GraphSeg: Constructing Segmented 3D Representations via Graph Edge Addition and Contraction

Haozhan Tang[1]   Tianyi Zhang[1]   Oliver Kroemer[1]   Matthew Johnson-Roberson[1,2]   Weiming Zhi[1]

Abstract—Robots operating in unstructured environments often require accurate and consistent object-level representations. This typically requires segmenting individual objects from the robot's surroundings. While recent large models such as Segment Anything (SAM) offer strong performance in 2D image segmentation. These advances do not translate directly to performance in the physical 3D world, where they often over-segment objects and fail to produce consistent mask correspondences across views. In this paper, we present *GraphSeg*, a framework for generating consistent 3D object segmentations from a sparse set of 2D images of the environment without any depth information. GraphSeg adds edges to graphs and constructs dual correspondence graphs: one from 2D pixel-level similarities and one from inferred 3D structure. We formulate segmentation as a problem of edge addition, then subsequent graph contraction, which merges multiple 2D masks into unified object-level segmentations. We can then leverage *3D foundation models* to produce segmented 3D representations. GraphSeg achieves robust segmentation with significantly fewer images and greater accuracy than prior methods. We demonstrate state-of-the-art performance on tabletop scenes and show that GraphSeg enables improved performance on downstream robotic manipulation tasks.

I. INTRODUCTION

Robots operating in diverse environments typically need to construct internal representations of their surroundings from their sensor inputs. In many cases, individual objects within these representations need to be segmented out to facilitate downstream interaction with these objects. This paper tackles the problem of obtaining object-level representations via the segmentation of tabletop scenes, common to robot manipulation, from a set of multi-view 2D images of the tabletop.

Advances in computer vision have led to large pre-trained models, most notably *Segment Anything* [1], which can efficiently segment a 2D image into a set of masks. However, these models do not extend directly into the segmentation of 3D representations or over a set of multi-view images. Pre-trained models that segment 2D images into semantically meaningful parts often *over-segment*, that is, they may break down a consistent object into multiple masks. Additionally, it is challenging to accurately find correspondences between the masks over different images, and obtain masks which correspond to masks over the robot's 3D representation.

In this paper, we propose *GraphSeg*, a framework that produces segmented object-level representations. GraphSeg models the correspondence of masks over multi-view images as graphs, and then formulates the matching of segmentation masks as a graph edge addition and contraction problem. A graph contraction problem seeks to simplify a graph by

[1] Robotics Institute, Carnegie Mellon University, Pittsburgh, USA.
[2] College of Connected Computing, Vanderbilt University, USA.
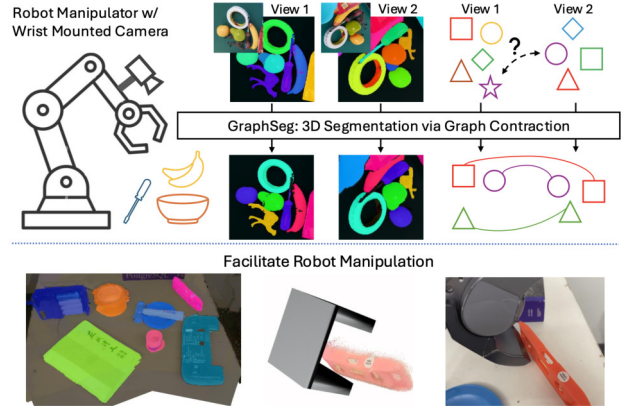
Fig. 1: Robots often need to work with object-level representations. In this work, we tackle the problem of building segmented 3D scenes from a set of images, and introduce the GraphSeg framework. GraphSeg solves multi-view 3D segmentation via a novel graph edge addition and contraction procedure. This facilitates downstream robot manipulation.

removing edges and merging adjacent vertices to obtain super-vertices, which contain several of the original vertices.

GraphSeg adds edges by considering the correspondence masks, based on both pixel-level correspondences over the 2D images, as well as inferred correspondences over the underlying 3D structure of the environment. We demonstrate that by formulating and efficiently solving our graph edge addition and contraction problem, GraphSeg can reliably produce consistent 3D segmentations, is robust to over-segmentation, and can operate even with very few images (sparse-view), outperforming existing methods by a large margin. Additionally, the fully segmented structure is produced by leveraging *3D foundation models*, which learn to recover 3D representations from multi-view images, and no depth readings from sensors are required. Concretely, our technical contributions are as follows:

1) The GraphSeg framework capable of producing cleanly segmented 3D scene representations;
2) The formulation of 3D segmentation as a problem of graph edge addition and contraction, along with algorithms to efficiently solve the graph contraction;
3) Extensive empirical evaluation showing state-of-the-art 3D segmentation performance, and demonstrating utility to downstream robot manipulation tasks.

