International Journal of Computer Vision (2022) 130:2940–2961
https://doi.org/10.1007/s11263-022-01670-0

Scene Reconstruction with Functional Objects for Robot Autonomy

Muzhi Han[1] · Zeyu Zhang[1,2] · Ziyuan Jiao[1,2] · Xu Xie[1] · Yixin Zhu[3] · Song-Chun Zhu[2,3] · Hangxin Liu[2]

Received: 19 February 2021 / Accepted: 9 August 2022 / Published online: 20 September 2022

Abstract
In this paper, we rethink the problem of scene reconstruction from an embodied agent's perspective: While the classic view focuses on the reconstruction accuracy, our new perspective emphasizes the underlying functions and constraints of the reconstructed scenes that provide *actionable* information for simulating *interactions* with agents. Here, we address this challenging problem by reconstructing a *functionally equivalent* and interactive scene from RGB-D data streams, where the objects within are segmented by a dedicated 3D volumetric panoptic mapping module and subsequently replaced by part-based articulated CAD models to afford finer-grained robot interactions. The object functionality and contextual relations are further organized by a graph-based scene representation that can be readily incorporated into robots' action specifications and task definition, facilitating their long-term task and motion planning in the scenes. In the experiments, we demonstrate that (i) our panoptic mapping module outperforms previous state-of-the-art methods in recognizing and segmenting scene entities, (ii) the geometric and physical reasoning procedure matches, aligns, and replaces object meshes with best-fitted CAD models, and (iii) the reconstructed functionally equivalent and interactive scenes are physically plausible and naturally afford actionable interactions; without any manual labeling, they are seamlessly imported to ROS-based robot simulators and VR environments for simulating complex robot interactions.

Keywords Functional scene representation · 3D scene reconstruction · Actionable information · Volumetric panoptic mapping · Physical reasoning · Robot interaction

1 Introduction

Perception of human-made environments and the objects within inevitably leads to the course of actions (Gibson, 1950, 1966), which naturally forms the basis for a human agent to interact with the environment and accomplish complex tasks. Crucially, what we "see" is much more than pixels and semantic labels (Knill and Richards, 1996). Instead, we further "see" *how* to interact with them for our task purposes. Likewise, an embodied AI agent or a robot must possess a similar perceptual capability to achieve a wide range of task goals in the physical world. However, this critical perspective is mostly unexplored by prior scene reconstruction literature in computer vision or Simultaneous Localization And Mapping (SLAM) methods in robotics. Oftentimes, prior arts only capture scenes' occupancy information and are evaluated pri-

Communicated by Akihiro Sugimoto.

Muzhi Han and Zeyu Zhang have contributed equally to this work.

✉ Yixin Zhu
yixin.zhu@pku.edu.cn

✉ Hangxin Liu
liuhx@bigai.ai

Muzhi Han
muzhihan@ucla.edu

Zeyu Zhang
zeyuzhang@ucla.edu

Ziyuan Jiao
zyjiao@ucla.edu

Xu Xie
xiexu@ucla.edu

Song-Chun Zhu
sczhu@bigai.ai

[1] UCLA Center for Vision, Cognition, Learning, and Autonomy, Los Angeles, USA

[2] Beijing Institute for General Artificial Intelligence (BIGAI), Beijing, China

[3] Institute for Artificial Intelligence, Peking University, Beijing, China

marily by reconstruction accuracy in the euclidean space. Without incorporating the *actionable* information—actions a semantic entity could afford and the associated physical constraints among entities—in a reconstructed scene, a robot can only perform relatively simple navigation or pick-and-place tasks, hindering its capability of planning and executing tasks in a long horizon.

Having the *actionable* information in a scene is crucial for the training and testing of modern embodied AI agents (Batra et al., 2020). Existing research efforts are mainly devoted to develop simulation platforms that provide (i) photorealistic views [e.g., Habitat (Savva et al., 2019), RoboTHOR (Deitke et al., 2020)] for navigation, (ii) articulated and interactive objects [e.g., iGibson (Xia et al., 2020), SAPIEN (Xiang et al., 2020)] for interaction, and (iii) physical simulation engines [e.g.VRGym (Xie et al., 2019)] for fine-grained fluent changes. While the *actionable* information can be explicitly specified and embedded in the simulation setup, or be recognized from a physical scene using dedicated vision modules, such as part-based object pose estimation (Li et al., 2020), functionality (Zhao and Zhu, 2013) and affordance (Min et al., 2016) recognition it is non-trivial to organize this information and unclear about how an agent could utilize such information for various tasks.

Take the scene in Fig. 1 as the example, wherein the robot is tasked to pick up a frozen meal from the fridge, microwave it, and serve it. The challenges of processing *actionable* information are three-fold. First, it needs to recognize the semantics and geometry information of objects (e.g., this piece of point cloud is a fridge). Although typical semantic mapping and segmentation techniques can achieve this goal (Hoang et al., 2020; Narita et al., 2019), a more robust and accurate approach is still in need to better handle the complexity in clustered real environments given a first-person-view RGB-D video stream. Second, mere semantics are inadequate to reflect the actions an object affords (e.g., whether or how the fridge can be opened). While some existing work attempted to identify the associations between symbolic actions and objects (Myers et al., 2015; Li et al., 2019) or the underlying object's kinematics (Sturm et al., 2011; Chang and Demiris, 2017; Martín-Martín and Brock, 2019), they are insufficient for robots to execute complex tasks with multiple steps at the motion level. Third, we quest for a more fundamental question: How to devise a scene representation with a succinct action specification and task definition to account for the action opportunities and the accumulated outcome of executed actions? Without addressing these challenges, a robot can hardly plan for the given task or verify whether its plan is valid before executing in the physical world.

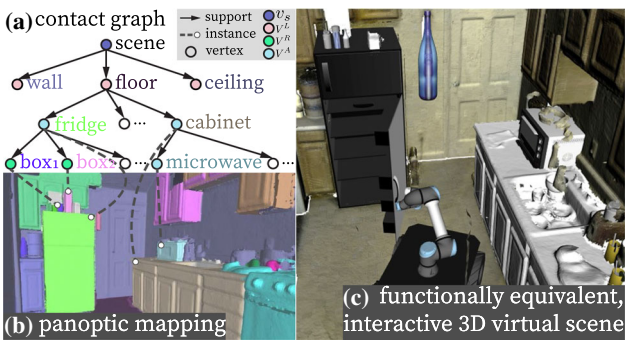


Fig. 1 The reconstruction of a functionally equivalent, interactive 3D scene. **(a)** A contact graph is constructed by the supporting relations that emerged from **(b)** panoptic mapping. By reasoning their affordance, functional objects within the scene are matched and aligned with part-based interactive CAD models. **(c)** The reconstructed functionally equivalent scene enables a robot to simulate its task execution with comparable outcomes in the physical world

In this paper, we propose a new task of reconstructing *functionally equivalent* and interactive scenes by representing the *actionable* information of scene entities to support agents' planning and simulation. Here we argue that a scene's functionality is composed by the functions of objects within the scene. Therefore, the essence of a *functionally equivalent* scene is to preserve most objects' four characteristics with a decreasing propriety: (i) their semantic class and spatial relations with nearby objects, (ii) their affordance, e.g., what interactions they offer, (iii) similar geometry in terms of size and shape, and (iv) similar appearance. To address this new task, we devise a robot perception system with three unique components; see an illustration in Fig. 2.

(A) *A robust 3D volumetric panoptic mapping module*, detailed in Sect. 3, accurately segments and reconstructs 3D objects and layouts in clustered scenes based on potentially noisy per-frame segmentation. The term "panoptic," introduced in Kirillov et al. (2019), refers to jointly segmenting *stuff* and *things* in semantic and instance levels. In this paper, we regard objects as *things* and layouts as *stuff*. This module produces a volumetric panoptic map using a novel per-frame panoptic fusion strategy and a global data fusion procedure performing data association, map integration, and map regularization; see Figs. 1b and 2a for examples of results.

(B) *A physical reasoning module*, detailed in Sect. 4, replaces the potentially noisy and incomplete object meshes segmented from the panoptic map with functional (rigid or articulated) CAD models. This step is achieved by a ranking-based CAD matching and an optimization-based CAD alignment, which accounts for both geometric and physical constraints. We further introduce a global physical violation check to ensure that the resulting reconstructed interactive scene is physically plausible.

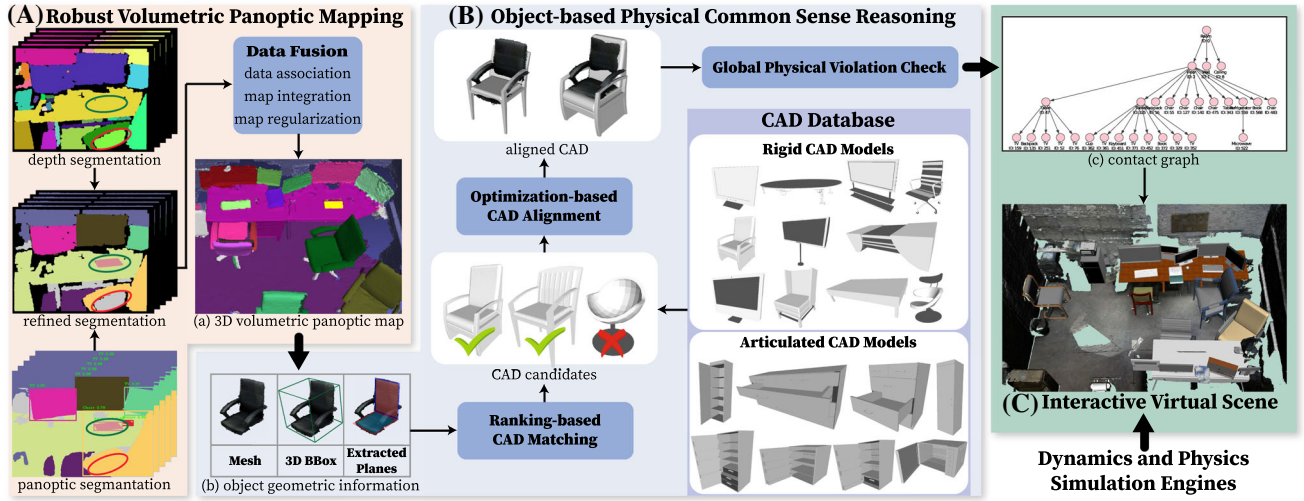


Fig. 2 System architecture for reconstructing a functionally equivalent scene. (**A**) Per-frame segmentation and global data fusion produce (a) a 3D volumetric panoptic map with fine-grained semantics and geometry, served as the input for (**B**) physical common sense reasoning that matches, aligns, and replaces segmented object meshes with functionally equivalent CAD alternatives. Specifically, (b) by geometric similarity, a ranking-based matching algorithm selects a shortlist of CAD candidates, followed by an optimization-based process that finds a proper transformation and scaling between the CAD candidates and object mesh. A global physical violation check is further applied to finalize CAD replacements to ensure physical plausibility. (**C**) This CAD augmented scene can be seamlessly imported to existing simulators; (c) contact graph encodes the kinematic relations among scene entities in a scene and reflects the planning space for a robot

(C) *A contact graph cg representation*, detailed in Sect. 2 and illustrated in Fig. 3, is constructed in accordance with the supporting and proximal relations among objects and imposes physical constraints as well as kinematic information for a robot's task execution. After retrieving *actionable* information annotated in CAD models, this novel representation indicates how an object can be moved or manipulated (e.g., a table can be moved in 3D space) and how nearby objects would move correspondingly (e.g., a box on the table would go through a similar transformation if not slid or tilted). The *cg* can be interpreted as and converted to a kinematic tree, which is updated following the robot's actions so that it can support long-horizon task and motion planning. As such, it serves as an ideal representation that bridges robot perception (scene reconstruction) with robot planning.
Part of this work is published in Han et al. (2021); comparing with it, this paper highlights the conversion from a sensed *cg* to the (URDF), conducts experiments and analysis in real-world setting, and further evaluations including a new study of evaluating resulted *cg* using (GED).

1.1 Related Work

Scene datasets are crucial for providing supervisions for existing data-driven methods in a plethora of scene reconstruction and scene understanding tasks. In literature, the development of such datasets follows three stages. Early work, such as NYU-Depth (Silberman et al., 2012) and SUN RGB-D (Song et al., 2015), provides single view RGB-D images with densely annotated object segmentation, bounding boxes, etc. These types of 2.5D data are primarily designed to support recognition and prediction tasks in computer vision. In the second stage, datasets provide full 3D (in contrast to 2.5D) scene data in the form of annotated meshes for more holistic computer vision tasks (Hua et al., 2016; Chang et al., 2017; Dai et al., 2017). More recently, researchers start to construct synthetic scene datasets (Yu et al., 2011; Song et al., 2017; Qi et al., 2018; Jiang et al., 2018) to overcome the tedious and error-prone labeling process and obtain scene data at a much larger scale. Despite success in all three stages, they still fall short for robot learning or planning due to the lack of proper means that converts a scanned or synthetic scene to an interactive one for robot task execution. In comparison, the proposed system can reconstruct interactive scenes from RGB-D streams and directly import them into simulators for training and testing of robots' complex task execution.

