **1.00** | **1.00** | 0.26 | **0.80** | 0.05 | **0.90** |
| Fridge | **1.00** | **1.00** | 0.11 | **0.49** | 0.04 | **0.22** |
| Glass | 0.00 | **1.00** | 0.00 | **0.18** | 0.00 | **0.07** |
| Jar | 0.63 | **1.00** | 0.02 | **0.20** | 0.01 | **0.12** |
| Sink | 0.60 | **0.65** | 0.13 | **0.47** | 0.06 | **0.39** |
| Table | **0.95** | 0.33 | 0.07 | **0.30** | 0.04 | **0.06** |
| Wine glass | **0.99** | **0.99** | **0.53** | 0.51 | **0.61** | 0.35 |

Furthermore, the dataset shares many similarities with SLAM-generated dataset where the points originate from keypoint descriptors. For example, large untextured objects (e.g., the table) have very few points, while smaller objects (e.g., bottles) can have many. This is not the case for lidar-scanned datasets and other techniques used in the S3DIS where the points are more uniformly spread.

The results are shown in Table 2. For smaller objects, our method significantly outperforms LDLS in all metrics. Larger objects, e.g., clock and fridge, both methods achieve perfect precision, but our method shows a considerably higher recall and IoU. However, in the case of larger objects like tables, our method, despite showing improved recall, tends to have lower precision compared to LDLS. As discussed in the previous section, this is related to the limitation in detecting the full 3D structure of tables. Overall, the superior performance in recall and IoU across most classes shows the effectiveness of our approach in broader object identification contexts, and LDLS does not work well for non-uniformly distributed point clouds as is common in image-based SLAM-generated datasets.

## 4.3 SMVLift on Real XR Hardware

As previously mentioned, LDLS is too computationally demanding. When executed on the synthetic dataset in Section 4.2 on a computer equipped with a GeForce RTX 3090 (24 GB) GPU, the average runtime was 241 ms excluding the object detection algorithm. Even for such powerful hardware, this cannot be considered for real-time applications. In this section, we demonstrate that SMVLift is capable of real-

Figure 4: (Left): Experimental setup of our scene. A user is scanning the scene with a Varjo XR-3. (Mid) Example screenshot from the Varjo XR-3 view where the three detected objects are present. (Right) The Varjo XR-3 is tethered, but future generations of XR hardware could (at least mechanically) fit more powerful hardware such as the Jetson Xavier NX 8 GB.


Figure 5: (Left): The sparse point cloud (1028 points) and the corresponding 3D bounding boxes aggregated from three views. (Right): An avatar is placed on the chair based on the bounding boxes computed in 3D space. The banana and apple are also known and could be tagged for interaction by a downstream task.

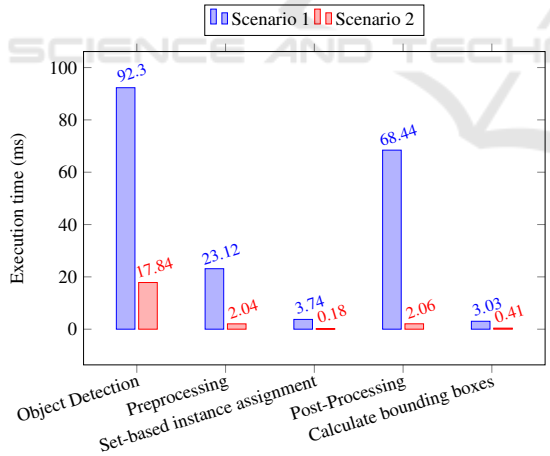


Figure 6: Profiling on a Jetson showing execution time for different configurations and parts of the pipeline.

time performance on limited hardware. To do so, we created a small dataset using the Varjo XR-3 headset, see Figure 4. A total of 33 images were captured from different positions in a room, facing a test scene. To simulate the offline SLAM process, c.f., Figure 1, we used COLMAP to generate the sparse point cloud. As the scene is poorly textured, with lots of neutral background, the SIFT keypoint extractor had trouble matching descriptors between frames, resulting in a final point cloud of only 1028 points. We argue that this is representative of many indoor scenarios, and regardless of the low-quality 3D reconstruction, the results are still usable for XR applications, as will be demonstrated.

In the online part of the pipeline, three images, previously not used by the framework, are used to estimate the pose and the 2D-3D correspondences in the localization part of the pipeline. Simultaneously the images are processed by the light-weight object detector YOLOv8n and the labels and segmentation masks are sent to the SMVLift block for processing. The corresponding 3D bounding boxes are shown in Figure 5. Furthermore, we use the computed bounding boxes to position an avatar on the chair to simulate its applicability in an XR application, see Figure 5. Note that the apple and the banana are correctly recognized and could be used by the avatar or the user for interaction.