II. RELATED WORK

Early deep learning methods for image segmentation were driven by advances in Convolutional Neural Networks, including Deeplab [2] and SegNet [3]. With the emergence of 3D
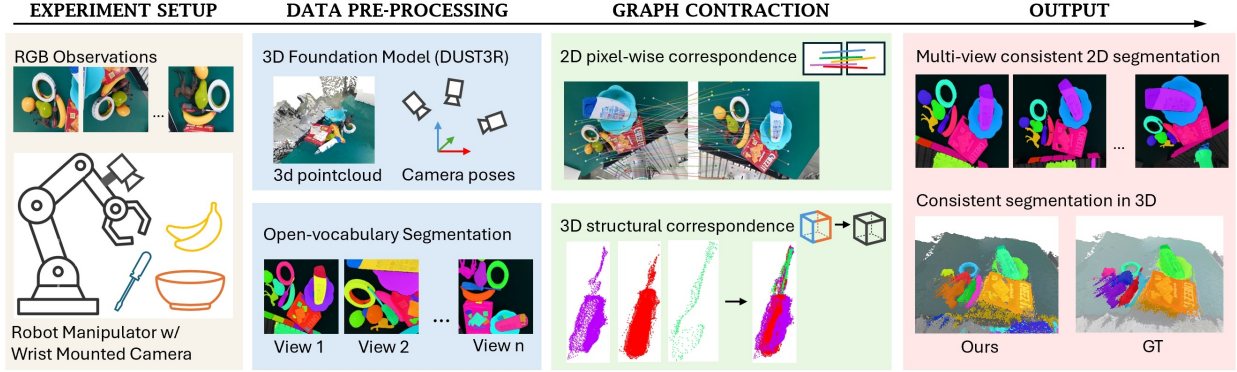
Fig. 2: GraphSeg enables consistent 3D segmentation. We can obtain a set of segmented 2D images by leveraging the pre-trained open-vocabulary segmentation model. We then leverage edge addition via correspondence and graph contraction, over both 2D and lifted 3D representations, to obtain segmented 3D representations.

datasets [4], [5] and the increasing demand for robot manipulation, many approaches have been proposed for 3D instance segmentation to support control and planning. These methods often fall into three categories: (1) closed-set 3D instance segmentation, (2) deep learning-based open-vocabulary 3D instance segmentation, and (3) zero-shot open-vocabulary 3D instance segmentation. **Close-set 3D instance segmentation** methods operate on a fixed set of classes, assigning parts of the 3D representation to each class [6]–[10]. These models use prior knowledge of predefined classes and perform poorly in open-vocabulary scenarios. Our method is designed for open-vocabulary settings without predefined instance priors. **Deep learning-based open-vocabulary 3D instance segmentation** approaches train on annotated 3D segmentation datasets to predict instance masks rather than assigning regions to fixed classes [11], [12]. However, due to limited 3D instance data, these models often fail to generalize in real-world tabletop scenarios without fine-tuning. Our method aims to generalize zero-shot to arbitrary tabletop settings for fixed-base robot manipulators. Although 3D data remains limited, the computer vision community has produced a wealth of 2D segmentation datasets [13]–[19], enabling robust class-agnostic 2D segmentation via foundation models like CropFormer [20] and SAM [21]. **Zero-shot open-vocabulary 3D instance segmentation** methods such as MaskClustering [22], SAM3D [23], and OVIR-3D [24] leverage 2D segmentation models and aggregate segmented point clouds using 3D spatial relationships. These are most comparable to our proposed GraphSeg, which we evaluate against. Other approaches [25], [26] project 3D data to 2D and apply visual-text embeddings like CLIP [27] and DINO [28] to guide segmentation. However, 3D-centric methods may fail in sparse view settings, and embedding-based methods struggle with semantically similar objects. Our approach differs by formulating segmentation as a graph edge addition and contraction problem, leveraging both optical 2D and structural 3D correspondence for improved reliability.

III. PRELIMINARIES

GraphSeg makes use of advances in *foundation models*. Foundation models are large deep learning models trained on large and diverse datasets, and are intended as plug-

and-play modules to facilitate downstream operations in a zero-shot manner [29]. In this work, we explore foundation models applied to segmenting 2D images, and constructing 3D representations. Here, we will briefly elaborate on pre-trained segmentation models, 3D foundation models, as well as *graph contraction* which is embedded into our GraphSeg formulation.

Pre-trained Segmentation Models: Segment Anything (SAM) [21] is an open-world segmentation model that produces high-quality mask proposals for any object in an image without relying on predefined semantic categories. Given an rgb image $I \in \mathbb{R}^{H \times W \times 3}$, SAM automatic mask generator output a set binary mask proposals

$$\text{SAM}(I) = \{m_1, m_2, \ldots\}, m_i \in \{0,1\}^{H \times W}. \quad (1)$$

It is trained on a large and diverse 2D dataset so that it generalizes across domains in 2D single image segmentation without being restricted to any semantic class. Combining with DINO [28] visual-semantic embedding, we can use a semantic prompt to reliably extract a specific instance mask, such as the bare tabletop from a single image.