To gather the scene semantics, modern **semantic mapping** (Narita et al., 2019; Grinvald et al., 2019; Pham et al., 2019a) and **object SLAM** (Yang and Scherer, 2019a; McCormac et al., 2018) methods can retrieve object semantic segmentation, 6 DoF poses, and 3D bounding boxes during reconstruction. Physical cues, such as support and collision (Yang and Scherer, 2019b; Wada et al., 2020; Sui et al., 2020) and robot proactive actions (Xu et al., 2015; Liu et al., 2018b), can be further integrated to better estimate

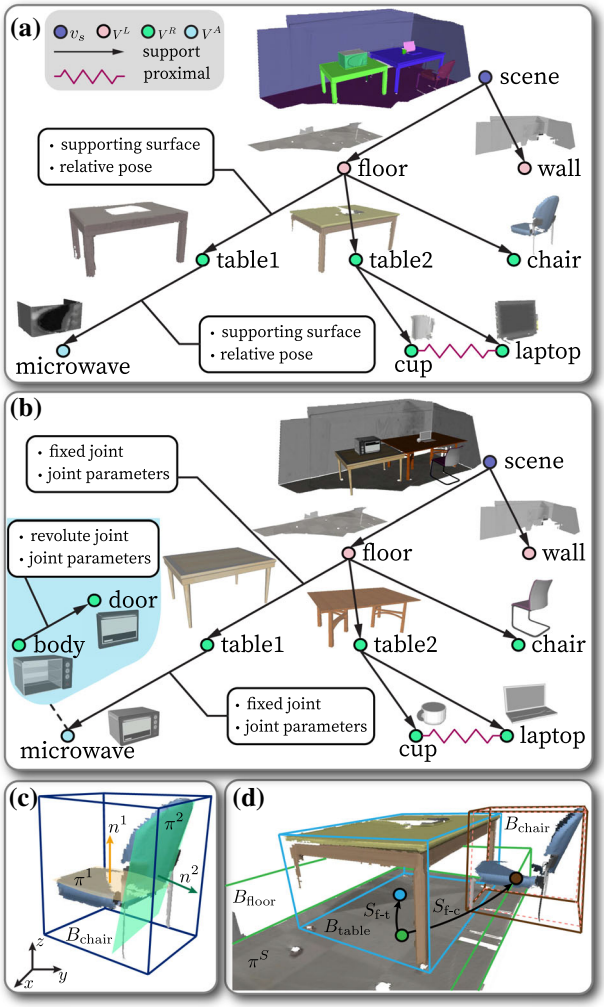


Fig. 3 3D scene representations and relations within. (**a**) The contact graph representation. Each node denotes an object or a piece of layout, reconstructed and segmented as meshes from the RGB-D stream using the proposed panoptic mapping module. The directed edges indicate supporting relations—The parent node supports the child node. (**b**) The object meshes are replaced by best-fitted CAD models to create a functionally equivalent and physically plausible reconstructed scene. The directed edges and the constructed kinematic relations define the action space for robot planning. By updating the kinematic relations, various action effects can be easily integrated. (**c**) The supporting relations can further facilitate a reasoning process that refines (**d**) the 3D bounding box estimation. Initial: dashed line. Refined: solid line

and refine the scene semantics. In parallel, significant efforts have been made for object instance segmentation from point clouds (Zhang et al., 2019); e.g., Yi et al. (2019) can segment an object with fine-grained part instances, and Pham et al. (2019b) jointly perform semantic and instance segmentation. The above work, however, could only produce incomplete objects (in contrast to full 3D) due to confined viewpoints in the physical world, which prohibits the complex robot interaction and task execution in the reconstructed scenes. To alleviate this issue, researchers have recently attempted to **align CAD models** to these incomplete objects based on a single RGB image (Huang et al., 2018b; Chen et al., 2019), a single RGB-D image pair (Gupta et al., 2015; Zou et al., 2019), and scanned scene meshes (Dai et al., 2017; Avetisyan et al., 2019a, b) to incorporate richer scene semantics. Following this trend, our system further introduces a physical reasoning procedure to align (part-based) CAD models to segmented objects to enable robot manipulation and interaction.

Devising an appropriate **scene representation** for scene reconstruction remains an open problem (Cadena et al., 2016). Existing SLAM and semantic mapping approaches reviewed above oftentimes represent a reconstructed scene and its entities as sparse landmarks (Pronobis and Jensfelt, 2012; Yang and Scherer, 2019a), surfels (McCormac et al., 2017; Hoang et al., 2020), volumetric voxels (Grinvald et al., 2019; McCormac et al., 2018), or semantic objects (Yang and Scherer, 2019a; McCormac et al., 2018). Such a paradigm only provides geo-information of what and where to a robot without any actionable information for its interactions or planning. Meanwhile, graph-based representations for 3D scene further identify the hierarchical and relational structure among the scene entities (Zhu and Mumford, 2007; Zhao and Zhu, 2011, 2013; Zheng et al., 2015; Huang et al., 2018a; Jiang et al., 2018; Chen et al., 2019; Armeni et al., 2019; Wald et al., 2020; Rosinol et al., 2020), providing better structural and contextual information of the reconstructed scenes. In particular, Rosinol et al. (2020) explicitly incorporate actionable information to support robot planning, though limited to navigation and traversal tasks as the representation only models the connectivity between entity nodes. By leveraging the advantages of prior arts and addressing the shortcomings, the proposed system takes a real RGB-D stream as input and produces a contact graph representation based on the identified supporting relations among scene entities. This representation for scene reconstruction indicates how an entity can be interacted with and what the effect would be after an interaction, capable of supporting more complex manipulation planning.

1.2 Contributions

To our knowledge, ours is the first work that introduces a comprehensive system that reconstructs a full 3D scene from an embodied agent's perspective to provide *actionable* information for simulating robot interactions. It makes three major contributions:

1. We introduce a novel scene representation, contact graph, whose structure is determined by the supporting and proximal relations among scene entities. It imposes physical constraints for a physically plausible scene and kinematic information that indicates whether and how an object can be interacted with. This contact graph representation is

constructed and maintained for the scene reconstruction, and converted to a kinematic tree, which reflects the full geometric state of a scene and can update to keep track of every interaction. As such, our contact graph representation can facilitate the functionally equivalent scene reconstruction, as well as the robot learning and planning for complex long-horizon tasks.

2. Leveraging (i) local geometric similarity based on relative sizes and extracted surface planes of objects, and (ii) global physical constraints regarding the plausibility of stable support and non-penetration, we align rigid or articulated CAD models to object meshes to generate a physically plausible, fully interactive scene.
3. We develop a volumetric panoptic mapping module based on Grinvald et al. (2019) and introduce new designs to improve the accuracy in per-frame segmentation and the consistency in global data fusion. We show that this implementation is more robust against noisy input data and generates more accurate panoptic segmentation results, especially suitable for challenging and clustered indoor scenes.

2 Contact-Based Scene Representation

We devise a graph-based representation, *contact graph* cg, to represent a 3D indoor scene and the relations among scene entities. Formally, a contact graph $cg = (pt, E)$ contains (i) a parse tree (pt) that hierarchically organizes the scene entities (Zhu and Mumford, 2007), and (ii) the proximal relations E among entities represented by undirected edges; see an example in Fig. 3a.

2.1 Representation

Scene parse tree $pt = (V, S)$ has been used to represent the hierarchical decompositional relations (i.e., the edge set S) among entities (i.e., the node set V) in various task domains, including 2D images and 3D scenes (Zhu and Mumford, 2007; Zhao and Zhu, 2011, 2013; Qi et al., 2018; Jiang et al., 2018; Huang et al., 2018b, a; Chen et al., 2019), videos and activities (Zhu et al., 2015, 2016; Qi et al., 2020; Jia et al., 2020), robot manipulations (Edmonds et al., 2017; Liu et al., 2018a; Edmonds et al., 2019; Liu et al., 2019; Zhang et al., 2020), and theory of mind (Yuan et al., 2020). In this paper, we adapt pt to represent supporting relations among entities instead of their decomposition. A pt is dynamically built and maintained during the reconstruction based on the identified supporting relations among segmented scene entities; for instance in Fig. 3a, the `table1` is the parent node of the `microwave`. Supporting relation is quintessential in scene understanding as it reflects the omnipresent physical plausibility; i.e., if the table were moved, the microwave would move together with it. This perspective of physical common sense goes beyond occupancy information (i.e., the geometric location of an object); in effect, it further provides actionable information and the potential outcome of actions for robot interactions and task execution in the scene.

Scene entity nodes $V = \{v_s\} \cup V^L \cup V^R \cup V^A$ include: (i) the scene node v_s, serving as the root of pt, (ii) layout node set V^L, including floor, ceiling, and the walls that bound the 3D scene, (iii) rigid object set V^R, wherein each object has no articulated part (e.g., a table), and (iv) articulated object set V^A, wherein each object has articulated parts to be interacted for robot tasks (e.g., fridge, microwave). Each non-root node $v_i = \langle o_i, c_i, M_i, B_i(\boldsymbol{p}_i, \boldsymbol{q}_i, \boldsymbol{s}_i), \Pi_i \rangle$ encodes a unique instance label o_i, a semantic label c_i, a full geometry model M_i (e.g., a triangle mesh or a CAD model), a 3D bounding box B_i (parameterized by its center position $\boldsymbol{p}_i$, orientation $\boldsymbol{q}_i$, and size $\boldsymbol{s}_i$, all in $\mathbb{R}^3$), and a set of surface planes $\Pi_i = \{\boldsymbol{\pi}_i^k, k = 1 \cdots |\Pi_i|\}$, where a plane $\boldsymbol{\pi}_i^k$ is represented by a homogeneous vector $[\boldsymbol{n}_i^{k\,T}, d_i^k]^T \in \mathbb{R}^4$ in the projective space (Hartley and Zisserman, 2003) with unit plane normal vector $\boldsymbol{n}_i^k$, where any point $\boldsymbol{v} \in \mathbb{R}^3$ on the plane satisfies a constraint: $\boldsymbol{n}_i^{k\,T} \cdot \boldsymbol{v} + d_i^k = 0$; see Fig. 3c for an illustration. Compared to other geometric primitives like generalized cylinders (Agin and Binford, 1973), planes are advantageous in that they can be extracted robustly from corrupted object meshes and are effective features in downstream computations.

Supporting relations S is the set of directed edges in pt from parent nodes to their child nodes. Each edge $s_{p,c} \in S$ imposes physical common sense between the parent node v_p and the child node v_c. These constraints are necessary to ensure that v_p supports v_c in a physically plausible fashion:

(1) *Geometrical plausibility.* The parent node v_p should have a plane $\boldsymbol{\pi}_p^s = [\boldsymbol{n}_p^{s\,T}, d_p^s]^T$ that is horizontal and is in contact with the bottom surface of the child v_c:

$$\begin{aligned} &\exists \boldsymbol{\pi}_p^s \in \Pi_p, \boldsymbol{n}_p^{s\,T} \cdot \boldsymbol{g} \leqslant a_{th}, \\ &s.t.\ \mathcal{D}(v_c, \boldsymbol{\pi}_p^s) = p_c^g - (-d_p^s + s_c^g/2) = 0, \end{aligned} \tag{1}$$

where $\boldsymbol{g}$ is a unit vector along the gravity direction, $a_{th} = -0.9$ is a tolerance coefficient ($a_{th} = -1$ for a perfect horizontal plane), and p_c^g and s_c^g denote the position and size of v_c's 3D bounding box along the gravity direction, respectively.

(2) *Sufficient contact area for stable support.* Formally,

$$\mathcal{A}(v_p, v_c) = \mathrm{A}(v_p \cap v_c)/\mathrm{A}(v_c) \geq b_{th}, \tag{2}$$

where $\mathrm{A}(v_c)$ is the bottom surface of v_c's 3D bounding box, and $\mathrm{A}(v_p \cap v_c)$ is the area of the overlapping rectangle containing the mesh vertices of v_p near $\boldsymbol{\pi}_p^s$ within v_c's 3D bounding box. We set threshold $b_{th} = 0.5$ for a stable support.

Proximal relations E introduce links among entities in the pt. It imposes additional constraints by modeling spatial relations between two non-supporting but physically nearby objects v_1 and v_2: Their meshes should not penetrate with each other, i.e., $\text{Vol}(M_1 \cap M_2) = 0$. Note that we only assign a proximal relation between two objects with overlapping 3D bounding boxes, i.e., when $\text{Vol}(B_1 \cap B_2) > 0$, instead of between every pair of objects to reduce computation cost. The non-penetration constraints will be applied when selecting physically plausible scene configurations, as detailed in Sect. 4.4.

2.2 Constructing Contact Graphs

For each scene entity x extracted from the volumetric panoptic map (see details on obtaining panoptic map in Sect. 3.4), we initialize a scene entity node v_x of cg by: (i) acquiring its o_x, c_x, M_x from the panoptic map, (ii) estimating a gravity-aligned, minimal 3D bounding box $B_x(\boldsymbol{p}_x, \boldsymbol{q}_x, \boldsymbol{s}_x)$ based on M_x using the method in Malandain and Boissonnat (2002), and (iii) detecting a set of surface plane Π_x on M_x by iteratively applying RANSAC (Taguchi et al., 2013) and removing plane inliers. We further classify each initialized scene entity node v_x as a layout node, a rigid object node, or an articulated object node based on its semantic class c_x.

