robotics

MDPI

Article

Open-Vocabulary Instance Segmentation for Scene Understanding in Mobile Robots

Macoris Decena-Gimenez, Pepe Ojeda, Jose-Raul Ruiz-Sarmiento * and Javier Gonzalez-Jimenez

Machine Perception and Intelligent Robotics Group (MAPIR-UMA), Malaga Institute for Mechatronics Engineering and Cyber–Physical Systems (IMECH.UMA), University of Malaga, 29071 Malaga, Spain; macorisd@uma.es (M.D.-G.); ojedamorala@uma.es (P.O.); javiergonzalez@uma.es (J.G.-J.)

* Correspondence: jotaraul@uma.es

Abstract

High-level robotic tasks, such as those involving planning and interaction, demand a certain degree of scene understanding through suitable representations of the environment that enrich geometric information with object-level semantics, commonly referred to as semantic maps. Traditional techniques to build these maps are restricted to working with a set of categories that is fixed during training, which limits the information the resulting maps can contain. To overcome this limitation, this work presents an open-vocabulary semantic mapping pipeline that integrates TALOS (TAgging–LOcation–Segmentation) with the probabilistic, instance-aware Voxeland framework. TALOS employs a modular generative approach in which vision–language and large language models infer categories for the objects in the scene, an open-vocabulary grounding model localizes their instances, and a category-agnostic segmentation model produces pixel-accurate masks. These predictions are combined with depth data and fused into persistent 3D maps using Voxeland while preserving geometric, semantic, and instance uncertainty. The approach was evaluated on diverse ScanNet v2 scenes and compared with two alternative perception front-ends: the closed-vocabulary Detectron2/Mask R-CNN and the open-vocabulary discriminative model YOLOE. Across five independent runs, TALOS achieves a mean mAP@0.5 of 0.1607 ± 0.0255, compared with 0.0834 for YOLOE and 0.0453 for Detectron2/Mask R-CNN, and exceeds both baselines in every run. Qualitative results further show a better balance between map completeness, geometric clarity, semantic coherence, and instance separation.

Keywords: mapping; computer vision; perception and detection; point cloud; mobile robots

Check for updates

Academic Editors: Mónica Ballesta, Oscar Reinoso García and María Flores Tenza

Received: 29 June 2026
Revised: 10 September 2026
Accepted: 14 September 2026
Published: 19 September 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Scene understanding by a mobile robot entails the extraction of semantic information from sensory data to characterize the elements in the scene, aiming to support sophisticated robotic tasks that go beyond purely geometric reasoning. This kind of information underpins high-level robotic capabilities such as decision-making, interaction with humans or other robots/agents, object manipulation, and contextualized navigation, all of which rely on the ability to interpret and reason about both the geometric structure of the environment and the semantics of its constituent elements [1,2]. Moreover, in modern robotic architectures incorporating Generative AI components, this semantic awareness is also key to enabling robots to reason and operate in a context-aware manner [3–6]. Therefore, achieving a rich and actionable representation of the environment has been a long-standing goal in robotics, motivating decades of research that have evolved from purely geometric

mapping approaches to more recent methods incorporating semantic information about the elements present in the scene [7,8].

A widely adopted strategy to endow robots with such semantic scene understanding is the building of semantic maps: spatial representations in which geometric elements are augmented with meta-information, such as object categories [9]. Typically, these maps are built from the output of visual perception techniques, such as object detection and semantic segmentation, which are nowadays based on deep learning methods. Many existing 3D semantic mapping systems and their applications demonstrate the value of fusing per-frame predictions into coherent 3D maps [10–13]. However, the perception front-ends in these systems typically rely on closed-vocabulary models: networks trained on a fixed set of categories defined at training time (e.g., Mask R-CNN [14] or similar architectures [15,16]). A common dataset choice for training is Microsoft COCO [17] with its 80 object categories, a vocabulary that is clearly insufficient to represent the richness and diversity of real-world scenes. Other standard benchmarks such as PASCAL VOC [18], Cityscapes [19], NYUDv2 [20], and SIFT Flow [21] suffer from the same constraint with equally limited label sets. This design introduces a fundamental bottleneck for robotics: any object whose category was not included in the training vocabulary is systematically ignored during mapping, regardless of its relevance to the task at hand. The limitations of closed-vocabulary perception become particularly acute in unstructured or evolving environments, where the diversity of objects far exceeds any pre-defined label set.

Recent advances in vision–language models have opened a promising path to overcome this constraint. In particular, large-scale pretraining on image–text data [22] has facilitated the emergence of open-vocabulary detection and segmentation methods that are able to recognize arbitrary categories specified at inference time through natural language [23–25]. In 3D scene understanding, recent works have shown that open-vocabulary features can be lifted into volumetric or point-cloud representations, enabling queries over previously unseen categories [7,26]. Class-agnostic segmentation foundations such as SAM [27] and its subsequent versions [28], combined with open-set grounding models like Grounding DINO [29], further extend the possibilities by decoupling localization from category assignment. Together, these developments motivate the integration of open-vocabulary instance segmentation into 3D semantic mapping pipelines for mobile robots.

In this context, TALOS (TAgging–LOcation–Segmentation) [30] was introduced in our previous work as a method that resorts to these cutting-edge models to perform semantic instance segmentation. Unlike discriminative open-vocabulary detectors [31], TALOS adopts a generative approach: it leverages Large Vision–Language Models (LVLMs) and Large Language Models (LLMs) to identify the semantic content of a scene through natural-language reasoning, and subsequently grounds the predicted categories using state-of-the-art localization [29] and segmentation [28] models. The pipeline is organized into three sequential and fully modular stages: Tagging, which infers the set of object categories present in the image; Location, which produces bounding boxes for each predicted category; and Segmentation, which generates pixel-accurate masks for each instance. This modular design enables independent replacement of the model used in each stage, facilitating experimentation and upgrades as new foundation models become available. It was initially validated through qualitative tests, yielding promising results. However, testing the results on individual images is insufficient to evaluate the fitness of this method as a tool for robotic scene understanding, as there are aspects of the problem (temporal coherence, viewpoint dependence, etc.) which cannot be addressed in this manner. Therefore, the logical next step of research is to evaluate the method in-situ as part of a semantic mapping pipeline, so the quality of the generated map can be analysed instead.

In this work we extend this approach in the following manners:

- **Integration of TALOS into a semantic mapping framework.** We describe the incorporation of TALOS as the perception front-end of Voxeland [13], a probabilistic semantic mapping framework, thus integrating 2D open-vocabulary predictions into uncertainty-aware 3D instances in the semantic map. For building said map, the 2D predictions are combined with 3D geometric information (typically coming from an RGB-D camera or a depth estimation method), producing semantically annotated point clouds that are the input to the mapping framework. Voxeland models both geometric and semantic uncertainty, which is especially valuable in mobile robotics to enable informed decision-making. Note that, given the modularity of the proposed pipeline, the mapping tool (Voxeland) could be substituted by any alternative which takes the same input, provided it is able to accept categories not belonging to a pre-defined closed set.
- **Modification of the underlying mathematical formulation of the mapping module (Voxeland) for open-vocabulary semantics.** A key aspect of the current proposal is the ability to work with semantic maps which are not constrained to closed sets of categories, not only at the image processing stage, but also within the finalized map of the environment. The probabilistic approach that Voxeland utilizes to represent the semantic information requires special considerations in this regard in order to maintain formal coherence, which are discussed in Section 3.2.
- **Evaluation of open-vocabulary perception in semantic mapping.** We assess the impact of different perception front-ends on the quality of the resulting semantic maps, comparing TALOS against both traditional closed-vocabulary detectors and discriminative open-vocabulary models.
- **Exploration of alternative implementations for the modular components of TALOS.** Given that TALOS is a modular method in which state-of-the-art models can be swapped at each stage, we conduct the evaluations under different pipeline configurations, highlighting the flexibility of the modular architecture and its impact on mapping performance. The corresponding controlled quantitative ablation is detailed in Appendix C.

Regarding the evaluation of the proposed semantic mapping pipeline, the quality of the resulting maps was assessed on scenes from the ScanNet v2 dataset [32] containing a wide variety of object instances. For comparative evaluation, alternative detection front-ends are also evaluated: the original Detectron2/Mask R-CNN closed-vocabulary baseline used in Voxeland [14,33], and the discriminative open-vocabulary detector YOLOE [31]. Across five independent runs, TALOS achieves a mean Average Precision (mAP@0.5) of 0.1607 ± 0.0255, outperforming YOLOE (0.0834) and Detectron2/Mask R-CNN (0.0453) in every run. Additionally, qualitative experiments were carried out to assess different aspects of the evaluated methods including map completeness, geometric precision, semantic coherence and instance separation, showing that incorporating TALOS enhances the overall quality of the generated maps. These results show that generative open-vocabulary perception, grounded through modular tagging, location and segmentation, provides a consistent improvement in 3D semantic mapping quality under the evaluated protocol compared with both closed-vocabulary and discriminative open-vocabulary alternatives.

2. Related Works

2.1. *Open-Vocabulary Instance* Segmentation

Instance segmentation aims to delineate each individual object in an image with a separate pixel-accurate mask and a categorical label, going beyond semantic segmenta-

tion, which assigns a single class per pixel without distinguishing individual instances. Closed-vocabulary architectures such as Mask R-CNN [14], Hybrid Task Cascade [34], and Mask2Former [35] have driven significant progress, yet they are trained on fixed label sets, typically those of benchmark datasets such as Microsoft COCO [17], and cannot recognize categories absent from their training vocabulary.

To overcome this closed-set constraint, several methods condition segmentation on textual descriptions of the target categories. OpenSeg [25], X-Decoder [36], Grounded SAM [37], and SAM 3 [38] leverage vision–language alignment to segment objects matching user-provided text or visual prompts. However, they still require manual specification of which categories to detect, which is impractical for autonomous systems that must operate without human intervention.