3D Foundation Models: 3D Foundation models projects sets of 2D images into 3D. Suppose we have a pair of RGB images with width W and height H, i.e. $I_1, I_2 \in \mathbb{R}^{W \times H \times 3}$, 3D foundation models, such as DUST3R [30] specifically, produces pointmaps $X^{1,1}, X^{1,2} \in \mathbb{R}^{W \times H \times 3}$, which map each 2D pixel to its predicted 3D coordinates aligned under the first image's coordinate frame. It also generates confidence maps $C^{1,1}, C^{1,2} \in \mathbb{R}^{W \times H}$ for each pointmap to quantify the uncertainty in the foundation model's predictions at each pixel. By matching each pixel's predicted 3D coordinates in one pointmap with the nearest coordinates in the corresponding pointmap, dense correspondences between pixels in the image pair can be established without relying on handcrafted features. The point cloud of the pair of I_1 and I_2 are then aligned to predict the corresponding camera poses P_1, P_2 in reverse. As a result, we can obtain pixel-wise correspondence, an aligned 3D point cloud, camera poses by using these foundation models as a black box function. As a result, we can obtain pixel-wise correspondence, an aligned 3D point cloud, camera poses by using these foundation models as a black box function. 3D foundation models have been used to understand the geometry
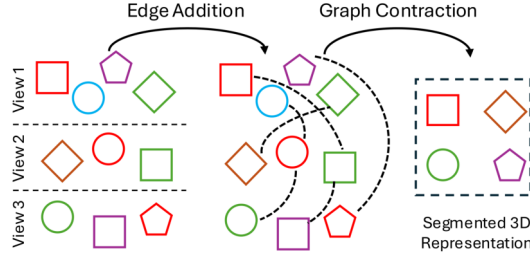
Fig. 3: At the core of GraphSeg is an edge addition and graph contraction process. The edge addition is achieved by finding correspondences between masks, via pixel-to-pixel features and 3D structural information.

of objects during grasping [31], [32], and can enable the construction of 3D representations [33]–[35].

Graph Contraction: In a typical graph, we have a set of vertices and edges. A graph partition divides the graph into subsets, where each subset consists of vertices that are associated with a representative vertex of that partition. The process of graph contraction—a classical concept in theoretical computer science—aims to produce such a partition by mapping together vertices that are connected, grouping them into the same subset. Formally, for a given graph (V, E),

$$\text{GraphContract}(V, E); \mapsto; f, \quad \text{where } f : V; \rightarrow; \tilde{V}, \quad (2)$$

and $\tilde{V}$ denotes the set of representative vertices corresponding to the graph partition.

IV. GRAPHSEG

Here, we are assumed to have a fixed-based robot manipulator with a mounted camera operating in a classical tabletop setup. We control the end-effector manipulator to take a small set of N RGB images of the tabletop, $\{I_1, \ldots, I_N\}$ where $I_i \in \mathbb{R}^{H \times W \times 3}$. We leverage the large open-world segmentation model, Segment Anything [1], to obtain 2D segmented masks from each image.

GraphSeg aims to output the corresponding segmented point cloud representation of the tabletop in the form of: (1) the relative aligned camera poses $\{P_1, \ldots, P_N\}$; (2) 3D point clouds associated with each image, $\{\hat{X}_1, \ldots, \hat{X}_N\}$; (3) Segmentation image $\{\hat{M}_1, \ldots, \hat{M}_N\}$ that corresponds to the result of segmenting the point cloud, where each point is assigned a mask class that is consistent over all of the images. We can then extract individual objects in the scene by extracting the corresponding 3D points associated with the object class.

A. Overview

The overview of our method is given in Figure 2. The crux of our method lies in finding correspondences between the abundant segmentation mask classes over a set of multi-view images. The main challenges in extending 2D segmentation to consistent multi-view image segmentation are that each of the images may contain different sets of objects, and that over-segmentation may occur in many of the images.

Our key insight is to formulate the matching of potentially inconsistent segmentation masks over multiple images as graph edge addition and contraction problems. We begin by assuming that each individual class in a mask of each image

is a vertex. Through edge addition and graph contraction, we reduce the set of vertices, where each new vertex is a combination of previous vertices. The process of edge addition and graph contraction is illustrated in fig. 3. Vertices are contracted together if they are masks of the same object captured over different views, and the final set of vertices corresponds to each segmented object, as shown in fig. 4.

Here, we seek to leverage both the image-level information and the estimated 3D spatial properties. This is achieved by constructing dual correspondence graphs through an edge addition process. One of these graphs estimates 2D pixel-level correspondences, and the other correspondence graph is constructed based on the 3D structure captured in each image, obtained by lifting the 2D image representation into 3D via 3D foundation model. After the edge addition process, these graphs are *contracted* with connected vertices being absorbed into a single vertex.