Given a set of scene entity nodes initialized on the fly, we apply a bottom-up process to build up the structure of cg by estimating supporting relations among the entities. Specifically, for each node v_c, we find a parent node v_p with a supporting plane $\boldsymbol{\pi}_p^s$ that best satisfies the constraints described in Eqs. (1) and (2). We consider all nodes $\{v_i\}$ whose bottom planes are spatially below the 3D bounding box of v_c as candidates of v_p, and acquire their gravity-opposed surface planes $\{\boldsymbol{\pi}_i^k\}$ as potential supporting planes. Then the most likely supporting relation is determined by maximizing the following score function:

$$S(v_c, v_i, \boldsymbol{\pi}_i^k) = \left\{1 - \min\left[1, \|\mathcal{D}(v_c, \boldsymbol{\pi}_i^k)\|\right]\right\} \times \mathcal{A}(v_i, v_c), \tag{3}$$

where the first term indicates the alignment between v_c's bottom surface bottom surface and the supporting plane, and the second term reflects an effective supporting area, both normalized to [0, 1]. We may also uncover an invisible supporting plane (e.g., a fully occluded tabletop). When v_c is well-overlapped with v_i but v_i has no valid supporting plane, the bottom plane of v_c will be registered as a new supporting plane of v_i. This advantage is, however, hard to guarantee at all time due to the complexity of real-world scenarios. Finally, we construct cg and assign the attributes for each supporting edge based on the estimated supporting relations.

We further refine the 3D bounding box B_i of each scene entity node v_i such that Eq. (1) is strictly satisfied and the cg is feasible. This step also compensates for the error of extracting geometric features directly from an incomplete reconstructed mesh. Fig. 3d illustrates an example of the refinement process. The reconstructed scene only produces a partial mesh of the chair; its legs are captured incompletely. Consequently, its 3D bounding box (in dashed line) only encloses the detected portion of the chair floating in the air. By determining the supporting relation between the floor and the chair, our system automatically extends the bounding box (in solid line) to the supporting plane on the floor, and thus reconstructs a physically plausible scene. In experiments, we also quantitatively evaluate this refinement process; see the result in Table 4. As the last step of cg construction, we determine the proximal relations by comparing pairwise 3D bounding boxes of scene entities.

2.3 Interpreting a Contact Graph

As shown in Fig. 3a and described above, a cg hierarchically organizes segmented scene entities with corresponding semantics, meshes, and extracted geometric features. To convey richer *actionable* information, we convert the cg to a functionally equivalent cg' by maintaining the overall graph structure and replacing each object mesh with a CAD model while preserving its semantic class, instance label, relative dimensions, and surface planes; see Fig. 3b.

The functionally equivalent cg' with CAD models naturally encodes the full (detected) geometry state of the scene. It can be interpreted as a kinematic tree, where nodes represent links, and edges represent joints connecting two links with assumed joint type, range, and joint value. Depending on the semantic class, individual objects may be replaced by articulated CAD models. For instance, the CAD model for the microwave in Fig. 3b consists of two parts, the body and the door, connected by a revolute joint. The cg' (the kinematic tree) is an ideal representation to support robot planning; its joint specifications reflect the possible ways a robot can change environment states and naturally define the task goal for a robot to achieve. Although the knowledge of the object structure is injected when designing the CAD model and is not likely to match with the real one strictly, it nevertheless provides an approximation for most of the possible actions an agent can take and what the actions like, sufficient for the agent's long-term planning.

3 Robust Panoptic Mapping

Robust and accurate mapping of scene entities and segmenting them from clustered environments are essential for constructing a cg and serving our downstream tasks. We

develop a robust 3D panoptic mapping module to generate object and layout segments in the form of meshes from RGB-D streams; see the pipeline in Fig. 2A. Based on the architecture of Voxblox++ (Grinvald et al., 2019), our mapping module incorporates crucial modifications to improve the robustness of mapping against noisy and inconsistent segmentation at each frame.

Voxblox++ builds a volumetric object-centric semantic map by (i) generating per-frame segments in point cloud form by combining RGB-based instance segmentation and depth-based geometric segmentation, and (ii) associating the segments across different frames and integrating them into a Truncated Signed Distance Field (TSDF)-based object-level global map. Each per-frame segment is obtained by assigning a semantic label and an instance label produced by instance segmentation to a geometric segment produced by geometric segmentation. Assuming that segments computed using geometry cues are consistent across different frames, Voxblox++ associates those per-frame segments from different views with global map segments by their 3D overlapping ratio and integrates them into the global map, while recording the history of predicted semantic and instance labels for each global map segment.

However, we observe two major limitations of Voxblox++. First, the generated per-frame segments may not preserve all predicted instances and some segments of far-away background may be labeled as foreground objects, hindering the mapping performance. We design two extra steps to handle this limitation, as detailed in Sect. 3.1. Second, Voxblox++ separately tracks semantic and instance labels in data association and map integration processes, making it less coherent when identifying instance and recognizing semantics for the same global map segment. Our solution is to jointly account for semantic and instance labels throughout the procedure to build a more consistent global map. We describe our implementation of this strategy in data association (Sect. 3.2), map integration and regularization (Sect. 3.3), and scene entity extraction (Sect. 3.4).

3.1 Per-Frame Segmentation and Fusion

Following Voxblox++ (Grinvald et al., 2019), we perform RGB-based panoptic segmentation and depth-based geometric segmentation for each frame and then combine the two sets of segments. Given a RGB-D image as the input, we use an off-the-shelf panoptic segmentation tool provided by Detectron2 (Wu et al., 2019) to produce panoptic segments in RGB domain. A convexity-based depth segmentation approach (Furrer et al., 2018) can segment the corresponding depth image following geometric boundaries. We denote each predicted 2D panoptic segment as M_i with semantic label c_i and instance label o_i (whereas each stuff class has only one instance label) and each 3D geometric segment (in point cloud) as G_j. Then the goal is to fuse the segmentations from two sources to generate per-frame point cloud segments $\{(P_k, c_k, o_k)\}$, which preserve the predicted geometric and semantic information.

Voxblox++ generates $\{(P_k, c_k, o_k)\}$ by assigning semantic and instance labels to geometric segments $\{G_j\}$ greedily based on the 2D overlap between the 2D projection of each G_j and $\{M_i\}$ on the image coordinate. In practice, this strategy leads to two drawbacks. The first one is that predicted instances will be ignored if they are not recognized geometrically in depth images. Figure 2A shows an example; the missing keyboard marked by a green circle in depth segmentation would be discarded by Voxblox++. We instead split a geometric segment G_j to extract the point cloud corresponding to a panoptic segment M_i if the 2D projection of G_j fully contains M_i when aligned. Then we assign semantic and instance labels for all G_j as well as the extracted point cloud segments as Grinvald et al. (2019) do to get $\{(P_k, c_k, o_k)\}$. Second, an inaccurately segmented object in RGB image may consist of far-away geometric segments in depth, e.g., the floor marked by a red circle is regarded as part of the chair in the panoptic segmentation in Fig. 2A. Our modification addresses this issue by adding an extra step of Euclidean clustering. We compute pairwise Euclidean distances among all geometric segments that belong to the same object instance and apply Euclidean clustering to obtain clusters of segments. Then we retrieve the largest cluster defined as having the largest total number of points in its segments and keep the segments within as part of the instance. The rest of segments are regarded as outliers and assigned to the background.

The above implementation relies on some defined heuristics that could limit the generalizability of our panoptic segmentation approach; one direction to overcome this limitation is to introduce data-driven methods, which is beyond the scope of the paper. Nevertheless, the two proposed steps are useful practices that significantly improve the per-frame segmentation. As an example shown in Fig. 2a, our method (i) correctly segments the keyboard and divides the two monitors when they are geometrically under-segmented, (ii) obtains geometrically refined panoptic segmentation of the table, chair, and floor, and (iii) excludes the far-away ground from the segmentation of the chair.

3.2 Data Association

We associate each per-frame point cloud segment to a global 3D segment (or global segment for short) in the global map, while associating its panoptic prediction with a global panoptic entity. Note that the global segments and panoptic entities are maintained and updated throughout the entire mapping process. Following Voxblox++ (Grinvald et al., 2019), we first draw the correspondence between per-frame segments and global segments greedily based on their 3D overlaps

given the camera pose. We denote that each global segment is indexed with a unique segment label $l \in \mathbb{L}$.

For each per-frame segment (P_k, c_k, o_k) associated with a global segment l_i, we aim to find its associated global instance label p_m by looking at the past panoptic predictions of segment l_i. We introduce a triple-wise count $\Phi(l, c, p)$ over a segment label l, a semantic label c, and an instance label p in the global map to jointly track the semantic and instance predictions. This is inspired by the observation that the prediction of instances and their semantic labels are inter-dependent in typical object detection and segmentation algorithms (Ren et al., 2016; He et al., 2017). Specifically, p_m is assigned with the instance label p that maximizes the count $\Phi(l_i, c_k, p) > 0$. When $\sum_p \Phi(l_i, c_k, p) = 0$, we assign a new global instance label $p_m = p_{new}$.

3.3 Map Integration and Regularization

We integrate per-frame segments into the 3D volumetric panoptic map by (i) integrating the segments into a TSDF volume (Oleynikova et al., 2017) with each TSDF voxel labeled with a global segment label l, and (ii) recording the associated panoptic entities. For any per-frame segment associated with (l_i, c_k, p_m), we increase the triple-wise count:

$$\Phi(l_i, c_k, p_m) = \Phi(l_i, c_k, p_m) + 1 \tag{4}$$

We also introduce a two-stage process to regularize the map by merging global segment labels and instance labels. Specifically, we first merge global segment labels pairwise if they share voxels over a certain ratio (Grinvald et al., 2019). Next, we merge two global instance labels $p_1, p_2 \in \mathbb{P}$ with the same semantic class $c \in \mathbb{C}$ if the duration of association with common segment labels exceeds a threshold:

$$\sum_{l \in \mathbb{L}_\cap} [\Phi(l, c, p_1) + \Phi(l, c, p_2)] \geq m_{th} \cdot \sum_{l \in \mathbb{L}} [\Phi(l, c, p_1) + \Phi(l, c, p_2)], \tag{5}$$

where $\mathbb{L}_\cap = \{l \in \mathbb{L} | \Phi(l, c, p_1) > 0, \Phi(l, c, p_2) > 0\}$. This step merges incorrectly split instances, which can be introduced by the overcautious filtering step when generating per-frame point cloud segments. We note that this map regularization process can be regarded as a delayed data association that corrects potentially wrong association of global segments and instances. It helps improve the consistency and scalability of the global map; i.e., it reduces the map size.

3.4 Panoptic Entities Extraction

After the above mapping process, we extract the panoptic entities (i.e., objects and layouts) from the global map as triangle meshes. For each global segment l, its semantic class $\hat{c}_l$ and global instance label $\hat{p}_l$ are determined following a greedy strategy:

$$\begin{aligned} \hat{c}_l &= \arg\max_{c \in \mathbb{C}} \sum_{p \in \mathbb{P}} \Phi(l, c, p), \\ \hat{p}_l &= \arg\max_{p \in \mathbb{P}} \Phi(l, \hat{c}_l, p). \end{aligned} \tag{6}$$

For each global instance label $p \in \mathbb{P}$, we group all global segments in the map with labels in the set $L_p = \{l \in \mathbb{L} | \hat{p}_l = p\}$ and extract the corresponding TSDF volume, from which a mesh is created. In a nutshell, our system outputs a set of scene entities in the form of triangle meshes with their instance labels and semantic labels.

4 Scene Reconstruction with CAD Replacement

Due to occlusion or limited camera angle, the reconstructed scene and the segment meshes are oftentimes incomplete and non-interactive before being recovered as full 3D models; Figs. 5a and 6a show some examples of incomplete meshes. We introduce a multi-stage framework to replace a segmented object mesh with a CAD model through (i) an object-level CAD matching, (ii) pose alignment of the CAD model, and (iii) a scene-level, global physical violation check; see Fig. 2B for an illustration of the framework.

4.1 CAD Pre-processing

We collect a CAD database consisting of both rigid and articulated CAD models, organized by semantic classes. The rigid CAD models are obtained from ShapeNetSem (Chang et al., 2015), whereas articulated ones are first assembled and then properly transformed into one model. Each CAD model is transformed to have its origin and axes aligned with its canonical coordinates. Figure 2B shows some instances of CAD models in the database, and Fig. 4 highlights some articulated CAD examples with coordinate frames on the articulated parts. All the objects can be uniformly scaled while persevering transformation and kinematic information for the subsequent matching and alignment. Similar to a segmented scene entity x, a CAD model y is parameterized by o_y, c_y, M_y, while we further extract its $B_y(\boldsymbol{p}_y, \boldsymbol{q}_y, \boldsymbol{s}_y)$, and Π_y.