An alternative line of work explores prompt-free open-vocabulary semantic segmentation, with methods such as TAG [39] and ZeroSeg [40]; however, these produce semantic rather than instance-level predictions. More recently, discriminative open-vocabulary instance segmentation models such as OpenSeeD [41], FC-CLIP [42], and YOLOE [31] combine category-agnostic detection with open-set recognition in a single architecture. Although effective, their behaviour is difficult to customize without retraining. Under the evaluated configuration, YOLOE produces more evaluable false positives than the closed-vocabulary Detectron2 baseline and qualitatively more cluttered 3D maps than both Detectron2 and TALOS.

The rapid development of Large Language Models (LLMs) and Large Vision–Language Models (LVLMs) motivates the exploration of generative approaches in which language models actively participate in the perception pipeline, leveraging contextual reasoning to guide open-vocabulary scene understanding.

2.2. Language-Driven 3D Scene Understanding

The success of LLMs and LVLMs has sparked a growing trend toward integrating generative language models into 3D scene understanding, enabling tasks ranging from open-vocabulary querying to embodied planning. A first line of research builds open-vocabulary 3D representations: OpenScene [7] projects dense CLIP features into 3D point clouds for open-vocabulary queries; ConceptFusion [26] fuses multi-modal foundation model features across views into a single 3D map supporting open-set retrieval; and ConceptGraphs [8] constructs open-vocabulary 3D scene graphs with object-level nodes and spatial-relation edges suited for both perception and planning.

A second direction applies LLMs and LVLMs directly to 3D visual understanding: 3D-LLM [43] injects point-cloud features into a large language model for 3D question answering, captioning, and grounding, while LL3DA [44] accepts raw point clouds together with visual and textual prompts for interactive instruction-following. Beyond passive understanding, language-driven approaches are also employed for embodied planning and navigation: SayPlan [45] grounds LLM-based task planning in hierarchical 3D scene graphs; VoxPoser [46] composes 3D value maps for zero-shot manipulation; and SG-Nav [47] uses hierarchical chain-of-thought prompting over online scene graphs for zero-shot object-goal navigation.

Collectively, these developments show that large generative models enable more flexible 3D perception than purely discriminative approaches, motivating pipelines, such as the one proposed here, that actively leverage language models for semantic inference.

2.3. Instance-Aware 3D Semantic Mapping

Instance-aware 3D semantic mapping builds spatial representations in which each individual object is identified, segmented, and labelled, enabling higher-level tasks such

as object counting, spatial relation extraction, and targeted manipulation. A substantial body of work addresses this problem by coupling a volumetric or point-cloud back-end with a 2D instance segmentation front-end. Voxblox++ [48] extends the Voxblox TSDF framework with per-instance reconstruction driven by Mask R-CNN; Wang et al. [49] fuse multi-view Mask R-CNN predictions for indoor 3D object detection; Li et al. [50] propose an incremental instance-oriented mapping pipeline from RGB-D streams; Mascaro et al. [51] introduce a label diffusion strategy that propagates 2D instance predictions into a 3D volumetric representation; Miao et al. [52] present a panoptic mapping system using Mask2Former [35]; and Voxeland [13] further advances this line by incorporating probabilistic evidence fusion with uncertainty quantification.

A common characteristic of all these systems is their reliance on closed-vocabulary perception models—whether Mask R-CNN [14] trained on the 80 Microsoft COCO categories or Mask2Former with a similarly fixed label set—bounding the semantic vocabulary of the resulting 3D map to the categories seen during detector training. Despite the advances discussed in Sections 2.1 and 2.2, the adoption of open-vocabulary detectors as front-ends for instance-aware 3D mapping remains largely unexplored. This gap motivates the present work, which integrates an open-vocabulary, language-driven perception pipeline into a state-of-the-art mapping framework to overcome the closed-vocabulary bottleneck.

3. Proposed Method

The main contribution of this work is the proposal of a complete semantic mapping pipeline (image to map) that is able to construct open-vocabulary representations of the environment. Our method utilizes a modular workflow with two main stages (see Figure 1):

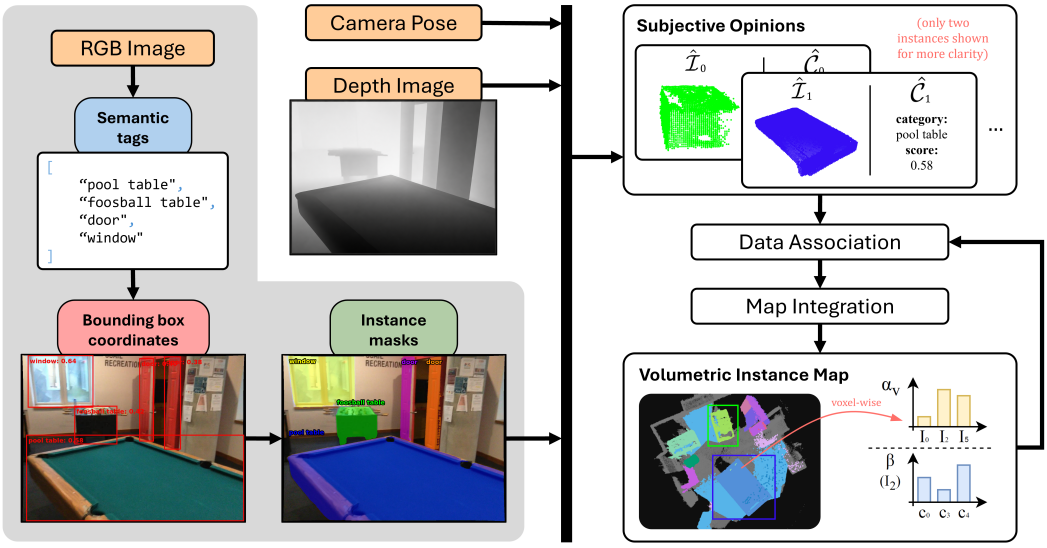


Figure 1. Overview of the complete pipeline, from RGB-D perception to probabilistic volumetric instance mapping. Briefly, the images captured by the camera are processed to extract semantic instances and their corresponding geometry as annotated point clouds. These point clouds are then fed into the mapping module and integrated into the probabilistic map as individual opinions.

- **Image processing**: The RGB image captured by the sensor is processed to extract and identify semantic instances. Our proposed pipeline uses TALOS to carry out this step (see Section 3.1), thus avoiding the object detections being limited to a closed set of categories. Still, given the modularity of our proposed approach, a different semantic segmentation tool may be used, as seen in the experiments presented in Section 4. Once the objects in the image have been identified and segmented, the corresponding

per-instance masks are combined with the depth readings to generate a semantically annotated 3D point cloud, which becomes the input to the mapping module.

- **Map integration and update**: To build a coherent map of the environment, each new observation is integrated with all previous information. This process is handled in our proposed pipeline by Voxeland (see Section 3.2). Voxeland [13] is a voxel-based semantic mapping framework that handles the integration of successive observations through a probabilistic approach, thus allowing for an explicit quantification of uncertainty and offering increased robustness to spurious data.

The following sections explore each of these stages in more detail.

It should be noted that, given the computational cost associated with the inference of LLMs and LVLMs, the proposed pipeline is not designed to be run online as the robotic agent explores the environment. Instead, the semantic reconstruction of the environment must take place after the robot has finished its initial exploration as a separate, offline process. It is only once this representation has been built that the robotic agent may exploit it to plan its relevant tasks, which represents a limitation of our proposal.

3.1. Open-Vocabulary Perception

This section describes TALOS (TAgging–LOcation–Segmentation), the open-vocabulary perception method incorporated in the proposed semantic mapping pipeline for semantic object instance segmentation. Given an RGB image, the method outputs a set of object instance detections, each consisting of (i) a pixel-accurate binary mask and (ii) an associated semantic label expressed as text. Unlike closed-set instance segmentation pipelines, the semantic output is not restricted to a pre-defined list of categories, but can be drawn from an open vocabulary. As illustrated in Figure 2, TALOS follows a sequential three-stage design: (i) **Tagging**, which estimates the set of semantic categories present in the image; (ii) **Location**, which localizes candidate instances for each category via bounding boxes; and (iii) **Segmentation**, which produces a precise mask for each localized instance.

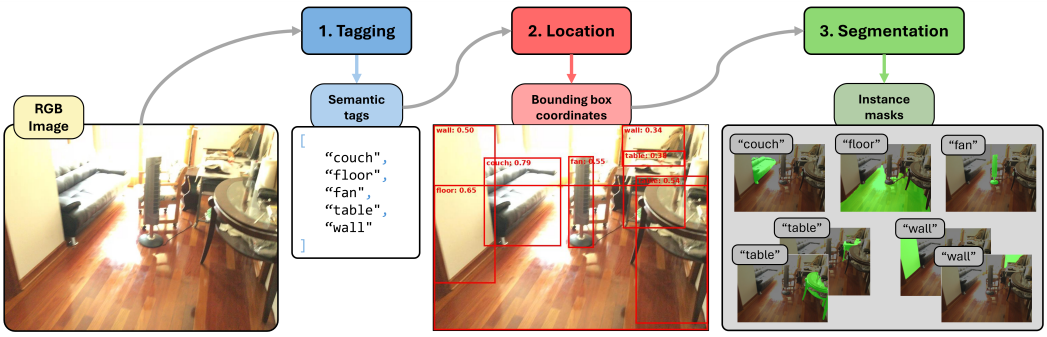


Figure 2. General diagram of the open-vocabulary semantic object instance segmentation method.

A key feature of TALOS is its modularity. Each stage is conceptually and programmatically decoupled from the others, enabling the use of different state-of-the-art models for each subtask and facilitating evaluations and upgrades. The following subsections detail the three stages and discuss alternative implementations for Tagging.