B. Pixel-to-Pixel 2D-based Edge Addition

We seek to leverage the visual information between the images in the collected image set to produce initial correspondences between masks from different images. Here, we each pass a pair of images into a pre-trained dense feature matching model [36]. We obtain a set of pixel-pairs across the two different input images, along with confidence estimates. We keep all confident pairs of pixel matches, and denote the set of matching pixels by $\mathcal{F}$. We add edges between two masks if the proportion of matching pixels in the mask exceeds a proportion threshold. That is, let $h(m_1, m_2)$ be the number of matching pixel pairs detected between masks m_1, m_2, i.e.

$$h(m_1, m_2) = |\{(p, p'), (p, p') \in \mathcal{F} | p \in m_1, v \in m_2\}|. \quad (3)$$

We add the edge between the vertices m_1 and m_2, if the number of matched pixels relative to the size of the smaller mask exceeds a threshold, i.e.,

$$\frac{h(m_1, m_2)}{\min(g(m_1), g(m_2))} \geq \tau_{2D}, \quad (4)$$

where $g(\cdot)$ is the pixel area of the mask, and τ_{2D} is a threshold for which we consider two masks to be correspondent and assign an edge. We then have a graph $G_{2d} = (V, E)$, where V contains the set of all 2D segmentation masks, and E contains all of the correspondence edges found by pixel-to-pixel matching. Here, we note that due to over-segmentation, a mask from one image may match with multiple masks in another image. The graph G_{2d} is then contracted to obtain the set $\{m'_1, \ldots m'_p\}$ where each m' is a super-vertex contains multiple 2D masks, which have been identified to be the same object, over the image set and corresponds to an initial estimate of the 3D segmentation.

C. Structural 3D-based Edge Addition

Advances in learning-based dense 3D reconstruction have enabled even sparse sets of multi-view images to be *lifted* into 3D. We seek to leverage the structural information of the 3D representations to further refine the initial segmentation obtained from contracting the pixel-to-pixel correspondence graph. Using only 2D pixel-to-pixel graph contraction, the
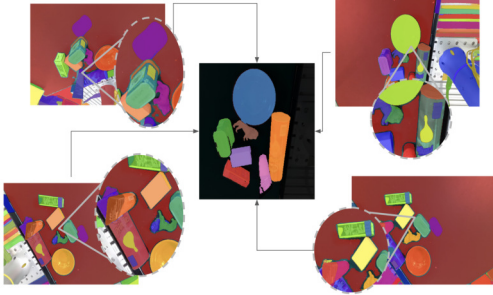
Fig. 4: GraphSeg can take over-segmented 2d images (e.g. the labels have been unnecessarily segmented from the can and bottle) and produce consistent 3D segmentations (shown in the center).

resulting multi-image mask can segment the point cloud reasonably well. However, we can still observe over-segmentation in the set of super-vertices. A key reason is due to excessive surface detail. The initial 2D segmentation model often produces multiple small masks on the same object surface in many images. This typically occurs for objects with rich surface features, making it difficult for the generator to determine whether surface regions belong to the same object across views.

We add further graph edges over the set of super-vertices. As the 3D foundation model produces a 3D point for each pixel in the input images, for each super-vertex, there is a one-to-one mapping to a point cloud. We measure the directed Chamfer distance of the corresponding points in 3D between two super-vertices, given by

$$D(X, Y) = \frac{1}{|X|} \sum_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} ||\mathbf{x} - \mathbf{y}||_2^2, \qquad (5)$$

where X and Y are 3D point clouds corresponding to the super-vertex masks. A small distance indicates that X is subsumed in Y and we can connect an edge correspondence, giving us a resulting graph $G_{3D} = (\hat{V}, \hat{E})$. Here, $\hat{V} = \{m'_1, \dots m'_p\}$ contains the solutions from contracting the pixel-to-pixel correspondence graph, and $\hat{V}$ contains edges if $D(m'_1, m'_2) \leq \tau_{3D}$, for any $m'_1 \neq m'_2$. Running graph contraction on the corresponding G_{3D} produces our solution super-vertices, each represents a 3D segmentation.

D. Graph Contraction via Randomized Star

To contract the constructed correspondence graphs, we leverage a variant of **randomized star contraction algorithm**, designed for efficient graph reduction. Each vertex v in the graph $G = (V, E)$ is assigned a random label $L(v) \in [0, 1]$ to break symmetry. For every vertex, define its closed neighborhood as

$$N[v] = \{v\} \cup \{u \mid (u, v) \in E\}. \qquad (6)$$

A vertex becomes a star center if its label is the smallest in its neighborhood, $L(v) = \min_{u \in N[v]} L(u)$. Otherwise, it contracts to the neighbor with the minimum label:

$$v^* = \arg \min_{u \in N[v]} L(u), \qquad (7)$$

and the edge (v, v^*) is contracted, merging v into the super-vertex represented by v'. This contraction step is performed in parallel across all vertices and recursively applied to the

Method	Graspnet-1B-main				Graspnet-1B-similar				Graspnet-1B-unseen			
	IOU	F1	d_{chamfer}	IoU$_{sel}$	IOU	F1	d_{chamfer}	IoU$_{sel}$	IOU	F1	d_{chamfer}	IoU$_{sel}$
SAM3D	0.319	0.4484	0.0556	0.317	0.3721	0.4877	0.0539	0.3708	0.346	0.4814	0.0625	0.3444
MaskClustering	0.0137	0.0311	0.0421	0.0125	0.0133	0.0309	0.0337	0.0138	0.0132	0.0315	0.0461	0.0109
GraphSeg⁻ (ours)	0.3916	0.5904	**0.0042**	0.3549	0.3966	0.5814	**0.0031**	0.3656	0.373	0.5563	**0.0027**	0.348
GraphSeg (ours)	**0.5945**	**0.7595**	0.0142	**0.5816**	**0.6522**	**0.7997**	0.0131	**0.6418**	**0.7308**	**0.848**	0.0065	**0.7281**

TABLE I. Quantitative comparison: We exhaustively test the performance of our method and compare it to the baseline method on the Graspnet-1B-main, Graspnet-1B-similar, and Graspnet-1B-unseen datasets [5].