4.2 Ranking-Based CAD Matching

Take the chair in Fig. 2b as an example: Given a segmented object entity x, the algorithm retrieves all CAD models in the same semantic category (i.e., chair) from the CAD database to best fit x's geometric information. Since the exact orienta-

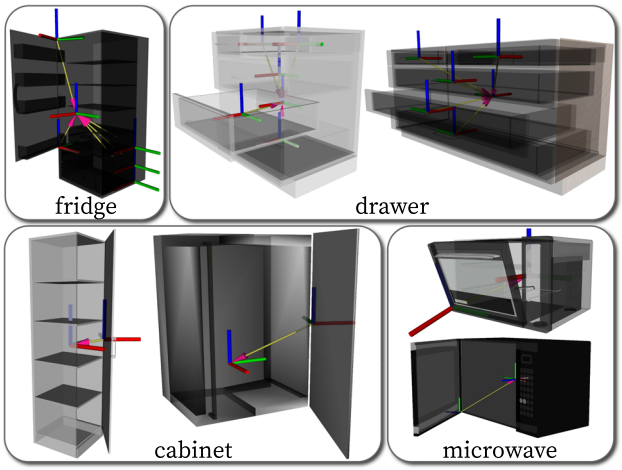


Fig. 4 Examples of articulated CAD models in the database

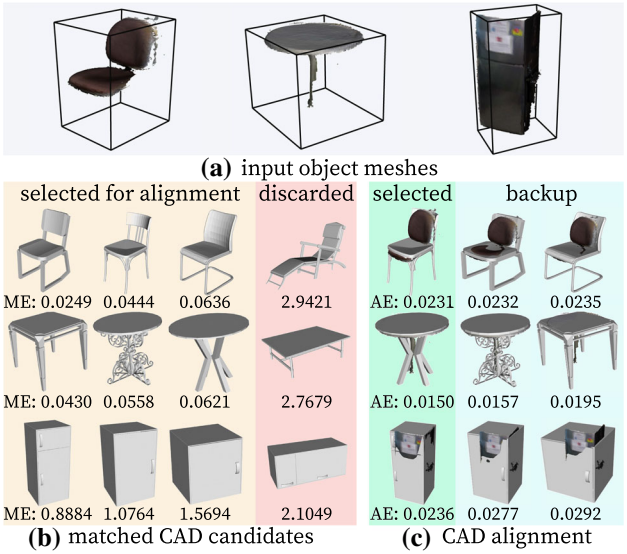


Fig. 5 Examples of matching and aligning CAD candidates to (**a**) input object meshes. (**b**) All CAD models within the same semantic class as the input object are retrieved for matching. Matching Error (ME) indicates the similarity in terms of both shape and the proximity in orientations. After selecting the candidates with the smallest MEs, (**c**) a fine-grained CAD alignment process selects the best CAD model with a proper transformation based on Alignment Error (AE)

tion of x is unknown at this step yet, we uniformly discretize the orientation space into 24 possible orientations. For each rotated CAD model y that aligned to one of the 24 orientations, the algorithm computes a Matching Error (ME):

$$D(x, y) = \omega_1 \cdot d_s(x, y) + \omega_2 \cdot d_\pi(x, y) + \omega_3 \cdot d_b(y), \quad (7)$$

where $\omega_1 = \omega_2 = 1.0$ and $\omega_3 = 0.2$ are the weights of three terms, set empirically. We detail these terms below.

(1) d_s computes the difference of relative 3D bounding box sizes between the segmented mesh and the CAD model:

$$d_s(x, y) = \left\| \frac{\boldsymbol{s}_x}{\|\boldsymbol{s}_x\|_2} - \frac{\boldsymbol{s}_y}{\|\boldsymbol{s}_y\|_2} \right\|. \quad (8)$$

(2) d_π penalizes the misalignment between their surface planes in terms of plane normal and relative distance:

$$d_\pi(x, y) = \min_{f_\Pi} \sum_{\boldsymbol{\pi}_i \in \Pi_x} \left[\left\| \frac{d({T_x}^T \boldsymbol{\pi}_i)}{\|\boldsymbol{s}_x\|_2} - \frac{d(f_\Pi(\boldsymbol{\pi}_i))}{\|\boldsymbol{s}_y\|_2} \right\| + 1 - \boldsymbol{n}(\boldsymbol{\pi}_i)^T \cdot \boldsymbol{n}(f_\Pi(\boldsymbol{\pi}_i)) \right], \quad (9)$$

where T_x denotes the homogeneous transformation matrix from the map frame on the ground to the frame of the bounding box B_x, $d(\cdot)$ the offset of a plane, $\boldsymbol{n}(\cdot)$ the normal vector of a plane, and $f_\Pi : \Pi_x \rightarrow \Pi_y$ a bijection function denoting the assignment of feature planes between x and y. Note that f_Π is also constrained to preserve supporting planes as defined in Eq. (1). As computing d_π involves solving an optimal assignment problem, we adopt a variant of the Hungarian algorithm (Jonker and Volgenant, 1987) to identify the best f_Π between the set of surfaces extracted from a segmented object mesh and that from a candidate CAD model. Then we can calculate the misalignment error term $d_\pi(x, y)$ that the candidate CAD introduces.

(3) $d_b(y)$ is a bias term that adjusts the overall matching error for less preferable CAD candidates:

$$d_b(y) = 1 + \boldsymbol{g}^T \cdot \boldsymbol{z}(y), \quad (10)$$

where $\boldsymbol{z}(y)$ denotes the up-direction of the CAD model in the oriented CAD frame, and $\boldsymbol{g}$ is a unit vector along the gravity direction. Generally, we prefer CAD candidates that are upright instead of leaning aside.

Figure 5b illustrates the matching process. Empirically, we observe that the discarded CAD candidates of "chair" and "table" due to large ME are indeed more visually distinct from the segmented object meshes. Moreover, the "fridge" model with a wrong orientation leads to a much larger ME and is thus discarded. These results demonstrate that our ranking-based matching process can select visually more similar CAD models with a roughly correct orientation. Our system maintains the top 10 orientated CAD candidates with the lowest ME for for a more accurate in the next stage.

4.3 Optimization-Based CAD Alignment

The overarching goal of this step to find an accurate transformation (instead of 24 discretized orientations in the previous step) that aligns a given CAD candidate y to the original object entity x, achieved by estimating a homogeneous transformation matrix between x and y:

$$T = \begin{bmatrix} \alpha R & \boldsymbol{p} \\ \mathbf{0}^T & 1 \end{bmatrix}, \ s.t. \ \min_T \mathcal{J}(x, T \circ y), \tag{11}$$

where $\circ$ denotes the transformation of a CAD candidate y, $\mathcal{J}$ is an alignment error function, α is a scaling factor, $R = Rot(\boldsymbol{z}, \theta)$ is a rotation matrix that only considers the yaw angle under the gravity-aligned assumption, and $\boldsymbol{p}$ is a translation. This translation is subject to the following constraint: $p^g = -d^s + \alpha \cdot s_y^g/2$, as the aligned CAD candidate is supported by a supporting plane $\boldsymbol{\pi}^s = [\boldsymbol{n}^{s^T}, d^s]$.

The objective function $\mathcal{J}$ can be written in a least squares form and minimized by the Levenberg-Marquardt (Mor, 1978) method:

$$\mathcal{J} = \boldsymbol{e}_b^T \Sigma_b \boldsymbol{e}_b + \boldsymbol{e}_p^T \Sigma_p \boldsymbol{e}_p, \tag{12}$$

where $\boldsymbol{e}_b$ is the 3D bounding box error, $\boldsymbol{e}_p$ the plane alignment error, and Σ_b, Σ_p the error covariance matrices of the error terms. Specifically: (i) $\boldsymbol{e}_b$ aligns the height of the two 3D bounding boxes while constraining the ground-aligned rectangle of the transformed B_y inside that of B_x:

$$\boldsymbol{e}_b = [\mathrm{A}(T \circ y) - \mathrm{A}(x \cap T \circ y), \alpha \cdot \boldsymbol{s}_y^g - \boldsymbol{s}_x^g]^T, \tag{13}$$

and (ii) $\boldsymbol{e}_p$ aligns all the matched feature planes as:

$$\begin{aligned} \boldsymbol{e}_p &= [\Delta\boldsymbol{\pi}_1, ..., \Delta\boldsymbol{\pi}_{|\Pi_x|}]^T, \\ \Delta\boldsymbol{\pi}_i &= [-d(\boldsymbol{\pi}_i) + d(T^{-T} \cdot f_\Pi(\boldsymbol{\pi}_i)), \\ &\quad 1 - \boldsymbol{n}(\boldsymbol{\pi}_i)^T \cdot \boldsymbol{n}(T^{-T} \cdot f_\Pi(\boldsymbol{\pi}_i))], \end{aligned} \tag{14}$$

where some of the notations are detailed in Sect. 2.

To evaluate how well an aligned CAD candidate fits the object mesh, we compute an Alignment Error (AE) defined as the root mean square distance between the object mesh vertices and the closest points on aligned CAD candidate; Fig. 5c shows both qualitative and quantitative results. The CAD candidate with the smallest AE will be selected, whereas others are potential substitutions if the selected CADs violate physical constraints, detailed next.

4.4 Global Physical Violation Check

Given a shortlist of matched and aligned CAD candidates, we propose a global physical violation check to finalize the CAD replacement and generate a physically plausible cg'. We first validate supporting relations and object-layout proximal relations for CAD candidates of each object. Specifically, for an object node v_p and its segmented object entity x, we discard an aligned CAD candidate y if it fails to satisfy Eq. (2) with any supporting child v_c of v_p. We also discard aligned CAD candidates that violate the proximal constraints with layout entities.

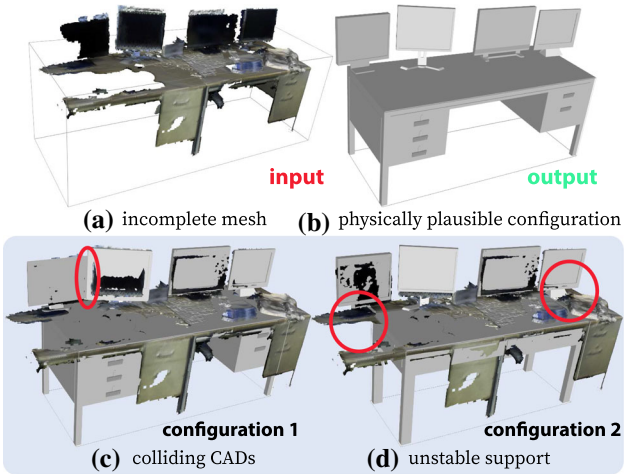


Fig. 6 Given (**a**) incomplete object meshes, our physical common sense reasoning for CAD replacement (**b**) generates a functionally equivalent and physically plausible configuration. Specifically, the CAD matching and alignment algorithms select and rank a shortlist of CAD candidates. A global physical violation check prunes invalid configurations, such as (**c**) collision and (**d**) unstable support

We further check the inter-object proximal constraints and jointly select CAD candidates for each object entity. We formulate this step as a constraint satisfaction problem; starting with a CAD candidate with the minimum AE for each segmented object, we adopt the min-conflict algorithm (Minton et al., 1992) to obtain a global solution of CAD replacement. Finally, as the CAD alignment step cannot guarantee the precise alignment of supporting planes, we adjust the position of CAD models so that Eq. (1) is strictly satisfied for each supporting relation.

Fig. 6 illustrates a typical example, where specific configurations of CAD replacements lead to unstable support or colliding geometry. The global physical violation check prunes invalid configurations and outputs a physically plausible one.

4.5 Kinematic Tree Conversion

The finalized cg' can be readily converted into a kinematic tree to support various robot planning tasks. In this work, we develop an interface to generate a kinematic tree in the form of URDF, which is commonly used in the robotics community.

A kinematic tree contains rigid bodies (links) as nodes, and joints connecting two bodies as edges. Each node in the kinematic tree can be created from either a scene root node, a layout node, a rigid object node, or a rigid part of an articulated object node in cg'. We preserve the joints within articulated CAD models in the kinematic tree, but alter the supporting edges in cg' to either fixed joints (no translation or rotation allowed) or floating joints (allow 3D translation

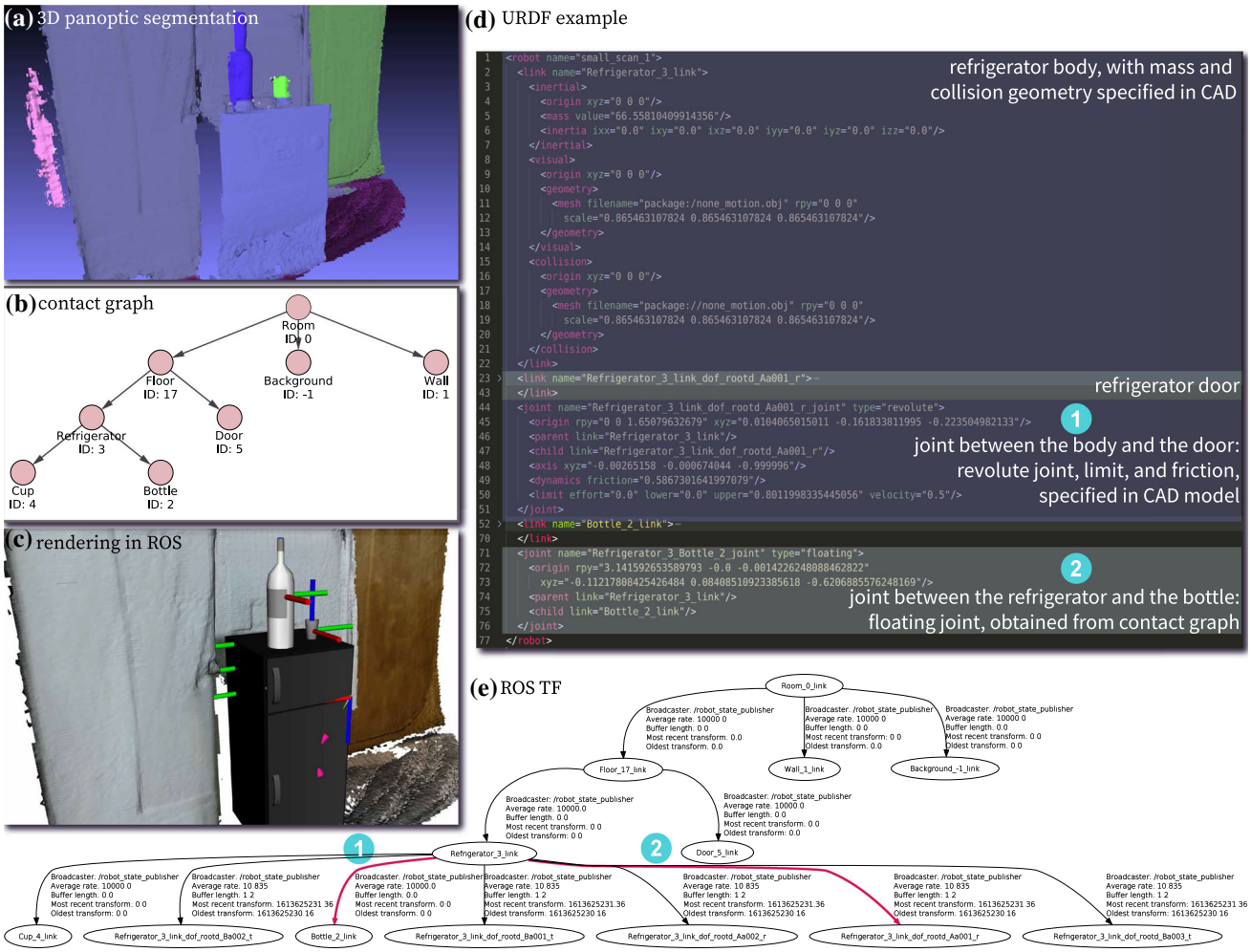


Fig. 7 Convert a contact graph $\mathbf{cg}'$ to a kinematic tree. (**a**) Given the 3D panoptic segmentation produced by our mapping module, (**b**) a contact graph is built and converted to (**d**) URDF with CAD models, which can be seamlessly (**c**) imported to and visualized in ROS Rviz; (**e**) the corresponding ROS TF describes the world states to robots

and 3D rotation unless constrained by collision) based on the semantics of the scene entity pairs. For example, a cup is connected to a table using a floating joint as a robot can freely manipulate it, and a table is linked to the floor via a fixed joint as it cannot be moved.