3.1.1. Tagging

The purpose of Tagging is to infer, from an input image, a set of textual labels that describe the object categories present in the scene. Formally, we define

$$\mathcal{T}(I) = L_T \tag{1}$$

where the input is an RGB image $I \in \mathbb{R}^{H \times W \times 3}$ and the output is a set of textual tags $L_T = \{t_1, t_2, \ldots, t_n\}$ such that $t_i \in V$, with V denoting an open vocabulary. The cardinality of L_T depends on the scene content and on the desired operating point (e.g., exhaustive tagging vs. listing only the most salient categories).

We consider three alternative implementations for the Tagging stage. The first two follow the generative design that motivates TALOS, while the third one illustrates that, due to the modularity of the pipeline, a discriminative open-vocabulary tagger can also be inserted as an efficiency-oriented variant.

(a) **Direct Tagging with a Large Vision–Language Model (LVLM).**
In the most direct generative configuration, a Large Vision–Language Model (LVLM) is queried with a natural-language prompt to produce the tag list directly, as illustrated by the first workflow in Figure 3. This approach is particularly flexible: the prompt can inject task and domain context (e.g., indoor environments), specify the expected granularity (e.g., "furniture" vs. "armchair"), and enforce a machine-readable output format (e.g., JSON with a list of strings). Moreover, prompts can include explicit normalization rules to encourage consistency across images, such as using singular nouns, preferring canonical category names over descriptive phrases, and avoiding fine-grained attributes unless required.

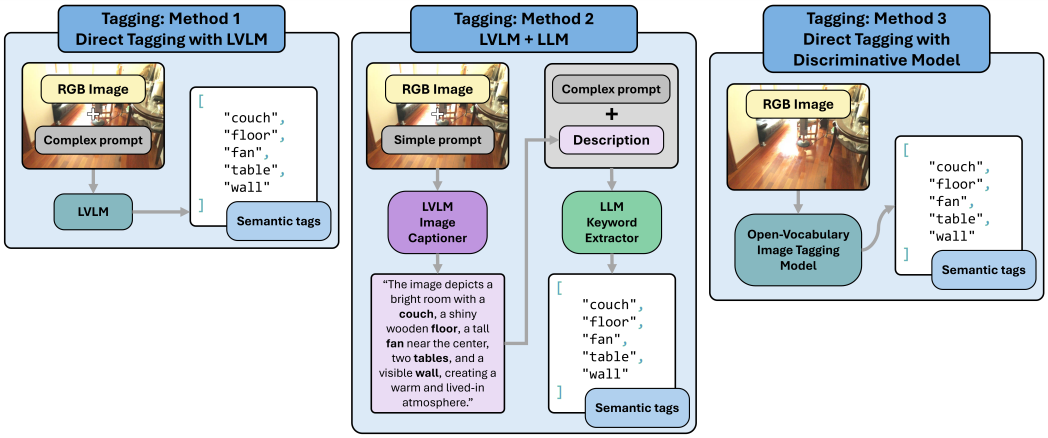


Figure 3. Overview of the three Tagging implementations considered in TALOS. The panels correspond, in order, to direct tag generation with an LVLM, a two-stage generative pipeline that first obtains an LVLM caption and then extracts structured object keywords with an LLM, and direct open-vocabulary multi-label prediction with a discriminative tagger. All three variants produce the semantic tag set L_T used by the subsequent Location and Segmentation stages. The plus sign indicates that both the image and the prompt are provided as input.

Because LVLMs generate free-form text, careful prompt design is important to reduce formatting errors and semantic drift. Since strict structure is required, the prompt can instruct the model to output only the list and nothing else, and to avoid synonyms or descriptive modifiers.

To further improve quality, TALOS can apply self-reflection [3]: one or more additional LVLM queries that provide the model with its previous answer and ask it to verify compliance with the requested format and to revise the tag set if inconsistencies are detected. This mechanism is particularly useful for removing redundant labels, correcting pluralization, and improving semantic coherence in the predicted set. The complete prompt used in this stage is provided in Appendix A.1.

(b) **Tagging via LVLM Captioning + LLM Keyword Extraction.**
Certain LVLMs, especially lightweight ones, may struggle to reliably follow complex instructions in the input prompt [53,54]. To mitigate this limitation, we consider the

two-stage strategy shown as the second workflow in Figure 3. First, an LVLM is used as an image captioner, producing a free-form description of the scene with a simple prompt (e.g., "Describe the image with detail."). Second, a Large Language Model (LLM) extracts the tag set L_T from the caption using a more restrictive prompt that enforces structure (e.g., JSON), normalization (e.g., singular nouns), and semantic constraints (e.g., output only object categories).

This decomposition separates visual understanding (caption generation) from instruction following (keyword extraction), enabling the use of lighter multimodal models in the first step while still obtaining a controlled structured output. As in the previous Tagging method, techniques such as self-reflection [3] can be applied to the keyword extraction stage to refine the final tag list. The complete prompt used for the LLM keyword extraction sub-stage is provided in Appendix A.2.

(c) **Direct Tagging with a Discriminative Open-Vocabulary Model.**
Although TALOS is primarily conceived as a generative, instruction-conditioned method, its modular structure also permits replacing the generative Tagging stage with a discriminative open-vocabulary image-to-tags model, such as RAM++ [55]. This third workflow is included in Figure 3 as an efficiency-oriented alternative. In this configuration, the tag set L_T is obtained through discriminative multi-label prediction: given an image, the model outputs a list of semantic labels and, typically, an associated confidence score for each label. This alternative is faster at inference time and considerably lighter in terms of GPU memory and computation than LVLM-based approaches, as it only requires a single forward pass through a compact network.
However, this configuration should be understood as a possible but not preferred realization of TALOS. This interpretation is also supported by the controlled ablation in Appendix C: although RAM++ provides the lowest latency, both evaluated generative representatives achieve higher 3D mapping performance. This result is consistent with the argument advanced in this paper that generative, prompt-conditioned models can be customized to the specific mapping task and can consequently achieve stronger mapping performance. The central motivation of TALOS is to exploit the semantic reasoning, prompt conditioning, and instruction-following capabilities of generative multimodal models, which make it possible to adapt the Tagging behaviour to the mapping task and to impose explicit constraints on the output vocabulary and format. By contrast, direct discriminative taggers offer limited task-specific customization and are therefore less aligned with this motivation. Moreover, although models such as RAM++ are commonly referred to in the literature as open-vocabulary or open-set recognition models, their standard use relies on prediction over a large predefined list of categories rather than on the free-form generation of labels from a genuinely open vocabulary, as in LVLM-based Tagging. Consequently, the direct discriminative Tagging strategy is mainly useful as an efficiency baseline, or for deployments in which reducing prediction time is substantially more important than operating over a genuinely open and task-adaptable category space.

3.1.2. Location

Given the input image and the tag set produced by the previous stage, Location identifies image regions corresponding to the candidate categories by predicting bounding boxes and localization confidences (see Figure 4):

$$\mathcal{L}(I, L_T) = L_L \tag{2}$$

where $L_L = \{l_1, l_2, \ldots, l_m\}$ is a list of localized instances and each element $l_i = (label_i, score_i, bbox_i)$ comprises (i) the category of instance i as $label_i \in L_T$, (ii) a confidence $score_i \in [0, 1]$, and (iii) its bounding box $bbox_i = (x_{\min}, y_{\min}, x_{\max}, y_{\max}) \in \mathbb{R}^4$.

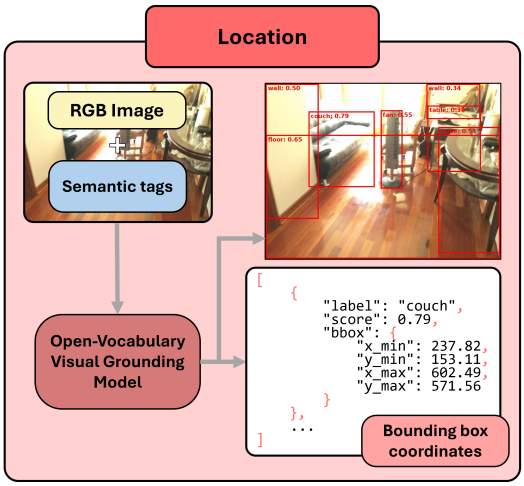


Figure 4. Overview of the Location stage. Given an image and a set of candidate tags from Tagging as inputs (joined by the plus sign), the visual grounding model predicts category-conditioned bounding boxes and associated confidence scores.

This stage is implemented with an open-vocabulary visual grounding model, which aligns text queries with image regions and returns category-conditioned detections. In practice, models such as Grounding DINO [29] take the input image and a list of textual queries (the tags in L_T) and output a set of bounding boxes per query, along with confidence scores indicating how strongly each box content matches the category. The output L_L can be filtered via confidence thresholding and by applying overlap-aware heuristics to reduce duplicated detections, e.g., avoiding multiple highly overlapping boxes that are erroneously assigned to the same underlying object instance before passing them to the Segmentation stage.

Importantly, the ability of open-vocabulary visual grounding models to localize object instances beyond the categories seen during training constitutes a major advantage of this stage, as it provides the overall pipeline with increased semantic richness and a broader scene understanding.

3.1.3. Segmentation

The Segmentation stage produces pixel-level instance masks for the localized boxes (see Figure 5):

$$\mathcal{S}(I, L_L) = L_S \tag{3}$$

where $L_S = \{m_1, m_2, \ldots, m_m\}$ and each $m_i \in \{0, 1\}^{H \times W}$ is a binary mask. There is a one-to-one correspondence between elements in L_L and L_S, ensuring that each localized instance has exactly one associated mask.

This stage is implemented using category-agnostic instance segmentation foundation models that can generate high-quality masks from spatial prompts (e.g., bounding boxes) without requiring the semantic class as input. One example of such segmentation models is SAM2 [28].

Finally, the TALOS pipeline outputs, for each instance i, the tuple $(m_i, label_i, bbox_i, score_i)$, combining pixel-level geometry (mask), semantics (label), and localization confidence (bounding box and score) in a compact and reusable representation.