Method	SAM3D	MaskClustering	GraphSeg (ours)
Graspnet-1B-main	0.3714	0.3815	**0.6982**
Graspnet-1B-similar	0.4146	0.3971	**0.743**
Graspnet-1B-unseen	0.3864	0.4025	**0.8121**

TABLE II. Each of the different methods filters out low-confident pixels at different rates. Here, we evaluate the quality of segmentation based on precision across all scenes. This is not sensitive to the number of points filtered.

resulting graph until a trivial structure remains. The use of random labels L ensures unbiased decision-making and balanced contraction, preserving essential connectivity while significantly reducing the graph's complexity.

After obtaining our super-vertices, we can easily obtain the entire 3D point cloud of any segmented object simply by specifying the 2D mask of the object in the set of images, by searching through each super-vertex for the ID of the 2D mask and checking which super-vertex the mask was subsumed into.

V. EMPIRICAL EVALUATIONS

In this section, we empirically evaluate the efficacy and robustness of GraphSeg. We both qualitatively compare Graph-Seg against state-of-the-art 3D segmentation methods, and quantitatively illustrate the robustness of GraphSeg to over-segmentation by the 2D segmentation model. Additionally, we stress test GraphSeg in the sparse multi-view setting, where there is limited overlapping visual features between the images. Many comparison methods filter away pixels from the inputs about which the methods are unsure. Next, we investigate the *pixel utility* of various methods, that is, how many pixels remain in the final result. Finally, we demonstrate the applicability of GraphSeg for grasping objects within a tabletop environment.

A. Experimental Setup

Datasets and Implementation: GraspNet-1Billion [5] dataset is a large and comprehensive dataset which consists of three large subsets of objects (main, similar, and unseen) and 97,280 images, along with camera poses, and fine-grained instance annotations on each pixel, which make it an optimal choice to evaluate our 3D segmentation method. The dataset has been carefully labelled, with a ground truth class assigned to each pixel. GraphSeg uses vision information only, and does not require any depth information included in the data. All of our experiments were conducted on a computer with Intel i9-14900HX CPU and NVIDIA GeForce RTX 4090 GPU. We use the default parameters provided by the original authors for the comparison methods, MaskClustering and SAM3D; We select the following hyperparameters for GraphSeg: the threshold for edge addition based on 2D correspondence is selected

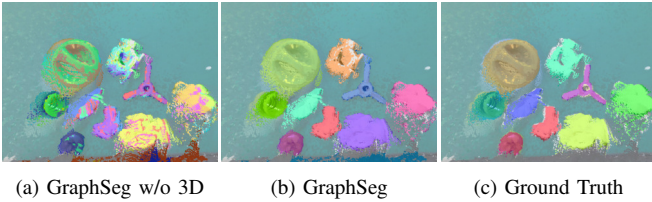(a) GraphSeg w/o 3D   (b) GraphSeg   (c) Ground Truth

Fig. 5: Although GraphSeg without 3D edge addition and graph contraction can identify individual objects, producing a small chamfer distance. It over-segments on feature-rich objects. This results in multiple masks being assigned to a single 3D object, as shown in (a). This is not the case after considering structural information as shown in (b).

as the value of the 78th percentile of all considered values. During the pixel-to-pixel edge construction, we subsample the number of sampled 2D correspondence points to 10000. The threshold for adding correspondence in the 3D edge addition step is given by $\tau_{3D} = 5e - 4$. In practice, GraphSeg is not overly sensitive to hyper-parameters, with values in the rough ballpark providing strong results.

Metrics: To evaluate the quality of 3D segmentation, we first want to find the subset of the segmented point cloud that corresponds to the target object. For each object in a scene, we find the corresponding point cloud by selecting (1) the segmented object class with the highest intersection over union (IoU) or (2) finding the segmented object class with the lowest chamfer distance, where the chamfer distance is defined by

$$d_{\text{chamfer}}(S, S') = \sum_{x \in S} \min_{y \in S'} \|x - y\|_2^2 + \sum_{y \in S'} \min_{x \in S} \|y - x\|_2^2. \quad (8)$$

We compute the IoU and $F1$ score between the ground truth object point cloud, and the corresponding segmented object point cloud found via highest IoU, denoted as IoU and $F1$. We also report the chamfer distance (denoted as d_{chamfer}) and IoU (denoted as IoU_{sel}) between the ground truth object point cloud and the corresponding segmented object point cloud found via the lowest chamfer distance. To show the necessity of considering correspondences based on both 2D structural 3D information, we also report the same metrics compute on the output of our method without the 3D graph edge addition and contraction, denoted as GraphSeg$^-$. The mean of IOU, $F1$, d_{chamfer}, and IoU_{sel} over all objects in all scenes in each dataset is reported in table I.