We show a detailed example of the kinematic tree conversion process in Fig. 7. Based on the 3D panoptic segmentation and the contact graph, our interface generates a kinematic tree in URDF, which can be further visualized as ROS TF and rendered in ROS Rviz.[1] In this example, the fridge is connected to the floor via a fixed joint, and the bottle to the fridge via a floating joint. A revolute joint is inserted to connect the fridge body and the fridge door as specified by the CAD model.

[1] Additional results are available online at https://sites.google.com/view/ijcv2022-reconstruction. Code can be found at https://github.com/hmz-15/Interactive-Scene-Reconstruction.

5 Experiments and Results

5.1 Dataset and Implementation

We evaluate our system primarily on the SceneNN dataset (Hua et al., 2016); it contains RGB-D sequences of various room-size indoor scenes and ground-truth scene meshes annotated with instance-level segmentation. We pick 20 test sequences/scenes that contain diverse object categories to quantitative evaluate the robust panoptic mapping module and demonstrate the interactive scene reconstruction. For baselines that require training on 3D segmentation data, we roughly follow the train/test split in Hua et al. (2018) while using the test set we pick.

In our work, we choose the panoptic segmentation model in Detectron2 (Wu et al., 2019), pre-trained on the COCO panoptic classes (Lin et al., 2014) for segmentation on RGB.

We use Furrer et al. (2018) as the geometric segmentation method for depth images. Of note, our system is designed in a modularized manner so that it is flexible enough to incorporate more powerful models when available. For instance, the segmentation module is designed as a server-side service that will be requested by a client in the perception system when a new image frame arrives and produce a list of segmented masks with labels in the response. Any segmentation methods being wrapped as a service following this protocol could be connected to our system.

5.2 Robust Panoptic Mapping

We evaluate our robust panoptic mapping module on three aspects: (i) 3D panoptic mapping quality, (ii) 3D object instance segmentation, and (iii) oriented 3D bounding box estimation. The first aspect focuses on how well the system reconstructs the scene and segments the objects and layouts within, whereas the latter two emphasize individual objects. Such a protocol design provides a holistic evaluation of the fundamental component of the proposed system: The accuracy of object segmentation and bounding box estimation are crucial for the overall quality of scene reconstruction when matching and aligning CAD models. An ablation study (noted as "w/o joint fusion") is also conducted, where we disable our modifications of jointly processing semantic and instance labels in data fusion, i.e., the procedure described in Sects. 3.2 and 3.3. This study will not only better demonstrate how much the introduced modifications influence the overall mapping performance, but also verify the effectiveness of the per-frame segmentation and fusion technique by comparing the ablated results with those from baselines.

For each sequence used in the experiment, our mapping module processes incoming RGB-D frames with ground-truth camera poses provided by the dataset. We consider 10 semantic classes including 2 *stuff* classes (wall and floor) and 8 most common *thing* classes (bed, table, chair, monitor, sofa, bag, cabinet, and fridge) for evaluation.

3D panoptic mapping This experiment evaluates the overall segmentation performance for panoptic mapping, following the criteria defined in Kirillov et al. (2019) and Narita et al. (2019):

$$PQ = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\mathrm{SQ}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\mathrm{RQ}}, \tag{15}$$

where the Segmentation Quality (SQ) is the averaged Intersection over Union (IoU) of **P**redicted and **g**round-truth panoptic masks on all matched predictions in the same class, and the Recognition Quality (RQ) is the F_1 score (Martin et al., 2004) of object recognition for the aforementioned 10 semantic classes. Panoptic Quality (PQ) is simply the product of SQ and RQ, which better reflects the overall segmentation results.

We compare our panoptic mapping module with Voxblox++ (Grinvald et al., 2019). Table 1 (white columns) shows their PQ, RQ, and SQ on 7 individual SceneNN sequences, averaged on 10 classes. Table 2 further tabulates per-class panoptic segmentation results of all 20 sequences. Of note, we compute PQ, RQ, and SQ in category-level for each semantic class (Table 2), and average the PQ, RQ, and SQ of all classes to obtain those values at the scene-level (Table 1).

Overall, our panoptic mapping module significantly outperforms the baseline as indicated by higher PQ for individual sequences and most of the semantic classes. Without apply-

Table 1 Quantitative class-averaged results of 3D panoptic segmentation and 3D instance segmentation on individual sequences in the SceneNN dataset (Hua et al., 2016)

	Ours				Voxblox++ [18]				ProgressFusion [59]
	Panoptic			Instance	Panoptic			Instance	Instance
ID	PQ	SQ	RQ	mAP	PQ	SQ	RQ	mAP	mAP
011	**45.5**	60.4	50.0	58.3	34.3	64.3	40.0	**80.8**	52.1
030	**50.4**	55.6	64.5	**58.3**	23.4	34.7	26	33.5	56.8
061	**43.0**	52.0	46.3	33.6	25.7	53.1	32.2	38.6	**59.1**
078	**54.7**	54.7	62.5	**50.0**	26.3	52.5	31.7	43.9	34.9
086	**27.3**	39.6	34.6	**40.8**	19.4	32.9	25.2	37.6	35.0
096	**12.5**	21.4	14.6	23.0	7.3	11.9	8.3	14.6	**26.5**
223	**49.5**	60.2	63.3	**60.0**	21.7	40.2	26.7	34.1	40.9

Note that ProgressFusion (Pham et al., 2019a) accounts for more classes than the other two methods. All values are in percentage

Table 2 Per-class 3D panoptic segmentation results in the SceneNN dataset (Hua et al., 2016)

		all	stuff	thing	wall	floor	bed	table	chair	monitor	sofa	bag	cabinet	fridge
Voxblox++ [18]	PQ	24.5	10.9	27.9	4.0	17.8	**18.0**	14.4	35.5	48.5	**46.0**	**24.0**	7.2	29.5
	SQ	77.6	73.7	78.6	69.3	78.0	72.0	71.3	77.0	81.4	82.8	84.0	86.0	73.9
	RQ	31.2	14.3	35.4	5.7	22.9	25.0	20.3	46.0	59.6	55.6	28.6	8.3	40.0
Ours (w/o joint fusion)	PQ	27.8	12.6	31.6	5.6	19.5	8.7	26.7	31.7	48.8	45.7	16.1	**21.9**	**53.4**
	SQ	77.5	71.8	78.9	64	79.6	65.9	73.8	76	89	82.2	72.6	78.5	93.4
	RQ	34.2	16.6	38.6	8.7	24.5	13.3	36.1	41.8	54.9	55.6	22.2	27.9	57.1
Ours	PQ	**35.4**	**44.2**	**33.2**	**25.2**	**63.1**	11.5	**27.4**	**40.1**	**65.7**	34.3	17.4	20.1	48.7
	SQ	80.5	79.3	80.9	73.5	85.0	77.6	76.1	79.1	88.8	80.0	78.3	81.7	85.2
	RQ	43.1	54.3	40.3	34.3	74.3	14.8	36.0	50.6	73.9	42.9	22.2	24.6	57.2

All values are in percentage
The grey columns are meant to indicates that categories "wall" and "floor" belong to "stuff", and the rest belong to "thing".

Table 3 Per-class 3D instance segmentation results on the SceneNN dataset (Hua et al., 2016)

	Input format	bed	table	chair	monitor	sofa	bag	cabinet	fridge
MT-PNet (Pham et al. 2019b)	*Full point cloud*	0.0	12.5	42.8	26.5	0.0	0.0	0.0	0.0
MLS-CRF (Pham et al. 2019b)	*Full point cloud*	0.0	27.3	50.9	38.6	0.0	0.0	0.0	0.0
OccuSeg (Han et al. 2020)	*Full point cloud*	**66.7**	**50.0**	**91.3**	**76.9**	50.0	–	5.7	–
Voxblox++ (Grinvald et al. 2019)	*RGB-D stream*	39.4	22.3	55.6	63.6	72.4	**56.4**	8.5	51.6
Ours (w/o joint fusion)	*RGB-D stream*	17.4	40.7	51.3	48.1	**82.8**	53.2	35.4	**94.5**
Ours	*RGB-D stream*	27.5	46.6	65.3	69.4	64.3	53.2	**43.9**	**94.5**

The numbers in bold and numbers in underscore indicate the best and the second best results, respectively. All values are in percentage
The italics indicate the input format with no other significance

Table 4 Per-class oriented 3D bounding box estimation results on the SceneNN dataset (Hua et al., 2016) based on mAP@0.5 metric

	all	bed	table	chair	monitor	sofa	bag	cabinet	fridge
MT-PNet [60]	10.4	25.8	12.8	19.3	25.0	0.0	0.0	0.0	0.0
MLS-CRF [60]	5.7	0.0	12.6	33.0	0.0	0.0	0.0	0.0	0.0
Voxblox++ [18]	24.1	**39.4**	19.5	31.8	37.0	47.9	0.0	4.0	13.4
Ours (w/o joint fusion)	28.5	17.4	21.4	36.6	29.4	55.8	**53.2**	14.1	0
Ours	45.3	27.5	54.9	44.6	**42.5**	53.7	**53.2**	**29.8**	**56.4**
Ours (refined)	**47.2**	22.9	**68.2**	**49.2**	38.7	**59.1**	**53.2**	**29.8**	**56.4**

All values are in percentage

ing joint fusion, our system still performs better than the baseline Voxblox++, showing the efficacy of our per-frame segmentation. The performance of our full model further demonstrates that our proposed strategies positively contribute to objects and layouts recognition (higher RQ value indicates higher accuracy) and segmentation (higher SQ value).

3D instance segmentation We also evaluate the performance of 3D instance segmentation on 8 *thing* classes using the mAP@0.5 metric, i.e., the Mean Average Precision (mAP) computed using an IoU with a threshold of 0.5. The evaluation is two-fold. First, we report the class-averaged results in the progressive mapping manner on 7 individual sequences compared with Voxblox++ (Grinvald et al., 2019) and ProgressFusion (Pham et al., 2019a), another online semantic mapping framework; see the grey columns in Table 1. Our approach performs better than Voxblox++ on almost all the sequences. Note that the ProgressFusion accounts for all NYUDv2 (Silberman et al., 2012) classes available in the dataset, and we evaluate the performance only on the 8 *thing* classes for our method and Voxblox++. While it is possible to re-train our 2D panoptic segmentation module to incorporate more classes, we believe the current experiment is sufficient to demonstrate the advantage of our panoptic mapping module without defeating its purpose of leveraging pre-trained perception models.

Second, in Table 3, we study the per-class mAP@0.5 of our approach approach compared with Voxblox++ and two

learning-based methods (Pham et al., 2019b; Han et al., 2020) that directly segment 3D instances from the full point cloud of scenes instead of continual RGB-D data stream. As the input formats are different, the results are not directly comparable. They nevertheless provide a better sense about how well our approach performs. We re-train (Pham et al., 2019b) and report the results of its two variants on our test set, and adopt the results reported in Han et al. (2020). Overall, our method performs significantly better than Voxblox++ in most classes, and our variant without joint fusion still slightly outperforms Voxblox++. Although OccuSeg appears to perform the best for object classes that are less likely to be severely occluded in the dataset, our approach also showcases a unique advantage of handling partially-visibly objects such as cabinets and fridges that usually attached to a wall.

Oriented 3D bounding box estimation We further evaluate the accuracy of oriented (gravity-aligned) 3D bounding boxes of object instances, which serve as essential geometric cues for physical reasoning and CAD replacement. Similarly, the mAP@0.5 metric is adopted to evaluate the oriented 3D bounding box estimation on the 8 *thing* classes. Table 4 tabulates results using the baseline method (Grinvald et al., 2019), two variants described in Pham et al. (2019b), our approach, and our approach with supporting-based refinement (detailed in Sect. 2.2). The results indicate that our approach predicts their oriented 3D bounding boxes accurately for most object classes compared with the baselines. The refinement process further improves the performance by completing the partially-observed object boxes. Looking at the two variants in Pham et al. (2019b), while MLS-CRF introduces an extra post-processing step using a Conditional Random Field (CRF) on top of the MT-PNet, its 3D bounding box estimation accuracy drops as extra points from the background are merged into the foreground objects in CRF regularization. An interesting disparity between Pham et al. (2019b)'s instance segmentation results (Table 3) and its bounding box estimation (Table 4) appears—having a zero-score in one place and turning to positive in another. This is because a subtle change in segmenting instances may lead to a large error in estimated bounding boxes.