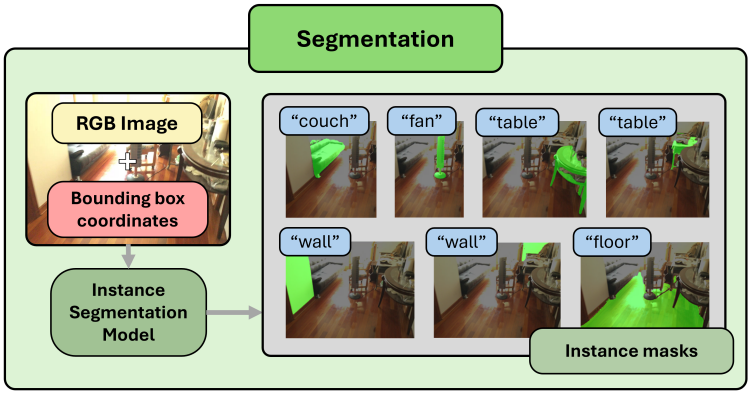


Figure 5. Overview of the Segmentation stage. Each bounding box produced by Location is used as a prompt for a category-agnostic foundation model to output a pixel-accurate instance mask. The plus sign indicates that both the image and the bounding box coordinates are provided as input.

3.2. Creation of the Semantic Map

Once the instances appearing in the image have been identified, categorized, and scored, they can be integrated into a persistent semantic map of the environment. In order to perform this integration, the extracted image-space segmentation masks are combined with depth information to generate semantically annotated 3D point clouds. This depth information can be obtained directly from the sensor in the case of RGB-D cameras, or estimated with machine learning techniques if only RGB information is available. Each projected point is annotated with the semantic information generated by TALOS, including its instance identifier, semantic category, and associated confidence score.

This 3D point cloud, corresponding to a single observation, can then be integrated into a persistent semantic map. A straightforward approach would be to represent the map as the mere accumulation of all previously observed point clouds. However, this strategy presents significant limitations. Beyond the inefficiency of such a representation, simple point cloud accumulation does not support effective data association, which can lead to issues such as instance duplication when previously mapped objects are observed again. To overcome these limitations, a more sophisticated mapping framework is required—one capable of addressing the data association problem and consistently fusing new observations into an existing semantic representation.

In this work, we build upon Voxeland [13], a mapping framework based on dense voxel maps for the representation of geometric and semantic information, while optionally maintaining a sparse instance-level layer. A key feature of Voxeland is its probabilistic integration of multiple observations, enabling explicit uncertainty quantification throughout the mapping process. In the following subsections, we briefly outline the core probabilistic formulation of Voxeland and describe the extensions introduced in this work to enable the open-vocabulary capabilities of our proposed pipeline.

3.2.1. Probabilistic Formulation

The semantic maps generated by Voxeland do not assign a single category to each voxel. Instead, they explicitly account for the fact that multiple observations of the same region may provide inconsistent evidence, and fuse such observations within a probabilistic framework. These inconsistencies can arise from several sources, including robot localization errors, noise in the depth measurements—which may cause 3D points to be incorrectly projected into the map—and failures in the semantic classification or segmentation of the RGB images, among others.

Specifically, Voxeland models three key quantities probabilistically:

- Occupancy: For each voxel, Voxeland maintains the probability of the voxel being occupied conditioned on the set of observations z, i.e., $p(o \mid z)$. This probability is estimated through binary Bayes filtering using the depth measurements acquired from each robot pose.
- Instance category: Each instance on the map has categorical probability distribution over the list of possible object classes. In the unmodified version of Voxeland, this refers to a closed set of categories C. The probability of instance k belonging to class c given the list of observations z that have been collected, $p(i_k = c|z)$, is estimated from a Dirichlet distribution [56] with a set of evidence concentration parameters $\beta = \{\beta_i \,|i \in C\}$. The expression that relates the conditional probability to the Dirichlet distribution is as follows:

$$p(i_k = c|\, z) = \frac{\beta_c}{\sum_{i \in C} \beta_i} \tag{4}$$

- Instance geometry: Lastly, each voxel has a certain probability of belonging to each of the instances in the set I^{obj}, which contains all instances identified in the map so far. Similarly to the semantic classification of the individual instances, this corresponds to a discrete probability distribution which is derived from a Dirichlet distribution. In this case, the distribution describes the probability that voxel j belongs to instance k given observations z—i.e., $p(v_j = k|z)$. The corresponding Dirichlet distribution is described by a parameter vector $\alpha = \{\alpha_i | i \in I^{obj}\}$, and the conditional probability is calculated from these parameters with the following expression:

$$p(v_j = k|z) = \frac{\alpha_k}{\sum_{i \in I} \alpha_i} \tag{5}$$

The concentration parameters of the Dirichlet distributions (α and β) are initialized uniformly for all values in the corresponding distribution, and increased when the corresponding voxel or instance is re-observed, as well as when instances are merged. For details of this update process, which are beyond the scope of this discussion, refer to [13].

3.2.2. Open Vocabulary Considerations

Although Voxeland was not originally designed for open-vocabulary semantic representations, its core probabilistic formulation can be integrated into the proposed pipeline with only minor modifications.

The main adaptation concerns the probabilistic treatment of the accumulated evidence associated with the object category of a given instance (β). In the original formulation of Voxeland, the category probabilities $p(i_k = c \mid z)$ remain strictly positive, even when instance k has never been observed as category c. This reflects the possibility of misclassification and preserves uncertainty over alternative semantic hypotheses, which is a fundamental aspect of Voxeland's probabilistic reasoning framework.

From an implementation perspective, this modification affects the construction and update of the Dirichlet parameter vector β, which models the distribution over object categories for each instance. In the original formulation of Voxeland, all category parameters are initialized at startup with the same constant value, $\beta_i = \epsilon \; \forall i$. This symmetric prior acts as a pseudocount over the finite category set, preventing $p(i_k = c \mid z)$ from collapsing to zero simply because category c has not yet been observed.

In an open-vocabulary setting, however, this strategy is no longer feasible. If the domain of the variable i_k (i.e., the object category associated with instance k) is considered to be an unbounded set C, it becomes impossible to initialize all unseen categories with a non-zero uniform prior mass while maintaining a valid probability distribution.

A straightforward alternative would be to assign zero probability to all unobserved categories, i.e., $p(i_k = c \mid z) = 0$, or equivalently $\beta_i = 0$, whenever category c has never been observed for instance k. However, this formulation fails to capture uncertainty in the classification process. For example, if instance k has only been observed once, the resulting posterior would assign $p(i_k = c_i \mid z) = 1$ to the observed category c_i and $p(i_k = c_j \mid z) = 0$ to every other category c_j. Such an overconfident estimate is undesirable, not only for downstream map exploitation, but also for the mapping process itself.

This issue becomes particularly relevant during instance association. When deciding whether two instances should be merged, Voxeland considers a semantic similarity metric based on their category distributions as one of the decision factors. Under the zero-probability formulation, if instance A assigns zero probability to the category observed for instance B, the similarity metric would strongly discourage merging them. However, if the classification of A is supported by only a few observations, substantial uncertainty still remains, meaning both instances could in fact correspond to the same physical object.

To address this issue, we propose defining a finite subset of relevant categories $C' \subset C$, thus allowing the evidence vector β to be interpreted as a probability distribution over C', rather than over the potentially unbounded category space C. In the most straightforward case, C' corresponds to the set of all categories observed so far during the mapping process, although this definition can be naturally extended to incorporate any other categories deemed relevant at a given time. For instance, categories that are not currently represented in the map, but appear in a user query or are known to be important for a particular downstream task, can also be incorporated into C'. Crucially, this subset can be redefined dynamically as new observations or semantic requirements arise, thereby preserving the flexibility and open-vocabulary capabilities of the proposed pipeline.

From an implementation perspective, whenever a new category c is introduced, its corresponding Dirichlet parameter β_c is appended to the evidence vector and initialized to zero. The computation of the posterior category distribution for a given instance (Equation (4)) is then modified as follows:

$$\begin{aligned} \beta'_i &= \beta_i + \frac{\epsilon}{|C'|}, \quad \forall i \in C' \\ p(i_k = c \mid z) &= \frac{\beta'_c}{\sum_{i \in C'} \beta'_i} \end{aligned} \tag{6}$$

where the prior mass ϵ is uniformly distributed over the active category set C'.

A limitation of the current approach, which remains an open avenue for future work, is that operating over an open-vocabulary semantic space can lead to excessive semantic fragmentation. In particular, the object detection stage may assign different natural-language labels to multiple observations of the same object, even when those labels are synonymous or semantically equivalent. In such cases, the mapping framework may fail to recognize this semantic equivalence, negatively affecting instance association and fusion. While this issue can be partially mitigated through careful prompt engineering of the LVLM, or by refining the extracted tags using the techniques described in Section 3.1.1, the use of unconstrained natural-language identifiers inherently introduces ambiguity when comparing semantically related categories.

A promising direction to address this issue is the use of semantic embedding models [57] to identify labels that correspond to semantically equivalent or closely related concepts. This could be integrated directly into the mapping framework, or alternatively applied as a pre-processing stage over the point cloud. A thorough exploration of this approach would require additional analysis beyond the scope of the present work, and is therefore left as future research.

4. Evaluation Methodology and Results

This section evaluates the performance of the proposed open-vocabulary semantic mapping pipeline. First, we detail the experimental setup, the selected dataset, and the alternative perception baselines used for comparison. Next, we define the metrics employed to assess the accuracy of the generated 3D semantic maps. Finally, we present a comprehensive comparative analysis divided into two parts: a quantitative evaluation of the mapping performance, and a qualitative assessment focusing on the geometric quality, semantic coherence, and instance separation of the reconstructed scenes.