B. Segmentation Quality

Our method achieved the state-of-the-art performance in all metrics except d_{chamfer} in all datasets. Although the best d_{chamfer} is reached by GraphSeg$^-$, it is because GraphSeg$^-$ tend to over-segment the scene as shown in fig. 5, which is supported by the corresponding lower IoU_{sel} compared to GraphSeg with 3D graph edge addition and contraction. MaskClustering (MC) reaches good d_{chamfer} but poor IoU, which is likely because it filters out many points, and it is challenging to find 3D relations in a sparse view setup. SAM3D provides reasonable performance in some scenes but not in all scenes. Compared to the two baseline methods, our method can robustly segment the scene and out-performs the very recent baselines significantly. Additionally, we also
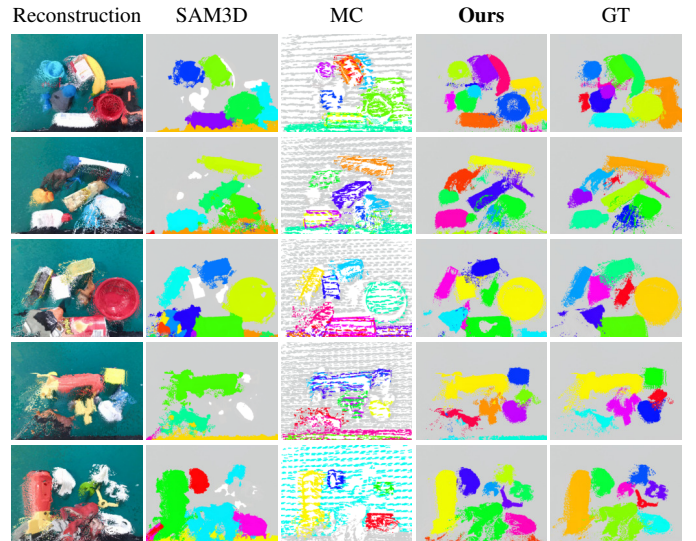

Fig. 6: Qualitative comparison of our method against the two baseline methods [22], [23] on GraspNet-1B [5]. MaskClustering aggressively filters out many points as a part of its filtering and point cloud refinement, producing significantly sparser 3D segmented point clouds.

Pixel Utility	MaskClustering	SAM3D	GraphSeg (ours)
Median	0.0268	0.8648	**0.9305**
Mean	0.0267	0.8506	**0.9263**

TABLE III. Here, we compare the mean and median of our method's pixel utility against baseline methods, over all our available datasets.

compare the precision between our method and the baseline methods. Methods such as MaskClustering aggressively filter out points which are uncertain, resulting in low recall, F1, and IoU measure, but do not adversely impact precision, lower Chamfer distance. As shown in table II, MaskClustering produces reasonable results as measured by precision and also Chamfer distance. However, our method still outperforms the compared baselines, measured by all of the metrics.

To qualitatively assess our method, we visualize the result of our method against MaskClustering [22] and SAM3D [23] in GraspNet-1B [5] in fig. 6. As shown, our method avoids the under-segmentation, displayed by SAM3D, by using 2D correspondence to first get an initial fine-grained segmentation. This reduces aggressive merging, when relying heavily 3D structure as is done by SAM3D. MaskClustering has an over-segmentation issue in the experiment, which is likely caused by the challenging sparse multi-view setup where the spatial relationship is harder to derive. Our method solves this by fusing 2D correspondence and 3D correspondence information to build additional edges and contracting the segmentation graph more robustly. An illustration of GraphSeg relative to comparison methods is also shown in fig. 7.

C. GraphSeg Produces Solutions with High Pixel Utility

Downstream robotics tasks generally benefit from denser and more complete point clouds. The process of constructing 3D representation, including the 3D foundation model, along with previous methods to construct consistent segmentation masks, filters out pixels for which the models are not confident. This is particularly true for MaskClustering, which relies on

(a) Ground Truth   (b) MaskClustering
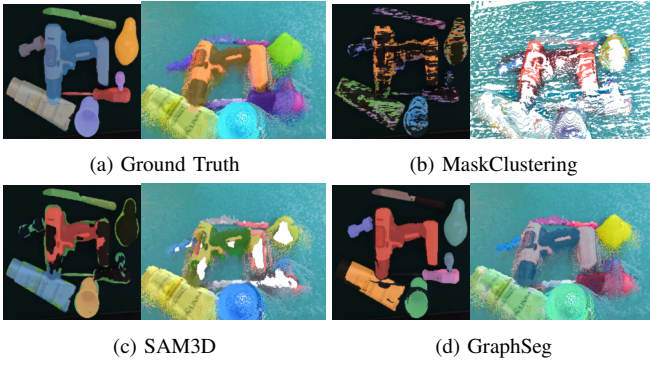
(c) SAM3D   (d) GraphSeg

Fig. 7: For each method, its segmentation output is visualized in 2D (on the left) and the corresponding segmented point cloud on the right. We can see that both SAM3D and MaskClustering suffer from over-segmentation, which caused objects in the segmented 3D points to be noisy, missing, or incomplete, while GraphSeg shows its robustness against over-segmentation to ensure objects are intact.