In summary, the above three quantitative evaluations demonstrate that our robust panoptic mapping module is well suited for (i) recognizing and segmenting scene entities progressively during mapping and (ii) estimating objects' 3D oriented bounding boxes in complex and clustered real indoor environments. The former capability is essential for selecting a proper CAD model to replace a segmented object, and the latter determines the size and scale of that CAD. The ablation study highlights the performance gain introduced by our data fusion procedure, demonstrating the success of jointly dealing with semantic and instance predictions during mapping.

5.3 Inferred Contact Graph

Having extracted object and layout meshes from the volumetric panoptic map, a contact graph cg can be built based on inferred supporting relations. Evaluating the structure of an inferred cg collectively reveals the the performance of object recognition, supporting relation identification, and overall results. To conduct this evaluation, we annotate the contact graphs for four scenes in the SceneNN dataset (Hua et al., 2016) based on their ground-truth segmentation shown in Fig. 8a. A Graph Editing Distance (GED) (Zhang and Shasha, 1989) metric is applied to evaluate the distance between an annotated contact graph and an inferred graph from a segmented map. Specifically, GED measures the dissimilarity of two graph by how many graph editing operations (here we consider insertion, removal of a node or an edge, and substitution of a node ID, a total of five operations) are needed to convert one graph to the other.

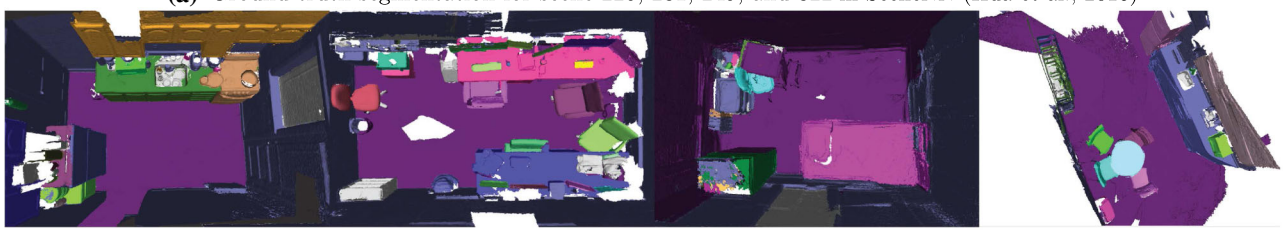
The results are reported in Table 5, where we compare the GED between (grey columns) the annotated contact graph cg_{gt} and that inferred from our mapping results cg_{ours}, and between (white columns) cg_{gt} and that inferred from ground-truth segmentation map cg_{map}. The $Total$ nodes column indicates the size of cg_{gt}, i.e., the number of scene entities a scene has. The $Total$ distance column shows the total editing operations required to covert cg_{ours} or cg_{map} to cg_{gt}, indicating the overall quality of the inferred cg. A qualitative illustration between two graphs is also shown in Fig. 9a. Three types of errors appeared in an inferred graph: (i) Wrong support (or wrong edge): a supporting relation is not assigned correctly, i.e., the parent node of an entity should be another. (ii) Missing detection (or missed node): an entity is not detected or segmented and thus not included in the graph. (iii) Wrong detection (or extra node): an entity that is not supposed to appear in the graph, and the reasons for having extra nodes could be having a wrong semantic label, one entity is segmented as multiple ones, or both. Figure 9b–d depict some examples of error in scene 322.

In Table 5, we observe that our system has difficulties in handling the clustered scene 225 and scene 231 with many small objects, indicated by the high costs of Missing and Wrong detection. The relatively low cost caused by Wrong support indicates that our criteria of determining supporting relations is effective.

5.4 Interactive Scene Reconstruction

Figure 8 showcases the qualitative results for reconstructing functionally equivalent and interactive scenes. Given a volumetric panoptic map (Fig. 8b) and a constructed contact graph, our system reconstructs a highquality, functionally equivalent, interactive scene by (i) replacing incomplete meshes with CAD models and (ii) performing physical

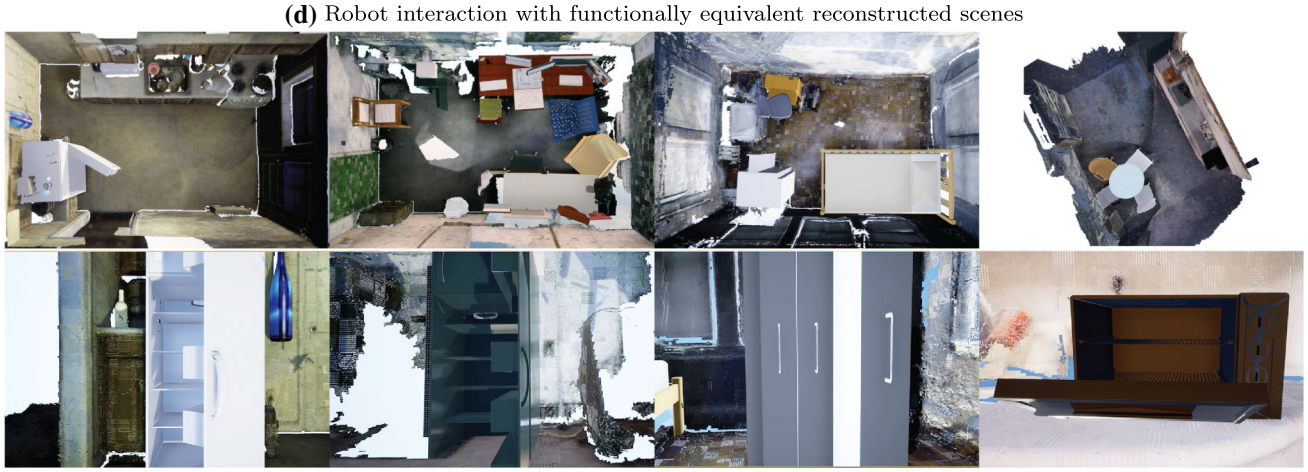
(a) Ground-truth segmentation for scene 225, 231, 249, and 322 in SceneNN (Hua et al., 2016)



(b) Segmentation results produced by the proposed panoptic mapping

(c) Scene reconstruction with functional and actionable CAD objects

(d) Robot interaction with functionally equivalent reconstructed scenes



(e) VR interaction with functionally equivalent reconstructed scenes

Fig. 8 Qualitative results of four reconstructed scenes with actionable CAD models. With functionally equivalent reconstruction, both robots and human users can virtually enter the reconstructed scene for Task and Motion Planning (TAMP) and VR applications

Table 5 GED of four scenes between annotated cg_{gt} and inferred contact graph from our panoptic mapping results cg_{ours} (i.e., Fig. 8b) and from ground-truth maps cg_{map} (i.e., Fig. 8a)

Scene	Total nodes		Total distance		Wrong support		Missing detection		Wrong detection	
	cg_{gt}	$v.s.$	cg_{ours}	cg_{map}	cg_{ours}	cg_{map}	cg_{ours}	cg_{map}	cg_{ours}	cg_{map}
225	20		12	4	1	2	5	0	5	0
231	29		9	4	0	2	2	0	7	0
249	11		7	0	3	0	1	0	0	0
322	17		5	2	1	1	2	0	1	0

Note that editing a wrong support will need two operations, removing an edge and adding an edge, resulting a graph distance of 2

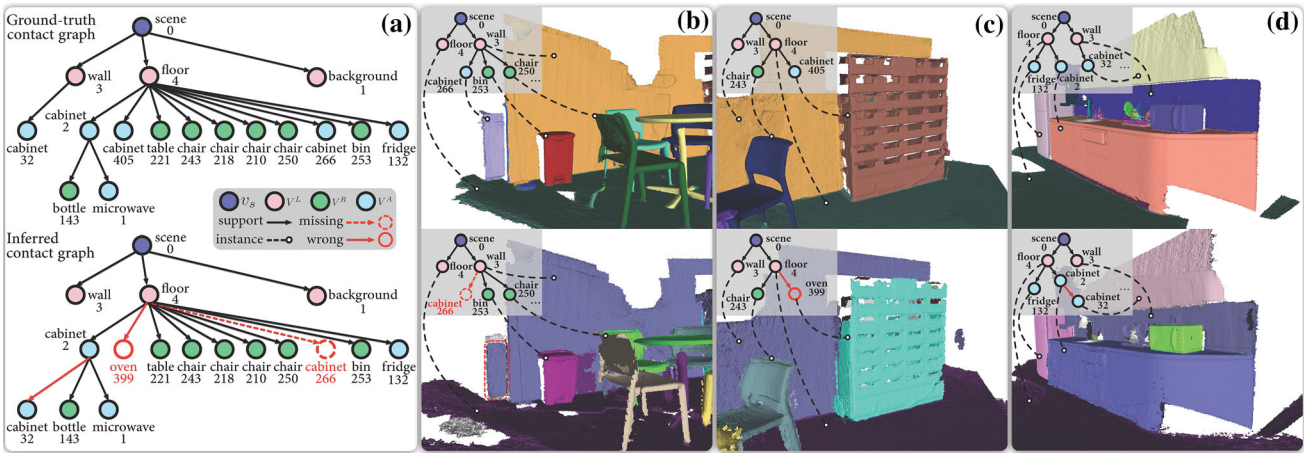


Fig. 9 Comparison between the ground-truth and inferred contact graph. **(a)** The annotated cg_{gt} and the cg_{ours} inferred from our panoptic mapping results for scene 322. **(b)–(d)** highlight a missing detection (cabinet 266 is not detected), a wrong detection (cabinet 405 is detected as oven 399), and a wrong support (cabinet 32 is supported by wall instead of supported by cabinet 2), respectively

reasoning on the contact graph, as shown in Fig. 8c. Nevertheless, we find that our system performs poorly or fails under two circumstances: (i) The incomplete object mesh has misguided or no feature planes, resulting in the misalignment of the CAD model; (ii) The object is not supported by its bottom face (e.g., cabinets on the wall), resulting in the incorrectly reconstructed scene due to the wrong estimate of supporting relations. Sect. 6 provides a more in-depth discussion of the system limitations.

By converting the scene contact graph into a kinematic tree in URDF, we are able to seamlessly import the reconstructed functionally equivalent and interactive scene into various existing simulators. Practically, we also specify physical proprieties (such as link mass, collision geometry, joint friction) in URDF to facilitate more sophisticated simulations. We demonstrate the usage of our reconstructed interactive scenes with several examples: (i) Fig. 8d shows the reconstructed scenes in the ROS environment, which subsequently connects the reconstructed scenes and robot Task and Motion Planning (TAMP). Detailed planning schemes and implementations could be found in Jiao et al. (2021a, b). (ii) Fig. 8e demonstrates that the reconstructed scenes can be loaded into the VR environment (Xie et al., 2019) for interactions with both virtual agents and human users, which opens a new avenue for future studies. (iii) Fig. 10 presents keyframes of a robot executing a long-horizon mobile manipulation task that involves interactions with articulated objects.

5.5 Reconstruction of Physical Scenes

To further evaluate our system under a real-world setting, we conduct experiments to reconstruct physical scenes using a handheld Kinect v2 sensor. We obtain accurate camera poses with a state-of-the-art feature-based SLAM system (Mur-Artal and Tardós, 2017) based on RGB-D streams. The resulting 3D volumetric panoptic map, reconstructed functionally equivalent and interactive scene, and an example of robot interaction are shown in Fig. 11a–c, respectively. This result reveals a huge potential of applying the proposed system to facilitate robot task execution in the physical world.

