SAM2Object: Consolidating View Consistency via SAM2 for Zero-Shot 3D Instance Segmentation

Supplementary Material

A. Implementation Details

Iou-based object association algorithm. In our approach, we define keyframe as the frame where new objects appear. The keyframe is served as the last frame of the previous video clip, and is also served as the first frame of the current video clip.

Although the **video predictor** of SAM2 [35] can track objects and ensure masks fully and accurately cover them even under conditions such as occlusion, disappearance, and viewpoint changes, its effectiveness gradually diminishes as the number of tracking frames increases, potentially leading to blurred boundaries.

When applying the **image predictor** of SAM2 to the first frame of the current video (the same as the last frame of the previous clip) for object segmentation, it produces clear mask boundaries. Due to constraints like occlusion and viewpoint, the resulting masks may struggle to maintain completeness.

To complement the strengths of both approaches, after using SAM2 to track objects within video clips between keyframes, we employ an IoU-based object association algorithm at the junctions of these clips (specifically at the keyframes). Specifically, for the keyframe of current video, we generate two sets of masks. The first set, known as tracking masks, is the tracking result of the last frame of the previous video clip obtained by the video predictor of SAM2. The second set, referred to as keyframe masks, is generated by the image predictor of SAM2 on the first frame of the current video clip. We utilize the IoU score to associate and merge the two mask sets from the same object. This method not only generates accurate mask prompts but also ensures that objects tracked in the previous video can continue to be tracked in the current video, thereby maintaining mask consistency. More specifically, we define the tracking masks as $A_m = \{a_i | i \in [1, m]\}$, and the keyframe masks as $B_n = \{b_j | j \in [1, n]\}$. Thus, the IoU score between a_i and b_j can be defined as

$$IoU = \frac{|a_i \cap b_j|}{|a_i \cup b_j|}. \tag{13}$$

Similar to work [24], we define the intersection over the tracking mask (IoT) and the intersection over the keyframe mask (IoK) to measure the overlap between two sets of masks for the same object. Their definitions are as follows:

$$IoT = \frac{|a_i \cap b_j|}{|a_i|}, IoK = \frac{|a_i \cap b_j|}{|b_j|}. \tag{14}$$

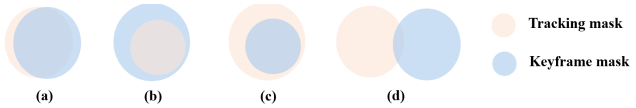

Figure 6. **Examples of four types of mask overlap.**

We calculate the overlap score between a_i and b_j based on the Eq. (13) and Eq. (14), resulting in four possible scenarios. Firstly, as shown in Fig. 6 (a), when the IoU exceeds a specified threshold (which we set to 0.8), we consider it a match for the same object and choose the keyframe mask as the final mask prompt due to its clearer boundaries. Secondly, when IoT exceeds 0.8 and IoK falls below 0.4, we relate this to the scenario shown in Fig. 6 (b), where most of the tracking mask is enclosed by the keyframe mask. This may occur because the image predictor of SAM2 is limited by factors such as viewpoint and occlusion, leading to under-segmentation of the object. Consequently, we select the tracking mask as the final mask prompt. Similarly, as illustrated in Fig. 6 (c), when IoK is greater than 0.8 and IoT is less than 0.4, the keyframe mask is largely surrounded by the tracking mask. This could be due to boundary blurring or expansion issues caused by the video predictor of SAM2 after tracking multiple frames. Therefore, we choose the keyframe mask as the final mask prompt. Fig. 6 (d) represents scenarios where none of the above three conditions are met, indicating that the two masks do not correspond to the same object.

Semantic instance segmentation. OpenMask3D [42] uses a pre-trained 3D instance segmentation model to generate class-agnostic 3D masks. These masks are used to select high-quality 2D image projections, which are then input into CLIP [34] to assign semantic labels. We follow the OpenMask3D approach to evaluate semantic instance segmentation. Specifically, we replace the class-agnostic 3D masks produced by the 3D instance segmentation model in OpenMask3D with the class-agnostic 3D masks output by our method on ScanNet200 [36]. All other steps remain consistent with the OpenMask3D process.

Parameter settings. All experiments are performed on a single RTX A6000. During the keyframe extraction process, we configure the window size to 100 frames. The threshold in the iterative graph clustering is empirically set as [0.9, 0.75, 0.6, 0.45, 0.3] for ScanNetV2 and [0.9, 0.75, 0.6] for ScanNet++.