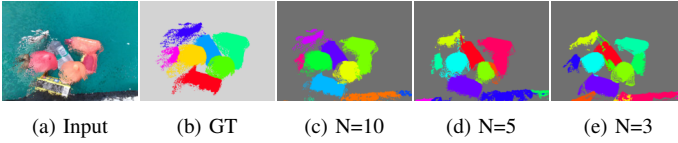(a) Input   (b) GT   (c) N=10   (d) N=5   (e) N=3

Fig. 8: Sparse View Robustness: GraphSeg is also tested under sparse views. We observe robust performance even when the number of images (denoted by N) is small.

aggressively filtering out uncertain points. More 2D pixels being utilized to project into 3D space provides us with a denser point cloud. Here, we seek to measure, for non-background pixels across images, how many are utilized by the resulting masks. The pixel utility is therefore computed by taking the number of non-background pixels identified by each method over the actual number of non-background pixels in ground truth across images, shown in table III. Our method has high ($> 90\%$) pixel utility comparing to other methods. Therefore, compared to baseline methods, GraphSeg retains a significantly larger number of pixels.

D. GraphSeg is Robust Under Sparse-View

To further evaluate the performance of GraphSeg, we test it under a more challenging scenario where the view is becoming increasingly sparse. Specifically, we evaluate the performance of GraphSeg, in which only $N = 5$ and $N = 3$ number of images are provided. This condition challenges GraphSeg with less pixel correlation and significantly less 3D spatial information. We visualize the result in fig. 8, from which we observe that GraphSeg retains its performance and provides reliable 3D segmentation even under sparse view conditions.

E. Downstream Manipulation

Segmentation of tabletop scenes to reveal object-level representations is crucial for robot manipulation. Typically, object representations, such as point clouds, are given to off-the-shelf grasp generators to generate feasible grasps for the robot. We demonstrate the effectiveness of GraphSeg on a real-world Unitree Z1 manipulator. A RGB camera is mounted on the end-effector of a robot manipulator, and objects are placed on the table top in front of the robot manipulator, then the robot

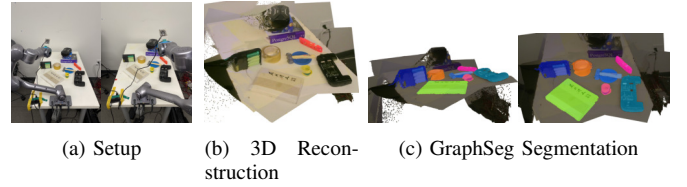(a) Setup   (b) 3D Reconstruction   (c) GraphSeg Segmentation

Fig. 9: (A) Real World Experiment Setup; (B) 3D reconstruction; (C) We run GraphSeg on the point cloud and RGB images, and output masked objects that we can grasp.
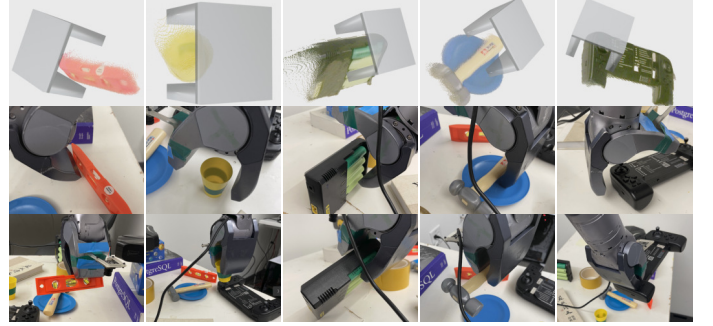

Fig. 10: Visualization of grasping various segmented instances. At the top, we illustrate the computed grasps, overlaid on the object-level representations built. We visualize the executed grasps for each object in the next two rows.

manipulator moves around to take several RGB images of the tabletop. We utilize a 3D foundation model to generate point cloud and camera poses, and align it to the robot base via a hand-eye calibration method [37];

We run GraphSeg on the point cloud and RGB images, and output the segmentations of objects that we can grasp. We leverage GraphSeg to extract individual object point clouds from a set of multi-view RGB images of an unstructured table setup. These point clouds are then inputted, alongside gripper parameters, to a grasp generator. We use grasp generator [38], and filter infeasible solutions based on collision-checking against a signed distance field of the environment. We demonstrate successful grasp, on the real-world manipulator, for a variety of objects, including a surveyor's level, a hammer, a small cup, a battery package, and a remote controller. The experimental setup, 3D reconstruction, along with masks over nearby objects on the table, are all illustrated in fig. 9. In fig. 10, we illustrate the resulting grasps computed over object-level representations, along with the successful grasps given in the subsequent rows.