We further analyze scene reconstruction results using three typical cases that highlight the advantages and failure conditions. In case 1 (Fig. 11d), the table is occluded by the chair and thus is identified as two instances floating in the air. These two tables are determined as floor-supported, and their 3D bounding boxes are further refined on the basis

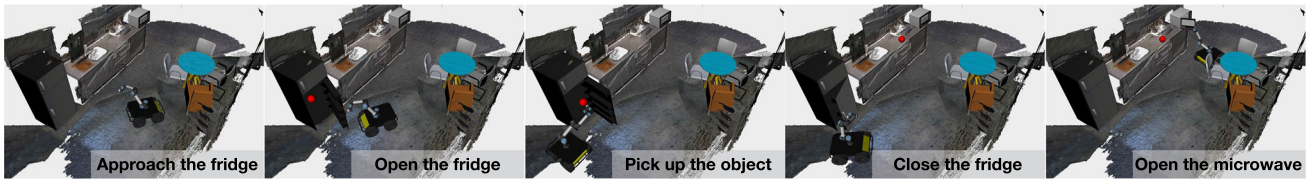
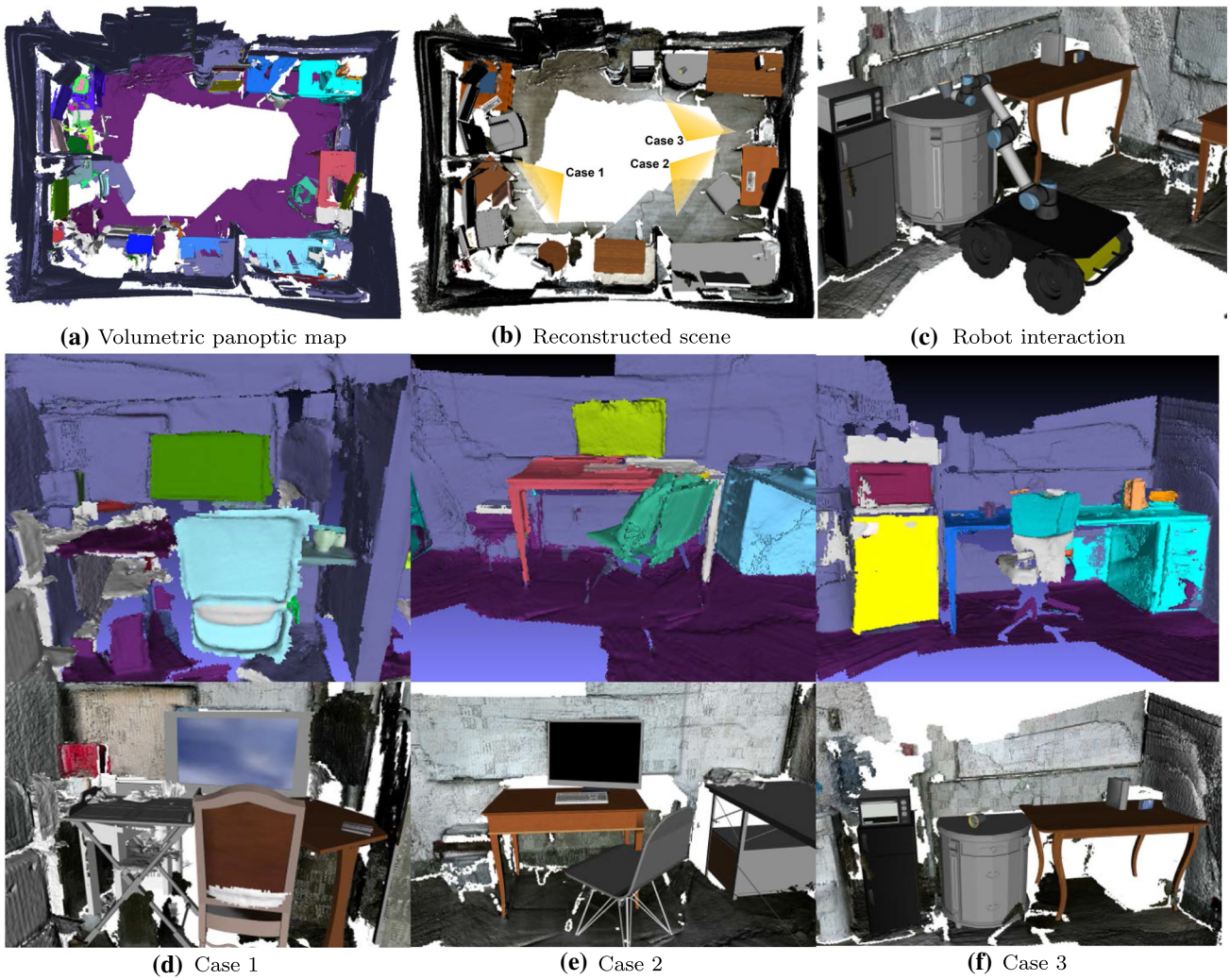


Fig. 10 Robot executing a mobile manipulation task with multiple steps: microwaving an item (indicated by the red ball) by first retrieving it from the fridge (Colour figure online)



(a) Volumetric panoptic map **(b)** Reconstructed scene **(c)** Robot interaction

(d) Case 1 **(e)** Case 2 **(f)** Case 3

Fig. 11 Reconstructing a physical scene with a handheld RGB-D sensor. (**a**) The panoptic segmentation and the overall mapping. (**b**) The reconstructed scene with CAD models replacing the segmented objects, which supports (**c**) a robot to simulate its Task and Motion Planning (TAMP). (**d**–**f**) Qualitative results of segmentation and reconstruction. Our system recognizes most of the objects and properly replaces them with CAD models that are similar to those objects in the physical scene; see Case 2 and 3. A common problem is due to occlusion, which causes inaccurate detection, e.g., one desk is recognized as two as it is occluded by the chair; see Case 1 and 3

of the supporting relations. The system eventually outputs two separate tables in the reconstructed interactive scene, where their poses align with the oriented 3D bounding boxes of the partial meshes. Case 2 (Fig. 11e) shows an example of a better reconstructed workspace. Given the incompletely segmented table and chair point cloud, our system can correctly estimate the supporting relations and their orientations, replace each mesh with a similar CAD model, and finally produce a functionally equivalent and physically plausible workspace, although the dimension of the table is not ideal as part of the point cloud behind the chair is not detected and segmented correctly. Case 3 (Fig. 11f) provides a more challenging example. The fridge and microwave are segmented and replaced by articulated CAD models, whereas the chair is not successfully detected and is removed from the reconstructed scene. Similar to case 1, the table is identified and

replaced with two instances. To avoid mesh penetration, the proximal constraints incorporated by the *cg* helps the CAD replacement process to select a rounded table on the left side, but it is not a satisfactory replacement due to the large discrepancy in shapes.

6 Discussion

We now discuss in greater depth six topics related to the presented work.

6.1 Scene Functionality

Most computer vision tasks focus on devising new methodologies and representations that are beneficial within the scope of computer vision. However, this paper seeks to address a new task of building a representational system with the emphasis of facilitating robot activities. The core of the system is to represent the scene *functionality*, one of the key common senses governing our understanding of a scene (Zhu et al., 2020). This goal is achieved by associating high-level cues from object semantics (e.g., whether they can be moved, opened, or can support other interactions) and low-level cues (i.e., replacing the object meshes with CAD models, whose underlying kinematics indicates how exactly they interact). Additional object attributes, affordance, or task-dependent information can be annotated to CAD models to depict the scenes more comprehensively. A subsequent, interesting open question is how to quantify the divergence between the actual scene and the reconstructed one with CAD replacements.

6.2 Scene Representation

The contact graph *cg* produced by the proposed system is a holistic, but approximate scene representation. By itself is indeed insufficient for robot task execution where more precious local scene representations are needed. Although the *cg* does not seem directly beneficial, its importance is two-fold when considering a robot designed to operate over a long period of time. Firstly, the representation maintains a global belief of the scene, helps a robot to anticipate the effects of (sequence of) actions, and incorporates the actual action effects back to the *cg*. This is essential for the robot to forward search for a task plan over a long horizon (Kaelbling, 2020). Secondly, given the variety of tasks a robot may anticipate, our *cg* can serve as a carrier for those necessary local representations that can be annotated, trained beforehand or built online with proper perception modules. Otherwise, different task-driven representations are isolated, lacking a proper organization.

6.3 Task and Motion Planning (TAMP)

Existing TAMP frameworks are oftentimes too brittle to handle a large variety of interactions in different environments. Kaelbling and Lozano-Pérez (2011) and Srivastava et al. (2014) propose new TAMP frameworks, making planning long-horizon manipulation tasks possible. Still, these frameworks focus on pick and-place tasks with carefully defined environmental constraints, making it difficult for complex indoor manipulation tasks. Garrett et al. (2020) devise a framework for a complex problem, which requires interactions with articulated objects. Similarly, this work is still restricted to carefully designed environments with limited variety in the setup. A key factor to this problem is the lack of simulation environments that support various interactive actions (e.g., door opening, object picking) and semantic relations among objects. Crucially, it could be time-consuming to generate these environments manually. In comparison, our framework can automatically generate interactive environments using real sensory data in challenging physical world and it demonstrates the capability to support more complex TAMP studies in the future Jiao et al. (2022), Zhang et al. (2022).

6.4 Embodied AI

Embodied AI researches focus on learning a policy, mostly in simulations, that can ultimately be applied to real-world applications. Therefore, a significant amount of work is to develop simulation platforms to support learning. Our perspective echoes the motivation of task-oriented vision—designing a proper vision system that better suits a given task (Ikeuchi and Hebert, 1992). Specifically, our work allows the agent to acquire a policy specific to the given environment for the given task by capturing and representing the *actionable* information in the environment from the agent's view. Thus, our work goes beyond panoptic segmentation and 3D reconstruction.

6.5 Suppoting Relations

Inferred supporting relations define the structure of contact graph. While this paper mainly concerns about stable support see Eq. (3), there are several other supporting configurations. For instance an object is hang on the wall, supported by two adjacent tables, placed on floor and tilted against another object. These types of supports are not explicitly modeled and may not be well handled. Our system can nevertheless reveal their supporting relations partially. For instances, the blue bottle in Fig. 1c is regarded as supported by the wall because no valid supporting parent is identified, and it is very close to the wall. Whereas in Fig. 9d, the upper cabinet that is supported by the wall (and possibly the ceiling as well)

is wrongly considered as supported by the lower cabinet. In other cases where an object is supported by multiple entities simultaneously, only one entity would be identify as a supporting parent based on overlapping area defined in Eq. (2). For a tilted object on the floor, only the floor would be identified as the supporting object. Nevertheless, more specific spatial relations can be modeled and incorporated into the contact graph representation to extend the system's capability.

6.6 Other Limitations

The system's performance heavily relies on 3D panoptic segmentation of scene entities and the CAD replacement of object meshes. Currently, our robust panoptic mapping module utilizes open-sourced software to generate panoptic segmentation on RGB frames. While its development is beyond this paper's scope, new models and methods are emerging in the fast-paced community, and our system is designed to easily incorporate newer methods to improve the mapping performance further and support subsequent processes by reducing error propagated in each stage.

Our CAD replacement algorithm matches and aligns CAD models to incomplete meshes based on simple geometric features, i.e., 3D bounding boxes and surface planes, which are potentially fragile when the meshes are noisy and incomplete. In the future, we may integrate deep learning-based methods (Avetisyan et al., 2019b; Pham et al., 2018) for more robust and accurate CAD replacement.

The articulated CAD models are unlikely to match the structure of real objects exactly. One potential solution is to detect and segment object parts and estimate the kinematics to assemble more fine-grained CAD models. The PartNet dataset (Mo et al., 2019) and related methods (Xu et al., 2022) provide an initial direction to start with.

In addition to the supporting relations and annotated kinematics information, various actionable information and object attributes may also contribute to and facilitate robot interactions. One central question remains unanswered is how to balance manual efforts and algorithmic efforts so that an intelligent robot can better excel in an ever-changing environment.

7 Conclusions and Future Work

This paper proposes a new task of reconstructing functionally equivalent and interactive scenes to simulate robot autonomy and develops a full system to demonstrate this new perspective. Contrasting to the classic view of scene reconstruction that focuses on the geo-information, our system captures semantics and associated actionable information in scene entities by (i) a novel panoptic mapping module that reconstructs individuals objects and layouts, (ii) a geometric and physical reasoning module to replace the incomplete objects meshes with part-based interactive CAD models, and (iii) a contact graph representation that facilitates physically plausible scene reconstruction, and reflects action opportunities and action outcomes in terms of kinematic information. In experiments, we first quantitatively demonstrate that our system can produce high-quality panoptic segmentation, a prerequisite for the subsequent processes. We further qualitatively showcase various reconstructed scenes with functional CAD model replacements, from dataset and real-world scanning, that support fine-grained interactions in ROS and VR environments.

In the future, we hope to improve the CAD matching and alignment processes by introducing more robust feature extraction and exploring learning-based methods. Another promising future direction is to incorporate sophisticated part-based object recognition and modeling. Together with a CAD assembling module, it is possible to generate a CAD model that matches a segmented object with much finer details and reflects its functionality better. Meanwhile, more functional and attribute information can be encoded to CAD models to better reveal the "Dark Matter" (Zhu et al., 2020) of a scene. Finally, we will explore the feasibility of promoting the embodied AI research from navigation tasks to fine-grained manipulation tasks using our reconstruction framework.

References

Agin, G. J. and Binford T. O. 1973 "Computer description of curved objects." *International Joint Conference on Artificial Intelligence (IJCAI)*.

Armeni, I., He, Z. Y., Gwak, J., Zamir, A. R., Fischer, M., Malik, J., & Savarese S. (2019). 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Avetisyan, A., Dahnert, M., Dai, A., Savva, M., Chang, A. X., & Nießner, M. (2019a). Scan2cad: Learning cad model alignment in rgb-d scans. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Avetisyan, A., Dai, A., & Nießner, M. (2019b). End-to-end cad model retrieval and 9dof alignment in 3d scans. In *International Conference on Computer Vision (ICCV)*.

Batra, D., Chang, A. X., Chernova, S., Davison, A. J., Deng, J., Koltun, V., Levine, S., Malik, J., Mordatch, I., & Mottaghi R. et al. (2020). Rearrangement: A challenge for embodied ai. arXiv preprint arXiv:2011.01975

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., et al. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics (T-RO), 32*(6), 1309–1332.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A., & Zhang Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. In *International Conference on 3D Vision (3DV)*.

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

Chang, H. J., & Demiris, Y. (2017). Highly articulated kinematic structure estimation combining motion and skeleton information. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 40*(9), 2165–2179.

Chen, Y., Huang, S., Yuan, T., Qi, S., Zhu, Y., & Zhu, S. C. (2019). Holistic++ scene understanding: Single-view 3d holistic scene parsing and human pose estimation with human-object interaction and physical commonsense. In *International Conference on Computer Vision (ICCV)*.