4.1. Experimental Setup and Datasets

All 3D semantic mapping experiments were conducted on ScanNet v2 [32], a large-scale RGB-D video dataset of indoor scenes comprising 2.5 million views across 1513 scans, each annotated with 3D camera poses, surface reconstructions, and fine-grained per-point semantic and instance labels. Compared to the original ScanNet v1, ScanNet v2 substantially expands and refines the semantic vocabulary, offering a broader and more realistic set of object categories.

For evaluation, the complete ScanNet v2 dataset was visually screened before running any of the evaluated detectors. Scenes were shortlisted to cover heterogeneous indoor settings and functional contexts, such as living rooms, bathrooms, and recreational rooms, with the aim of increasing the diversity of objects and spatial arrangements across the evaluation set. The first 11 shortlisted scenes were retained because they collectively satisfied the sole mandatory numerical criterion for accepting the subset: their mean number of ground-truth semantic categories per scene had to exceed the ScanNet v2 mean. The resulting subset contains 26.1 categories per scene on average, compared with 16.6 across all 1513 scans. Although purposive rather than statistically representative, the subset therefore provides a semantically rich evaluation set whose selection was independent of detector performance. The scene IDs are listed alongside the per-scene evaluation results below. For each scene, a 3D semantic map was built with Voxeland for each baseline and five independent maps were built with TALOS; all resulting maps were compared against the ground-truth 3D instance annotations provided by ScanNet v2. The selected scenes collectively contain 530 ground-truth object instances distributed across 119 semantic categories.

The detector configurations evaluated within the Voxeland 3D semantic mapping framework are:

- **Detectron2** [33]: an instance segmentation system based on Mask R-CNN [14] trained on the Microsoft COCO dataset, representing the standard closed-vocabulary perception front-end previously integrated in Voxeland.
- **TALOS** (proposed): the open-vocabulary semantic instance segmentation pipeline described in Section 3.1. For these experiments, the pipeline was configured using Direct LVLM Tagging with MiniCPM-o 2.6 [58] for the Tagging stage, Grounding DINO [29] for Location, and Segment Anything Model 2 (SAM2) [28] for Segmentation. A comprehensive list of the alternative models implemented for each TALOS stage is provided in Appendix B. This configuration was evaluated through the controlled quantitative component-and-configuration ablation detailed in Appendix C. The ablation held Location and Segmentation fixed and compared all 16 implemented Tagging configurations through two increasingly demanding LVIS-based stages followed by evaluation within Voxeland. Its final Voxeland phase yielded global mAP@0.5 values of 0.1799 for Direct LVLM Tagging with MiniCPM-o 2.6, 0.1525 for the two-stage MiniCPM-o 2.6 $\rightarrow$ MiniCPM-o 2.6 configuration, and 0.0685 for Direct Tagging with RAM++, thereby supporting the Direct LVLM configuration used in the experiments below.

- **YOLOE** [31]: a real-time open-vocabulary detection and segmentation model that unifies text-prompt, visual-prompt, and prompt-free detection within a single highly efficient architecture.

This three-way comparison isolates the contribution of vocabulary openness and model expressiveness: Detectron2 represents the closed-vocabulary paradigm, YOLOE a fast discriminative open-vocabulary alternative, and TALOS a generative, instruction-conditioned open-vocabulary pipeline. It should be noted that, given its role as a baseline, we employ YOLOE in its most basic form for the evaluation: running in prompt-free mode, with the default confidence and NMS IoU thresholds (0.25 and 0.7, respectively). Thus, the results obtained below do not correspond to a configuration optimized for any given application.

Because the MiniCPM Tagging stage is stochastic, the complete TALOS map-generation and evaluation procedure was repeated five times. MiniCPM was invoked through Ollama v0.7.1 using its default decoding parameters: temperature $= 0.8$, top-$p = 0.9$, and top-$k = 40$. TALOS results are reported as the arithmetic mean and sample standard deviation across these five runs, whereas the Detectron2 and YOLOE results correspond to single runs.

The experiments were carried out on an edge computer equipped with three NVIDIA RTX 6000 Ada Generation GPUs, each with 48 GB of VRAM, an AMD Ryzen Threadripper PRO 7975WX CPU with 32 cores running at 4.0 GHz, and 515 GB of DDR5 RAM. However, only one of the three GPUs was used for the experiments reported in this work. The computer is governed by the CSAR system architecture [59], which provides LXC container orchestration to support isolated development and testing environments while effectively sharing hardware resources. CSAR also allows computationally demanding components, such as LVLMs and LLMs, to be deployed and executed on the edge computer in collaboration with the components running on the robot, thus not compromising the limited resources of robotic platforms.

The implementation of the proposed semantic mapping pipeline is publicly available at https://github.com/MAPIRlab/OpenVocabularyMapping (accessed on 10 September 2026) (For reproducibility, the software revisions used in the experiments were TALOS at commit `1ae0a7e`, Voxeland at commit `389e3ec`, Detectron2 at commit `536dc9d` from the original https://github.com/facebookresearch/detectron2 (accessed on 10 September 2026) repository, and YOLOE at commit `40cd606` from the original https://github.com/THU-MIG/yoloe (accessed on 10 September 2026) repository. These identifiers allow the evaluated software state to be recovered if any component evolves).

4.2. Evaluation Metrics

Performance is reported in terms of mean Average Precision at an IoU threshold of 0.5 (denoted mAP@0.5), a standard metric for instance segmentation that jointly assesses localization quality, class correctness, and detection completeness.

4.2.1. Point-Cloud IoU

Since Voxeland outputs 3D semantic maps as point clouds (centroids of occupied voxels) rather than 2D masks, standard pixel-wise IoU cannot be directly applied. Instead, the Intersection-over-Union between two instance point clouds is computed via KDTree radius search: for each point in the smaller cloud, a nearest-neighbour lookup is performed in the larger cloud within a search radius $r = 4v$, where $v = 0.02$ m is the voxelization resolution used for ground-truth preprocessing; a point is counted as matched if at least one neighbour is found within r. The IoU is then defined as

$$\mathrm{IoU}(P_1, P_2) = \frac{|P_1 \cap P_2|}{|P_1| + |P_2| - |P_1 \cap P_2|}, \tag{7}$$

where $|P_1 \cap P_2|$ denotes the count of matched points. To improve robustness, two complementary IoU variants are computed in parallel, one on the original resolution clouds and one on voxel-downsampled clouds with voxel size 0.05 m, and the maximum of the two is retained.

4.2.2. Instance Matching and TP/FP/FN Classification

Predictions are processed in descending order of confidence and greedily matched to the unmatched ground-truth (GT) instance that yields the highest IoU and has a consistent semantic label. Each prediction and GT instance can be matched at most once. A matched pair is classified as a true positive (TP) if and only if (i) IoU ≥ 0.5 and (ii) the predicted semantic label is consistent with the GT label. Class consistency is verified through an asymmetric synonym dictionary: a prediction is accepted for a given GT category if the predicted label either exactly matches the GT category or belongs to its predefined set of valid synonyms (e.g., the prediction `chair` is accepted when the GT label is `armchair`, since `chair` is a valid synonym of `armchair`; however, the reverse does not hold unless explicitly listed). The synonym dictionary is constructed by first traversing the ground-truth annotations of all evaluated scenes and extracting every unique category label that appears in them; these labels form the dictionary keys, and for each key a list of plausible alternative names is manually defined. This asymmetric design prevents overly permissive matching while allowing for common label variations between open-vocabulary models and dataset annotations. False positives (FPs) are evaluable predictions that remain unmatched to a GT instance at IoU ≥ 0.5, while false negatives (FNs) are GT instances that are left unmatched by any prediction.

The evaluation vocabulary is defined globally as the union of the 119 GT classes found across all evaluated scenes. An unmatched prediction that can be mapped to this vocabulary is counted as an FP even when its class is absent from the particular scene. Predictions that cannot be mapped to the global vocabulary are excluded from AP and reported separately as out-of-vocabulary predictions (OOV/Ign.). Consequently, the reported counts satisfy $N_{\mathrm{pred}} = N_{\mathrm{TP}} + N_{\mathrm{FP}} + N_{\mathrm{OOV}}$ and $N_{\mathrm{GT}} = N_{\mathrm{TP}} + N_{\mathrm{FN}}$.

4.2.3. AP and mAP Computation

Per-class Average Precision (AP) is computed using the VOC all-point interpolation method [18]: detections are sorted by decreasing confidence score, the precision–recall curve is monotonized by replacing each precision value with the maximum precision at any higher recall level, and the area is numerically integrated at every recall change point. For each class, detections and GT instances are pooled across all evaluated scenes before its AP is computed. The reported mAP@0.5 is the unweighted mean over all classes that have at least one GT instance in the evaluated scenes.

4.3. Quantitative Results

Tables 1 and 2 summarize the evaluation results obtained by running Voxeland with each of the three detector configurations on the 11 selected ScanNet v2 scenes. The TALOS entries summarize five runs, while the baseline entries correspond to single runs. Table 1 reports a single mAP@0.5 per detector computed by aggregating per-class APs across all scenes, whereas Table 2 shows the per-scene mAP@0.5 obtained by averaging per-class APs within each individual scene; consequently, the values in Table 1 are not the arithmetic mean of the per-scene values in Table 2.

Table 1. Detector ranking by mAP@0.5 of Voxeland maps built across 11 ScanNet v2 scenes (530 GT instances). Detectron2 uses Mask R-CNN in these experiments. TALOS values are the mean ± sample standard deviation over five runs; baseline values correspond to single runs. The best mAP@0.5 is highlighted in bold. TP: true positives; FP: false positives; OOV/Ign.: predictions outside the global evaluation vocabulary and ignored by AP; FN: false negatives.

Detector	mAP@0.5	GT Inst.	Pred Inst.	TP	FP	OOV/Ign.	FN
TALOS (ours)	**0.1607 ± 0.0255**	530	2703.6 ± 437.1	115.2 ± 8.4	2177.4 ± 425.6	411.0 ± 35.4	414.8 ± 8.4
YOLOE	0.0834	530	6515	65	1677	4773	465
Detectron2	0.0453	530	1556	42	1332	182	488

Table 2. Per-scene mAP@0.5 of Voxeland maps for each detector configuration on ScanNet v2 scenes. Detectron2 uses Mask R-CNN. TALOS values are the mean ± sample standard deviation over five runs; baseline values correspond to single runs. The best mean performance in each scene is highlighted in bold.