B. Additional Qualitative Results

We show more visual comparisons between our method and SAM3D [47] and SAI3D [49] on ScanNetV2 [5] dataset in Fig. 8 and ScanNet++ [48] dataset in Fig. 9. SAM3D and SAI3D struggle with projection errors, using a frame-by-frame segmentation method across multiple views that results in a lack of view consistency. In contrast, our approach consolidates view consistency and filters out low-quality 2D masks, enabling us to achieve comprehensive and robust 3D instance segmentation.

As depicted in Fig. 10, we present the visual comparison results of semantic instance segmentation on ScanNet200 [36]. In semantic instance segmentation tasks, as shown in Fig. 10, given a text prompt, our method can effectively identify various objects in complex environments, even when they are surrounded and occluded by others (such as the "piano" and "guitar"). Additionally, our method can accurately identify small objects (such as the "hat" and "fire extinguisher") as well as objects located in wall corners (like the "trash can" and "pillow"). Since our method can generate precise and complete 3D segmentation, we can maintain clear object boundaries even in complex scenes. This capability is due to our consolidation of multi-view consistency.

C. Additional Quantitative Results

We show more quantitative results of class-agnostic 3D instance segmentation on ScanNetV2 [5].

Efficiency. Table. 5 presents the time required for all steps of graph construction and clustering (Sec. 3.3 and Sec. 3.4), showing that each scene, containing an average of 158K points, takes about 17s with our method, compared to 13s with SAI3D [49]. Only a **single** projection is required throughout the entire process to obtain the mask label matrix for all points across all frames. In each iterative clustering, superpoint graph construction, superpoint clustering, and superpoint updates are required. We can index the mask label matrix to obtain the superpoint mask label. The construction of the graph is efficient, and the number of superpoints gradually decreases after each iteration, making processing time gradually reduced.

Table 5. Time of graph construction and clustering on ScanNetV2.

Projection	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	All
4.82s	6.21s	2.05s	1.67s	1.49s	1.40s	17.64s

Ablation on the number of 2D images. In Fig. 7, following the approach of SAI3D, we conducted robustness experiments on ScanNetV2 to evaluate performance under varying proportions of images. Our method is robust across different image quantities and consistently outperforms SAI3D. As the number of images decreases, the difference between frames increases, leading to more extracted keyframes and ultimately making nearly every frame a keyframe. Consequently, when the image density becomes extremely sparse, our method may essentially transition to frame-by-frame segmentation. However, with the enhancement of mask consolidation, our method still outperforms SAI3D.

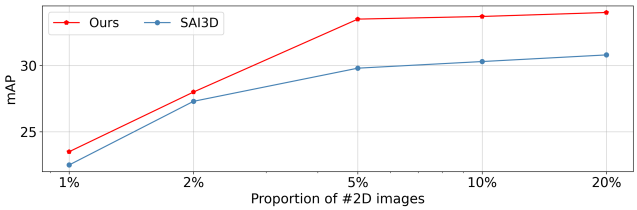

Figure 7. Correlation of performance with the number of images.

Ablation of mask consolidation. Initially, Eq. (11) did not explicitly incorporate detailed weighting strategies, and the weights were treated equally. We multiply three sub-weights to obtain the overall weight, followed by weight normalization, eliminating explicit scaling issue. The overall weight becomes significant only when all three sub-weights are large, ie. superpoints are only merged when all sub-weights have high scores. We conduct ablation experiments for the mask consolidation module and as shown in Table. 6, "Visibility" and "Distance" are relatively more important.

Table 6. Ablations for mask consolidation on ScanNetV2.

Visibility	Distance	Purity	mAP	AP_{50}	AP_{25}
			32.1	49.9	68.3
✓			33.0	50.5	68.9
✓	✓		33.7	52.1	69.5
✓	✓	✓	34.0	52.7	70.3

Comparing gaussian model. We also conducted comparative experiments with the 3D Gaussian segmentation method. Since it generates its own 3D scenes, to ensure fairness, we projected its 2D rendered segmentation results onto the 3D point cloud to create 3D segmentation. Our method achieve 44.2, 64.6 and 80.4 in terms of mAP, AP_{50} and AP_{25}, significantly surpassing those of Gaga [28] (4.6, 13.3, 33.3).

New object coverage. In our method, if a new object is missed, it lacks a corresponding 2D mask and is excluded from the lifting process, ultimately remaining unsegmented. By projecting the 3D labels onto 2D, we found that, on average, each object has a corresponding 2D mask in **97%** of its visible frames. Furthermore, only **0.6%** of all objects remain unsegmented in the final segmentation. This clearly demonstrates that our method is effective in most cases.