Since the Varjo XR-3 is one of the most advanced PCVR mixed reality headsets available on the market and is tethered to a powerful computer with GPU support, it cannot be considered to have limited computational power. In order to get a reasonable measurement of the real-time performance of the proposed

pipeline for a standalone XR device, we use a Jetson Xavier NX 8 GB for benchmarking. This system-on-module device comes in a small form factor with GPU support and is widely used as an AI edge device, due to its cloud-native support and hardware acceleration made possible with the NVIDIA software stack. We consider two different scenarios:

**Scenario 1.** The synthetic dataset setup: YOLOv8x, 8 images, 22474 points, with 36 objects.

**Scenario 2.** The setup used in this section: YOLOv8n, 3 images, 1028 points, with 3 objects.

To utilize the most of the hardware, the device is configured to run in the maximum performing power mode (20 W) and the object detection models are converted to quantized TensorRT-optimized model files using INT8 precision. The preprocessing is done on GPU and the set-based instance assignment algorithm is implemented on CPU, while the post-processing is done using Open3D compiled with CUDA support. The results are shown in Figure 6. The total execution time for Scenario 1 is 190.63 ms per frame and 22.53 ms per frame in Scenario 2. Keep in mind that the localization part of the pipeline is not taken into account; however, it should be noted that it can be executed in parallel to the object detector.

## 5 DISCUSSION

Our approach shows notable strengths in identifying smaller objects and in scenarios where comprehensive detection is crucial. However, there are limitations in detecting the full structure of larger objects, like tables, which affect the precision in specific contexts. This insight is not problematic for applications in XR environments, where interaction often focuses on object surfaces like tabletops. Overall, the findings highlight the potential of our method in diverse applications, balancing between detailed detection and practical constraints in real-world scenarios.

## 6 CONCLUSIONS

In this work, we introduced a fast incremental algorithm (SMVLift) for lifting 2D semantics to 3D on constrained hardware. By working with sparse point clouds, on-device performance is made possible. For robustness, we aggregated the semantic masks from multiple views, by using a novel set-based instance segmentation algorithm. Our method was compared to a state-of-the-art algorithm and showed comparable or superior results despite being significantly more lightweight. In addition, we showed that our method can be incorporated into a real XR application by positioning an avatar on a chair using a Varjo XR-3 headset. Finally, we showed that the method is capable of real-time performance on a Jetson Xavier NX and argued that, due to the mechanical form factor of such devices, the computational capacity of future generations of XR devices are likely to be running algorithms such as the one proposed in the paper.

## REFERENCES


Armeni, I., Sax, A., Zamir, A. R., and Savarese, S. (2017). Joint 2D-3D-Semantic Data for Indoor Scene Understanding. https://arxiv.org/abs/1702.01105.

Hau, J., Bultmann, S., and Behnke, S. (2022). Object-level 3D semantic mapping using a network of smart edge sensors. In *IEEE International Conference on Robotic Computing (IRC)*, pages 198–206. IEEE.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988.

Heo, J., Bhardwaj, K., and Gavrilovska, A. (2023). FleXR: A system enabling flexibly distributed extended reality. In *Proceedings of the Conference on ACM Multimedia Systems*, pages 1–13.

Jocher, G., Chaurasia, A., and Qiu, J. (2023). Ultralytics YOLOv8. https://github.com/ultralytics/ultralytics.

Mascaro, R., Teixeira, L., and Chli, M. (2021). Diffuser: Multi-view 2D-to-3D label diffusion for semantic scene segmentation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 13589–13595.

Ngo, T. D., Hua, B.-S., and Nguyen, K. (2023). ISBNet: A 3D point cloud instance segmentation network with instance-aware sampling and box-aware dynamic convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13550–13559.

Wang, B. H., Chao, W.-L., Wang, f. Y., Hariharan, B., Weinberger, K. Q., and Campbell, M. (2019). LDLS: 3-D object segmentation through label diffusion from 2-D images. *IEEE Robotics and Automation Letters*, 4(3):2902–2909.

Wang, Y., Shi, T., Yun, P., Tai, L., and Liu, M. (2018). PointSeg: Real-time semantic segmentation based on 3D LiDAR point cloud. https://arxiv.org/abs/1807.06288.

Wu, Z., Zhao, T., and Nguyen, C. (2020). 3D reconstruction and object detection for HoloLens. In *Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–2. IEEE.

Zhang, H., Han, B., Ip, C. Y., and Mohapatra, P. (2020). Slimmer: Accelerating 3D semantic segmentation for mobile augmented reality. In *IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 603–612.