Scene	Detectron2	TALOS (Ours)	YOLOE
scene0000_01	0.1098	0.2096 ± 0.0166	**0.2222**
scene0002_01	0.0306	**0.2228 ± 0.0443**	0.1182
scene0034_00	0.0667	**0.1920 ± 0.0861**	0.1283
scene0051_01	0.1050	**0.1875 ± 0.0441**	0.0797
scene0115_02	0.0556	**0.1370 ± 0.0609**	0.0007
scene0126_00	0.1154	**0.2142 ± 0.0380**	0.1375
scene0233_00	0.0179	**0.1465 ± 0.0287**	0.0601
scene0404_00	0.0024	**0.1514 ± 0.0393**	0.0400
scene0646_00	0.0893	**0.1416 ± 0.0266**	0.1307
scene0648_01	0.1235	**0.1950 ± 0.0409**	0.1164
scene0696_01	0.0355	**0.1134 ± 0.0689**	0.0114

As shown in Table 1, TALOS achieves a mean mAP@0.5 of 0.1607 ± 0.0255 across five runs, compared with 0.0834 for YOLOE and 0.0453 for Detectron2; every TALOS run outperforms both baselines. On a semantically rich dataset such as ScanNet v2, open-vocabulary methods are expected to surpass a closed-vocabulary baseline like Detectron2, given its inherent category limitations from training exclusively on COCO. YOLOE produces 6515 instances, of which 1742 can be mapped to the global evaluation vocabulary: 65 are true positives and 1677 are false positives. The remaining 4773 predictions cannot be mapped to that vocabulary and therefore do not affect mAP. Consequently, the raw prediction count alone should not be interpreted as an error count, although the number of evaluable false positives indicates that over-generation affects precision under the evaluated configuration. Across its five runs, TALOS averages 115.2 ± 8.4 true positives and 2177.4 ± 425.6 evaluable false positives. Its mean false-positive count is therefore higher than those of both baselines; nevertheless, its higher number of true positives and confidence-ranked detections yields a higher mAP@0.5 in every run. As shown in Table 2, TALOS achieves the best mean mAP in 10 out of 11 scenes.

4.4. Qualitative Analysis

The quantitative evaluation is complemented by a qualitative analysis of the maps reconstructed with each detector configuration. The qualitative figures retain the TALOS maps used in the original submission, corresponding to the first of the five quantitative runs. The following comparison focuses on three properties that are particularly relevant for robotic scene understanding: the geometric quality of the objects integrated into the map, the semantic accuracy of their predicted classes, and the separation of individual object instances.

4.4.1. Geometric Quality of the Integrated Objects

Figure 6 provides an overview of the geometric characteristics of the resulting maps across three ScanNet v2 scenes. For clarity, each color in this figure represents a semantic category rather than an individual object instance; instance-level behavior is examined separately below. Detectron2 produces comparatively sparse maps with limited semantic information. This behavior is consistent with its closed-vocabulary nature: objects outside the restricted set of categories learned from Microsoft COCO cannot be represented in the reconstructed map. Under the evaluated configuration, YOLOE produces visually cluttered maps consistent with over-generation; this interpretation is based on its evaluable false positives and the qualitative evidence, rather than on treating all OOV predictions as errors. Its maps contain numerous scattered points, strongly overlapping categories, and diffuse object geometries, making the underlying scenes difficult to recognize. Neither extreme is desirable for robotic applications: sparsely populated maps omit useful information, while geometrically diffuse maps with scattered and overlapping categories are difficult to interpret and query reliably. TALOS provides a more balanced result. Its maps are more densely populated than those produced by Detectron2, while remaining substantially clearer and more interpretable than those obtained with YOLOE. Some mild point dispersion remains visible, but it is considerably less pronounced than in YOLOE despite both approaches operating with open vocabularies.

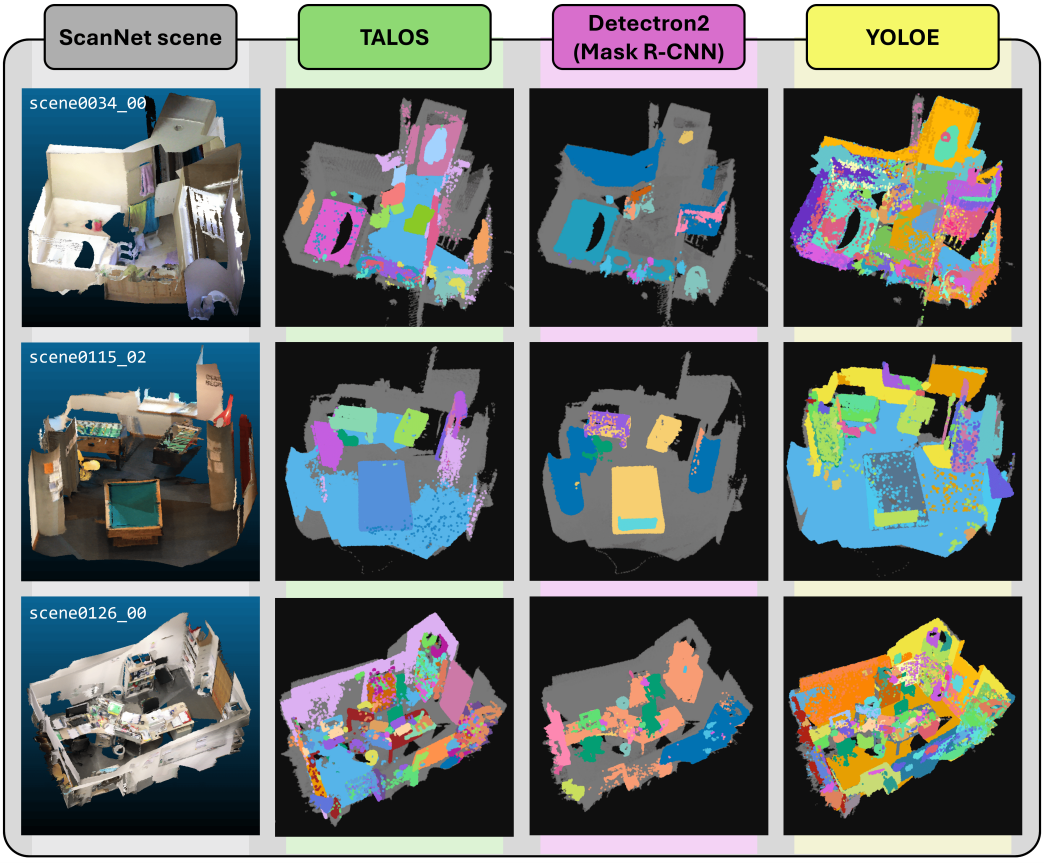


Figure 6. Geometric comparison of Voxeland maps reconstructed with TALOS, Detectron2 (Mask R-CNN), and YOLOE. The left column shows raw ScanNet v2 captures from three scenes, while the remaining columns show the corresponding reconstructed semantic maps. For visual clarity, each color represents a semantic category rather than an individual object instance.

4.4.2. Semantic Accuracy of the Integrated Objects

Geometric consistency alone is not sufficient for semantic mapping: the labels associated with the reconstructed objects must also remain coherent with the underlying scene. The quantitative results already show that TALOS achieves the highest mAP@0.5 when semantic correctness is taken into account. Figures 7 and 8 illustrate the practical impact of these differences in two representative scenes.

Figure 7 shows a ScanNet v2 scene containing a bathroom. TALOS reconstructs a semantically faithful representation of the scene, correctly identifying salient categories such as `bathtub`, `sink`, `toilet`, `floor`, and `rug`. Its remaining errors are comparatively minor: a small amount of point dispersion is visible around the bathtub, and objects located to the left of the actual scale are also assigned the `scale` category. Detectron2, in contrast, classifies the bathtub as a `sink` and assigns implausible categories such as `refrigerator` and `tv` to parts of the walls. YOLOE produces a substantially less coherent map, labeling walls with erroneous or overly generic categories such as `bathroom mirror` and `home interior`, assigning `kitchen floor` to the floor, and interpreting a large part of the bathtub as a `faucet`.

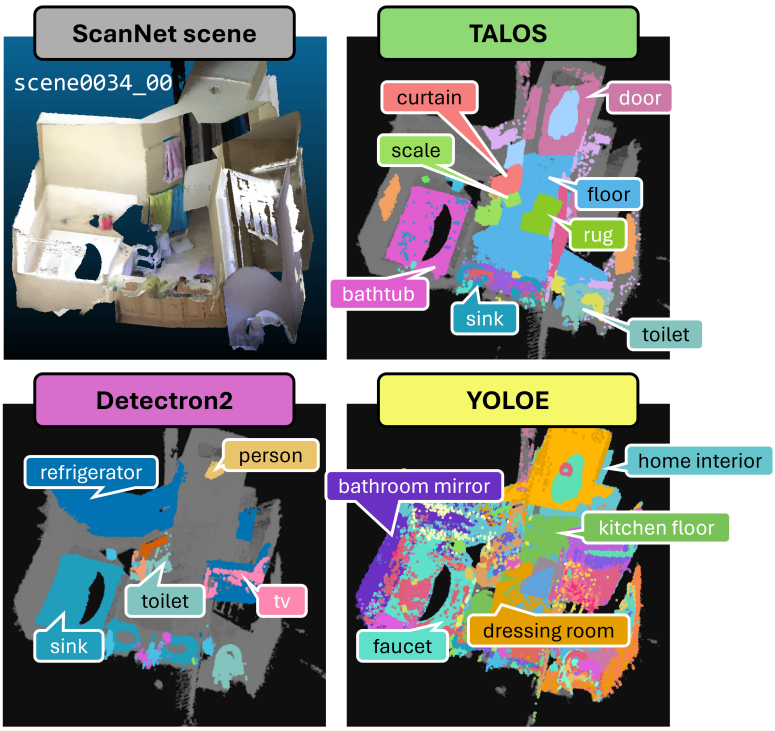


Figure 7. Semantic comparison for the bathroom in ScanNet v2 scene0034_00. The raw scene capture is shown alongside the maps reconstructed with TALOS, Detectron2 (Mask R-CNN), and YOLOE. Selected category labels are annotated to highlight representative semantic predictions. For visual clarity, colors group semantic categories rather than individual object instances.