VI. CONCLUSIONS

We propose GraphSeg, a framework that generates consistent 3D segmentations for tabletop scenes from a few RGB images. GraphSeg formulates segmentation as a graph edge–addition and contraction task, merging initially over-segmented 2D masks via both pixel-level and 3D structural correspondences. By leveraging 3D foundation models to recover scene geometry, GraphSeg preserves more object details and avoids over-segmentation. We empirically evaluate GraphSeg against state-of-the-art baselines, and demonstrate the performance and robustness of GraphSeg. We demonstrate that the dense object-level 3D representations produced enable downstream robot manipulation in the real-world.

REFERENCES

[1] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4015–4026, 2023.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[4] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5828–5839, 2017.

[5] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11444–11453, 2020.

[6] T. Vu, K. Kim, T. M. Luu, T. Nguyen, and C. D. Yoo, "Softgroup for 3d instance segmentation on point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2708–2717, 2022.

[7] W. Hu, H. Zhao, L. Jiang, J. Jia, and T.-T. Wong, "Bidirectional projection network for cross dimension scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14373–14382, 2021.

[8] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3d: Mask transformer for 3d semantic instance segmentation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8216–8223, IEEE, 2023.

[9] Z. Hu, X. Bai, J. Shang, R. Zhang, J. Dong, X. Wang, G. Sun, H. Fu, and C.-L. Tai, "Vmnet: Voxel-mesh network for geodesic-aware 3d semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15488–15498, 2021.

[10] L. Han, T. Zheng, L. Xu, and L. Fang, "Occuseg: Occupancy-aware 3d instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2940–2949, 2020.

[11] Y. Zhou, J. Gu, T. Y. Chiang, F. Xiang, and H. Su, "Point-sam: Promptable 3d segmentation model for point clouds," in *Proceedings of the International Conference on Learning Representations 2025*, 2025.

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

[13] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[14] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, 2010.

[15] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019.

[16] H. Caesar, J. Uijlings, and V. Ferrari, "Coco-stuff: Thing and stuff classes in context," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[17] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.

[18] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, 2010.

[19] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European Conference on Computer Vision*, Springer, 2012.

[20] Q. Lu, J. Kuen, S. Tiancheng, G. Jiuxiang, G. Weidong, J. Jiaya, L. Zhe, and Y. Ming-Hsuan, "High-quality entity segmentation," in *ICCV*, 2023.

[21] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[22] M. Yan, J. Zhang, Y. Zhu, and H. Wang, "Maskclustering: View consensus based mask graph clustering for open-vocabulary 3d instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28274–28284, 2024.

[23] Y. Yang, X. Wu, T. He, H. Zhao, and X. Liu, "Sam3d: Segment anything in 3d scenes," *arXiv preprint arXiv:2306.03908*, 2023.

[24] S. Lu, H. Chang, E. P. Jing, A. Boularias, and K. Bekris, "Ovir-3d: Open-vocabulary 3d instance retrieval without training on 3d data," in *Conference on Robot Learning*, pp. 1610–1620, PMLR, 2023.

[25] X. Zhu, R. Zhang, B. He, Z. Guo, Z. Zeng, Z. Qin, S. Zhang, and P. Gao, "Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2639–2650, 2023.

[26] P. Nguyen, T. D. Ngo, E. Kalogerakis, C. Gan, A. Tran, C. Pham, and K. Nguyen, "Open3dis: Open-vocabulary 3d instance segmentation with 2d mask guidance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4018–4028, 2024.

[27] 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*, pp. 8748–8763, PmLR, 2021.

[28] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," *arXiv preprint arXiv:2203.03605*, 2022.

[29] R. Bommasani and et al., "On the opportunities and risks of foundation models," *CoRR*, 2021.

[30] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, "Dust3r: Geometric 3d vision made easy," in *CVPR*, 2024.

[31] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, "3d foundation models enable simultaneous geometry and pose estimation of grasped objects," *arXiv preprint arXiv:2407.10331*, 2024.

[32] H. Tang, T. Zhang, M. Johnson-Roberson, and W. Zhi, "Bi-manual joint camera calibration and scene representation," *arXiv preprint arXiv:2505.24819*, 2025.

[33] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, 2023.

[34] T. Zhang, K. Huang, W. Zhi, and M. Johnson-Roberson, "Darkgs: Learning neural illumination and 3d gaussians relighting for robotic exploration in the dark," 2024.

[35] T. Zhang, W. Zhi, K. Huang, J. Mangelson, C. Barbalata, and M. Johnson-Roberson, "Recgs: Removing water caustic with recurrent gaussian splatting," *arXiv preprint arXiv:2407.10318*, 2024.

[36] J. Edstedt, Q. Sun, G. Bökman, M. Wadenbäck, and M. Felsberg, "Roma: Robust dense feature matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19790–19800, 2024.

[37] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, "Unifying representation and calibration with 3d foundation models," *IEEE Robotics and Automation Letters*, 2024.

[38] A. Ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.