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

Deitke, M., Han, W., Herrasti, A., Kembhavi, A., Kolve, E., Mottaghi, R., Salvador, J., Schwenk, D., VanderBilt, E., & Wallingford, M. et al. (2020). Robothor: An open simulation-to-real embodied ai platform. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Edmonds, M., Gao, F., Liu, H., Xie, X., Qi, S., Rothrock, B., et al. (2019). A tale of two explanations: Enhancing human trust by explaining robot behavior. *Science Robotics, 4*(37), eaay4663.

Edmonds, M., Gao, F., Xie, X., Liu, H., Qi, S., Zhu, Y., Rothrock, B., & Zhu, S. C. (2017). Feeling the force: Integrating force and pose for fluent discovery through imitation learning to open medicine bottles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Furrer, F., Novkovic, T., Fehr, M., Gawel, A., Grinvald, M., Sattler, T., Siegwart, R., & Nieto J. (2018). Incremental object database: Building 3d models from multiple partial observations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Garrett, C. R., Paxton, C., Lozano-Pérez, T., Kaelbling, L. P., & Fox D. (2020). Online replanning in belief space for partially observable task and motion problems. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Gibson, J. J. (1950). *The perception of the visual world*. Houghton Mifflin.

Gibson, J. J. (1966). *The senses considered as perceptual systems*. Houghton Mifflin.

Grinvald, M., Furrer, F., Novkovic, T., Chung, J. J., Cadena, C., Siegwart, R., & Nieto, J. (2019). Volumetric instance-aware semantic mapping and 3d object discovery. *IEEE Robotics and Automation Letters (RA-L), 4*(3), 3037–3044.

Gupta, S., Arbeláez, P., Girshick, R., & Malik J. (2015). Aligning 3d models to rgb-d images of cluttered scenes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Han, L., Zheng, T., Xu, L., & Fang, L. (2020). Occuseg: Occupancy-aware 3d instance segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Han, M., Zhang, Z., Jiao, Z., Xie, X., Zhu, Y., Zhu, S. C., & Liu H. (2021). Reconstructing interactive 3d scenes by panoptic mapping and cad model alignments. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Hartley, R., & Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge University Press.

He, K., Gkioxari, G., Dollár, P., & Girshick R. (2017). Mask r-cnn. In *International Conference on Computer Vision (ICCV)*.

Hoang, D. C., Lilienthal, A. J., & Stoyanov, T. (2020). Panoptic 3d mapping and object pose estimation using adaptively weighted semantic information. *IEEE Robotics and Automation Letters (RA-L), 5*(2), 1962–1969.

Hua, B. S., Pham, Q. H., Nguyen, D. T., Tran, M. K., Yu, L. F., & Yeung S. K. (2016). Scenenn: A scene meshes dataset with annotations. In *International Conference on 3D Vision (3DV)*.

Hua, B. S., Tran, M. K., & Yeung, S. K. (2018). Pointwise convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huang, S., Qi, S., Xiao, Y., Zhu, Y., Wu, Y. N., & Zhu, S. C. (2018a). Cooperative holistic scene understanding: Unifying 3d object, layout, and camera pose estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Huang, S., Qi, S., Zhu, Y., Xiao, Y., Xu, Y., & Zhu, S. C. (2018b). Holistic 3d scene parsing and reconstruction from a single rgb image. In *European Conference on Computer Vision (ECCV)*.

Ikeuchi, K., & Hebert M. (1992). Task-oriented vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Jia, B., Chen, Y., Huang, S., Zhu, Y., & Zhu, S. C. (2020). Lemma: A multi-view dataset for learning multi-agent multi-task activities. In *European Conference on Computer Vision (ECCV)*.

Jiang, C., Qi, S., Zhu, Y., Huang, S., Lin, J., Yu, L. F., et al. (2018). Configurable 3d scene synthesis and 2d image rendering with per-pixel ground truth using stochastic grammars. *International Journal of Computer Vision (IJCV), 126*(9), 920–941.

Jiao, Z., Niu, Y., Zhang, Z., Zhu, S. C., Zhu, Y., & Liu, H. (2022). Sequential Manipulation Planning on Scene Graph. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Jiao, Z., Zhang, Z., Jiang, X., Han, D., Zhu, S. C., Zhu, Y., & Liu, H. (2021a). Consolidating kinematic models to promote coordinated mobile manipulations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Jiao, Z., Zhang, Z., Wang, W., Han, D., Zhu, S. C., Zhu, Y., & Liu H. (2021b). Efficient task planning for mobile manipulation: A virtual kinematic chain perspective. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Jonker, R., & Volgenant, A. (1987). A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing, 38*(4), 325–340.

Kaelbling, L. P. (2020). The foundation of efficient robot learning. *Science, 369*(6506), 915–916.

Kaelbling, L. P., & Lozano-Pérez, T. (2011). Hierarchical task and motion planning in the now. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (2019). Panoptic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Knill, D. C., & Richards, W. (1996). *Perception as Bayesian inference*. Cambridge University Press.

Li, X., Liu, S., Kim, K., Wang, X., Yang, M. H., & Kautz, J. (2019). Putting humans in a scene: Learning affordance in 3d indoor environments. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, X., Wang, H., Yi, L., Guibas, L. J., Abbott, A. L., & Song, S. (2020). Category-level articulated object pose estimation. In *International Conference on Computer Vision (ICCV)*.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*.

Liu, H., Zhang, Y., Si, W., Xie, X., Zhu, Y., & Zhu, S. C. (2018a). Interactive robot knowledge patching using augmented reality. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Liu, H., Zhang, C., Zhu, Y., Jiang, C., & Zhu S. C. (2019). Mirroring without overimitation: Learning functionally equivalent manipulation actions. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Liu, L., Xia, X., Sun, H., Shen, Q., Xu, J., Chen, B., et al. (2018). Object-aware guidance for autonomous scene reconstruction. *ACM Transactions on Graphics (TOG), 37*(4), 1–12.

Malandain, G., & Boissonnat, J. D. (2002). Computing the diameter of a point set. *International Journal of Computational Geometry & Applications, 12*(06), 489–509.

Martin, D. R., Fowlkes, C. C., & Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 26*(5), 530–549.

Martín-Martín R., & Brock, O. (2019). Coupled recursive estimation for online interactive perception of articulated objects. *International Journal of Robotics Research (IJRR)*, 1–37.

McCormac, J., Clark, R., Bloesch, M., Davison, A., & Leutenegger S. (2018). Fusion++: Volumetric object-level slam. In *International Conference on 3D Vision (3DV)*.

McCormac, J., Handa, A., Davison, A., & Leutenegger, S. (2017). Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Min, H., Luo, R., Zhu, J., Bi, S., et al. (2016). Affordance research in developmental robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems, 8*(4), 237–255.

Minton, S., Johnston, M. D., Philips, A. B., & Laird, P. (1992). Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence, 58*(1–3), 161–205.

Mo, K., Zhu, S., Chang, A. X., Yi, L., Tripathi, S., Guibas, L. J., & Su, H. (2019). Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Moré, J. J. (1978). The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical analysis* (pp. 105–116). Springer.

Mur-Artal, R., & Tardós, J. D. (2017). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics (T-RO), 33*(5), 1255–1262.

Myers, A., Teo, C. L., Fermüller, C., & Aloimonos, Y. (2015). Affordance detection of tool parts from geometric features. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Narita, G., Seno, T., Ishikawa, T., & Kaji Y. (2019). Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., & Nieto, J. (2017). Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Pham, Q. H., Hua, B. S., Nguyen, T., & Yeung, S. K. (2019a). Real-time progressive 3d semantic segmentation for indoor scenes. In *Proceedings of Winter Conference on Applications of Computer Vision (WACV)*.

Pham, Q. H., Nguyen, T., Hua, B. S., Roig, G., & Yeung, S. K. (2019b). Jsis3d: Joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pham, Q. H., Tran, M. K., Li, W., Xiang, S., Zhou, H., Nie, W., Liu, A., Su, Y., Tran, M. T., & Bui, N. M. et al. (2018). Shrec'18: Rgb-d object-to-cad retrieval. In *3DOR: Proceedings of the 11th Eurographics Workshop on 3D Object Retrieval*.

Pronobis, A., & Jensfelt, P. (2012). Large-scale semantic mapping and reasoning with heterogeneous modalities. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Qi, S., Jia, B., Huang, S., Wei, P., & Zhu, S. C. (2020). A generalized earley parser for human activity parsing and prediction. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 43*, 2538–2554.

Qi, S., Zhu, Y., Huang, S., Jiang, C., & Zhu, S. C. (2018). Human-centric indoor scene synthesis using stochastic grammar. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 39*(6), 1137–1149.

Rosinol, A., Gupta, A., Abate, M., Shi, J., & Carlone, L. (2020). 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Robotics: Science and Systems (RSS)*.

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., & Malik J. et al. (2019). Habitat: A platform for embodied ai research. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision (ECCV)*. Springer.

Song, S., Lichtenberg, S. P., & Xiao, J. (2015). Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Song, S., Yu, F., Zeng, A., Chang, A. X., Savva, M., & Funkhouser, T. (2017). Semantic scene completion from a single depth image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Srivastava, S., Fang, E., Riano, L., Chitnis, R., Russell, S., & Abbeel, P. (2014). Combined task and motion planning through an extensible planner-independent interface layer. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Sturm, J., Stachniss, C., & Burgard, W. (2011). A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research, 41*, 477–526.

Sui, Z., Chang, H., Xu, N., & Jenkins, O. C. (2020). Geofusion: Geometric consistency informed scene estimation in dense clutter. *IEEE Robotics and Automation Letters (RA-L), 5*(4), 5913–5920.

Taguchi, Y., Jian, Y. D., Ramalingam, S., & Feng, C. (2013). Point-plane slam for hand-held 3d sensors. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Wada, K., Sucar, E., James, S., Lenton, D., & Davison, A. J. (2020). Morefusion: Multi-object reasoning for 6d pose estimation from volumetric fusion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wald, J., Dhamo, H., Navab, N., & Tombari, F. (2020). Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, Y., Kirillov, A., Massa, F., Lo, W. Y., & Girshick, R. (2019). *Detectron2*. https://github.com/facebookresearch/detectron2

Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., & Wang H, et al. (2020). Sapien: A simulated part-based interactive environment. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xia, F., Shen, W. B., Li, C., Kasimbeg, P., Tchapmi, M. E., Toshev, A., et al. (2020). Interactive Gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters (RA-L), 5*(2), 713–720.

Xie, X., Liu, H., Zhang, Z., Qiu, Y., Gao, F., Qi, S., Zhu, Y., & Zhu, S. C. (2019). Vrgym: A virtual testbed for physical and interactive ai. In *Proceedings of the ACM Turing Celebration Conference*-China, pp. 1–6.

Xu, K., Huang, H., Shi, Y., Li, H., Long, P., Caichen, J., et al. (2015). Autoscanning for coupled scene reconstruction and proactive object analysis. *ACM Transactions on Graphics (TOG), 34*(6), 1–14.

Yang, S., & Scherer, S. (2019a). Cubeslam: Monocular 3-d object slam. *IEEE Transactions on Robotics (T-RO), 35*(4), 925–938.

Yang, S., & Scherer, S. (2019b). Monocular object and plane slam in structured environments. *IEEE Robotics and Automation Letters (RA-L), 4*(4), 3145–3152.

Yi, L., Zhao, W., Wang, H., Sung, M., & Guibas, L. J. (2019). Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yuan, T., Liu, H., Fan, L., Zheng, Z., Gao, T., Zhu, Y., & Zhu, S. C. (2020). Joint inference of states, robot knowledge, and human (false-)beliefs. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Yu, L. F., Yeung, S. K., Tang, C. K., Terzopoulos, D., Chan, T. F., & Osher, S. J. (2011). Make it home: Automatic optimization of furniture arrangement. *ACM Transactions on Graphics (TOG), 30*(4), 1–12.

Zhang, Z., Jiao, Z., Wang, W., Zhu, Y., Zhu, S. C., & Liu, H. (2022). Understanding Physical Effects for Effective Tool-use. *IEEE Robotics and Automation Letters (RA-L), 7*(4), 9469–9476.

Zhang, J., Zhao, X., Chen, Z., & Lu, Z. (2019). A review of deep learning-based semantic segmentation for point cloud. *IEEE Access, 7*, 179118–179133.

Zhang, K., & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing, 18*(6), 1245–1262.

Zhang, Z., Zhu, Y., & Zhu, S. C. (2020). Graph-based hierarchical knowledge representation for robot task transfer from virtual to physical world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Zhao, Y., & Zhu, S. C. (2011). Image parsing with stochastic scene grammar. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhao, Y., & Zhu, S. C. (2013). Scene parsing by integrating function, geometry and appearance models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zheng, B., Zhao, Y., Yu, J., Ikeuchi, K., & Zhu, S. C. (2015). Scene understanding by reasoning stability and safety. *International Journal of Computer Vision (IJCV), 112*(2), 221–238.

Zhu, S. C., & Mumford, D. (2007). A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision, 2*(4), 259–362.

Zhu, Y., Gao, T., Fan, L., Huang, S., Edmonds, M., Liu, H., et al. (2020). Dark, beyond deep: A paradigm shift to cognitive ai with human-like common sense. *Engineering, 6*(3), 310–345.

Zhu, Y., Jiang, C., Zhao, Y., Terzopoulos, D., & Zhu, S. C. (2016). Inferring forces and learning human utilities from videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhu, Y., Zhao, Y., & Zhu, S. C. (2015). Understanding tools: Task-oriented object modeling, learning and recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zou, C., Guo, R., Li, Z., & Hoiem, D. (2019). Complete 3d scene parsing from an rgbd image. *International Journal of Computer Vision (IJCV), 127*(2), 143–162.