Figure 8 shows a ScanNet v2 recreation room containing pool and foosball tables. TALOS again yields a semantically coherent map. Although one of the foosball tables is generalized as a `table` and a small amount of point dispersion remains visible on the right-hand side of the floor, the principal elements of the room are represented with plausible labels. Detectron2 correctly identifies only the chair among the highlighted objects: it labels the pool and foosball tables as `dining table` or `oven`, and assigns `refrigerator` to wall pillars. YOLOE once more generates a visually cluttered scene, assigning labels such as `bureau`, `martial arts gym`, `animation film`, and `information desk` to parts of the tables, as well as `dressing room` to walls. These outputs suggest that, when faced with ambiguous

visual content, YOLOE tends to favor generating a detection even when its semantic grounding is weak, rather than abstaining from an uncertain prediction. Together with the 4773 OOV predictions reported in Table 1, these qualitative errors suggest that YOLOE's high OOV count is driven at least partly by semantically incorrect or weakly grounded labels, rather than solely by valid, more specific descriptions. Since OOV predictions cannot be evaluated exhaustively against the available GT vocabulary, this observation should be interpreted as qualitative evidence rather than a direct measurement of their error rate.

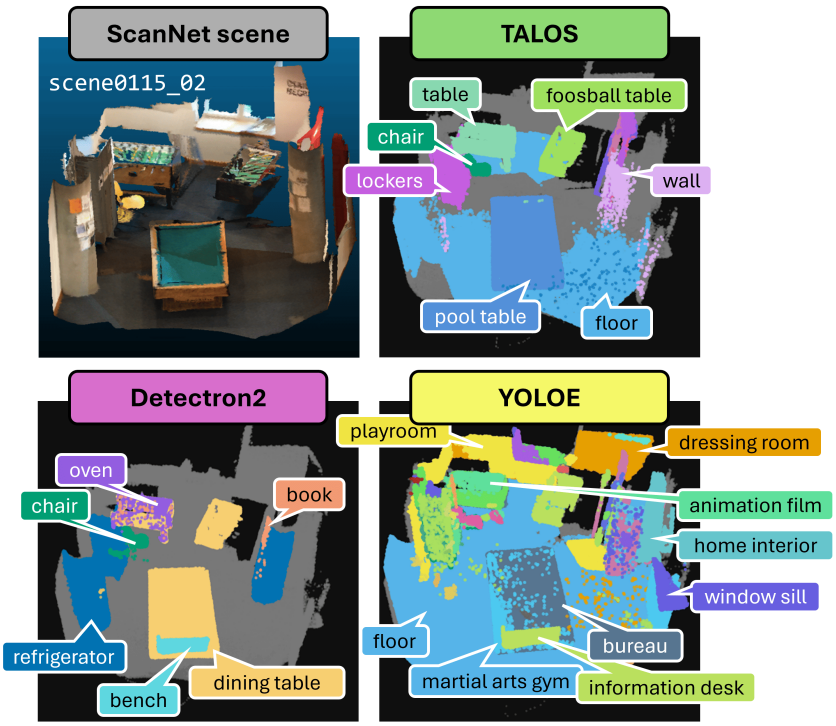


Figure 8. Semantic comparison for the recreation room in ScanNet v2 scene0115_02. The raw scene capture is shown alongside the maps reconstructed with TALOS, Detectron2 (Mask R-CNN), and YOLOE. Selected labels illustrate the semantic coherence of TALOS and representative classification errors from the baseline detectors. For visual clarity, colors group semantic categories rather than individual object instances.

The two examples also reveal a limitation of TALOS: some regions remain empty or undetected. Nevertheless, a conservative omission is generally preferable to a confidently incorrect semantic detection. This distinction is especially important in robotic applications, where failing to execute a command because a target object has not been detected is typically safer than executing that command on an incorrectly identified object.

4.4.3. Separation of Individual Object Instances

The practical value of the evaluated detectors extends beyond assigning categories to scene regions. Their instance-segmentation capabilities allow Voxeland to integrate individual object observations incrementally into the map. Consequently, useful robotic maps must not only preserve object geometry and semantic labels, but also distinguish separate occurrences of the same object type at different locations. Figures 9 and 10 focus on cropped portions of ScanNet v2 scenes to make this instance-level behavior easier to inspect. Unlike the previous figures, colors in these visualizations represent individual predicted instances.

Figure 9 revisits the bathtub region from the bathroom scene. The selected crop should ideally contain one principal instance, the bathtub, or at most two if the faucet is also represented separately. Detectron2 produces three instances, but assigns them incorrect categories. TALOS produces six instances and therefore still exhibits some over-segmentation; however, the dominant reconstruction is clearly the correctly labeled `bathtub`, while the residual `sink` and secondary `bathtub` instances contain comparatively few vertices. The `faucet` is also correctly represented as a separate object. YOLOE is markedly less balanced: it produces 90 instances within this small region, with no clear dominant prediction and with similarly sized fragments assigned to incorrect categories.

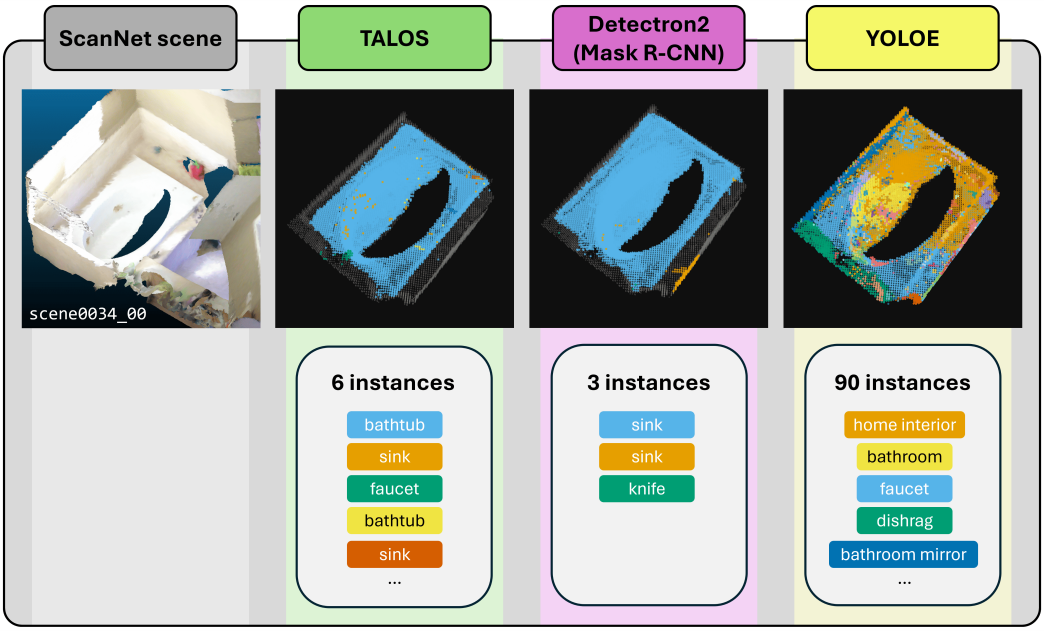


Figure 9. Instance-level comparison for a crop containing the bathtub from ScanNet v2 scene0034_00. The raw scene fragment is shown alongside the reconstructed instances produced by TALOS, Detectron2 (Mask R-CNN), and YOLOE. Each color represents a distinct predicted object instance. The category lists show representative instances and the total number of predicted instances within the selected crop.

Figure 10 shows a crop from a ScanNet v2 scene containing a desk, two chairs, and several electronic devices. In this case, TALOS predicts the smallest number of instances and provides high-quality dominant instances in both geometric and semantic terms. Detectron2 also obtains a comparatively reasonable reconstruction, although it labels the desk as a `mouse` and does not include the rug on the floor. YOLOE again produces a map with low visual clarity: its 139 predicted instances reflect substantial over-segmentation of the chairs, floor, and keyboard, alongside several erroneous categories.

Overall, the qualitative analysis supports the quantitative findings: among the three evaluated configurations, TALOS provides the most balanced trade-off between map completeness, geometric clarity, semantic coherence, and instance separation.