Figure 8. **Additional visual results of class-agnostic 3D instance segmentation on ScanNetV2 [5] dataset.**

Figure 9. **Additional visual results of class-agnostic 3D instance segmentation on ScanNet++ [48] dataset.**

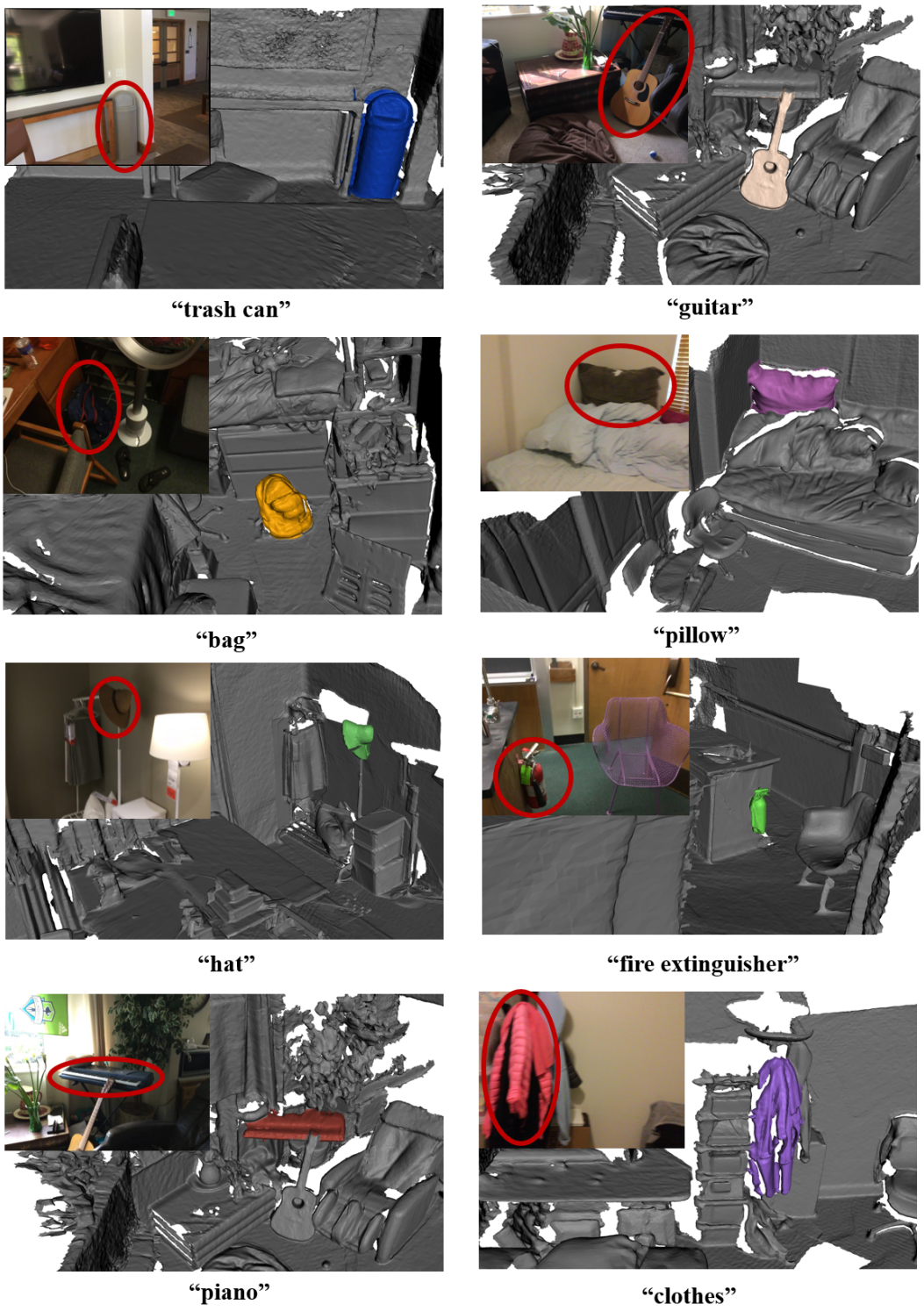

Figure 10. **Visual results of semantic instance segmentation on ScanNet200 [36] dataset.** Given a text prompt, we can accurately locate its position within the scene.