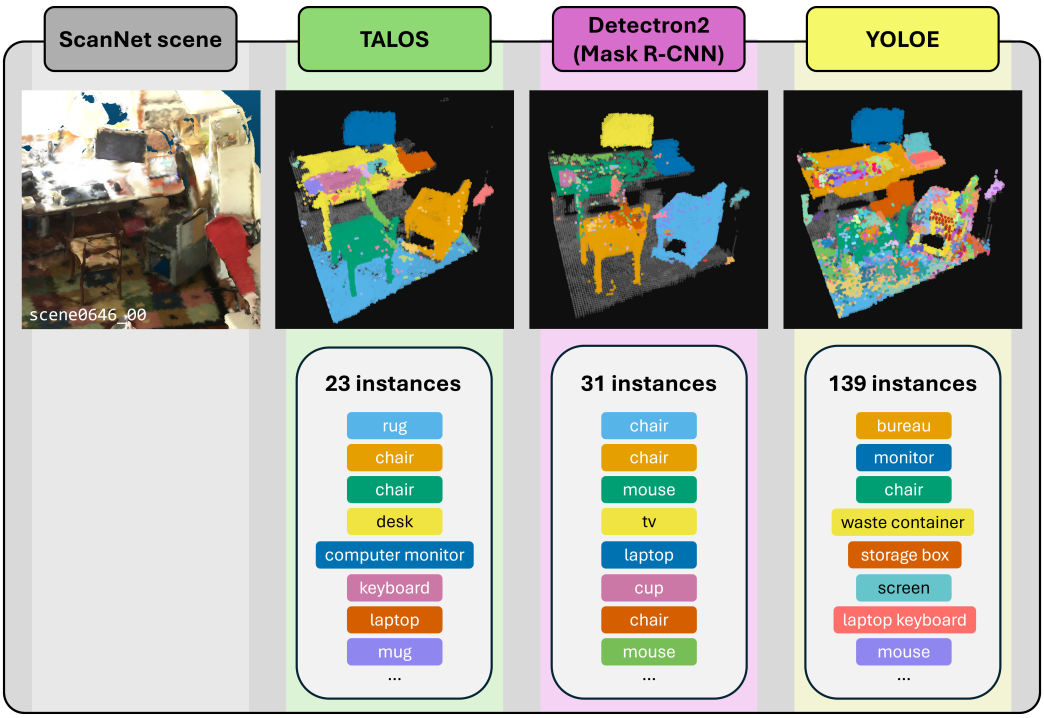


Figure 10. Instance-level comparison for a crop from ScanNet v2 scene0646_00 containing a desk, two chairs, and electronic devices. The raw scene fragment is shown alongside the reconstructed instances produced by TALOS, Detectron2 (Mask R-CNN), and YOLOE. Each color represents a distinct predicted object instance. The category lists show representative instances and the total number of predicted instances within the selected crop.

5. Conclusions

In this work, we presented an open-vocabulary pipeline for semantic mapping. While the pipeline itself is modular, the paper focused on an implementation based on TALOS and Voxeland. We discussed the operation of TALOS and how its three-step process (TAgging–LOcation–Segmentation) is employed to identify and classify object instances in RGB images without relying on a closed set of categories. We also discussed the adaptation of the Voxeland framework to the open-vocabulary pipeline.

Using this implementation, we presented both qualitative and quantitative results for the semantic maps reconstructed from scenes in the ScanNet dataset. Across all five runs, the TALOS-based implementation of the pipeline outperforms both a closed-vocabulary baseline and an alternative open-vocabulary implementation based on YOLOE. Nevertheless, its absolute mean mAP@0.5 of 0.1607 remains modest. Future work will therefore investigate task-specific prompt and configuration optimization, together with newer and more capable generative models, to improve mapping accuracy and increase the reliability of tasks that use semantic maps constructed with this technique.

Future work should also address how these open-vocabulary categories relate to one another, and include this reasoning in the mapping process. An interesting option would be to use semantic embeddings of the category labels as a way to identify related concepts. This also relates to the other main avenue for future research in this line, which is related to the exploitation and querying of the semantic maps. These mechanisms should be implemented in a way that takes advantage of the flexibility of the open-vocabulary categories, and semantic embeddings are a promising approach for this purpose, since they allow further context (locations, sizes, possible uses, etc.) to be included in a single identifier alongside the class label itself.

Author Contributions: Conceptualization, M.D.-G. and J.-R.R.-S.; methodology, M.D.-G.; software, M.D.-G. and P.O.; validation, M.D.-G.; formal analysis, M.D.-G. and P.O.; investigation, M.D.-G. and P.O.; data curation, M.D.-G.; writing—original draft preparation, M.D.-G., P.O. and J.-R.R.-S.; writing—review and editing, M.D.-G., P.O., J.-R.R.-S. and J.G.-J.; visualization, M.D.-G.; supervision, J.-R.R.-S. and J.G.-J.; project administration, J.-R.R.-S. and J.G.-J.; funding acquisition, J.-R.R.-S. and J.G.-J. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the projects MINDMAPS (PID2023-148191NB-I00) and Voxeland (PPRO-B1-2023-017, JA.B1-09), funded by the Ministry of Science, Innovation, and Universities of Spain and by the University of Málaga, respectively, as well as by the Andalusian Plan for Research, Development, and Innovation (PPRO-TEP960-G-2023).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Tagging Prompts

Appendix A.1. Direct Tagging *with LVLM*

The following prompt is provided to the LVLM along with the input image to obtain the tag list directly.

```
I'm going to send you an image.  Your task is to tell me only the objects that
exist in the image.  Your response must be in JSON format (I'll show you an
example below), and specifically, you must tell me the keywords (categories)
of the objects in the image.

The format of your response must strictly be JSON. It is very important that
you respect the format I'm about to explain, because I will use your response
directly in an automated pipeline that requires this exact format.  The JSON
format with the object category detections must look like this (a list with
an undetermined number of elements):

["table", "dog", "floor", "shoe"]

You must always follow this list format with the words of the detected objects.
From now on, we will refer to these detected categories or objects as "labels".

Below, I'll first explain 7 strict and clear rules you must always follow when
producing your response, and then I'll provide a practical example so you un-
derstand exactly how to perform the task.

The 7 strict rules to determine which object labels to include from an
input image:

1. Do not write entire rooms.  For example, do not use "kitchen", "room", or
"space" as labels, since these refer to whole areas, not objects.

2. If there are floors or walls, just say "floor" or "wall".  For instance,
if there are tiles on the floor, do not say "tile", "tiles", or "tiled floor"—
just say "floor".  If the wall is made of bricks, say "wall", not
"brick wall".

3. Labels must be singular.  Do not say "pair of shoes", say "shoe".  Do not
say "group of people", say "person".

4. Avoid including colors or highly specific characteristics.  Do not say "red
sofa", say "sofa".  Do not say "wooden desk", say "desk".  Do not say "acous-
tic guitar", say "guitar".  Do not say "sport bag", say "bag".

5. Be specific enough.  For example, if there is clearly an apple, say "ap-
ple", not "fruit".  If there is a violin, say "violin", not "instrument".  If
there is a cat, say "cat", not "animal".

6. Avoid generic terms like "item" or "object".  If an object is not clearly
identified, do not include it in your response.
```

```
7. If the description indicates that no objects are clearly detected, return
an empty JSON list.  That is completely acceptable.

These are the 7 rules you must always follow for object labeling in JSON based
on an input image.

Now I'll provide two real-world examples:

EXAMPLE 1

Imagine I send you an image that could be described like this:

"The image shows an indoor scene featuring a small, low wooden table or shelf
against a wall with beige tiles.  On top of the table, there are several items
including a red envelope, some papers, and possibly books or magazines.  The
bottom shelf of the table appears to contain more items, but they're not
clearly visible from this angle.

In front of the table, on the floor, there is a pair of shoes with black soles
and white laces, indicating that someone might have recently removed them.
The room has curtains that are partially open, allowing light in from outside.
There doesn't seem to be any text or distinctive branding visible in this im-
age.  The style of the photograph is casual, capturing an everyday setting
without any particular artistic intent."

Your response should be (following the format and rules above):

["table", "shelf", "wall", "envelope", "papers", "book", "magazine", "floor",
"shoe", "curtain"]

EXAMPLE 2

Now imagine I send you an image that could be described like this:

"The image shows an indoor scene with a focus on two main objects:  a guitar
and a bed.  The guitar is positioned to the left of the frame, leaning against
what appears to be a wall or a stand.  It's a dreadnought-style acoustic gui-
tar with visible wood tones typical of such instruments.

To the right, there is a single bed with a dark-colored mattress and bedding.
The bed appears unoccupied, and it takes up most of the right side of the im-
age.  There are no people in the picture, giving the impression of an empty
or quiet room.

In the foreground, there's a small pile of clothes, possibly discarded, which
adds a lived-in feel to the space.  The floor includes various shades of brown
and beige.  There are no texts visible in this image, and the style appears
to be a candid or unposed photograph, likely taken without much artistic in-
tent, perhaps as a snapshot of daily life."

Your response should be:

["guitar", "bed", "wall", "stand", "mattress", "pile of clothes", "floor"]

The task explanation is now complete.  From this point on, following all the
instructions above, you must produce the response for the image I will attach.
```

Along with this prompt, the input image would be provided so that the model can produce the corresponding JSON label list.

Appendix A.2. Tagging *via LVLM Captioning + LLM Keyword Extraction*

The following prompt is provided to the LLM in the keyword extraction sub-stage, together with the image caption previously generated by the LVLM.

```
I am going to send you a textual description of an image.  Your task is to tell
me only the objects that exist in the image according to the textual descrip-
tion.  Your answer must be in JSON format (I will show you an example later),
```

and specifically, you will give me the keywords (categories) of the objects in the image.

The format of your response must strictly be in JSON. It is very important that you respect the format I will explain to you, since I will use your response directly in an automated pipeline that requires that format. The JSON format for the detected object categories you analyze should be as follows (a list with an undetermined number of elements):

["table", "dog", "floor", "shoe"]

You must always follow this list format with the words for the detected objects. From now on, we will refer to these detected categories or objects that you will write as "labels."

Next, I will first explain very strict and clear rules that you must always follow for your response, and then I will provide a practical example so you can see how to perform your task.

The 8 strict rules to determine which object labels you must include based on the textual description of the image are the following:

1. Do not write entire rooms. You must not write, for example, "kitchen," "room," or "space" as labels, since these are not objects but entire spaces.

2. If the description mentions floors or walls, simply say "floor" or "wall." For example, if there are tiles on the floor, do not say "tile," "tiles," or "tiled floor"; just say "floor." If there are bricks on the wall, do not say "brick wall"; just say "wall."

3. Labels must be in singular form. Do not say "pair of shoes"; say "shoe." Do not say "group of people"; say "person."

4. Avoid including colors or very specific characteristics in the labels. Do not say "red sofa"; say "sofa." Do not say "wooden desk"; say "desk." Do not say "acoustic guitar"; say "guitar." Do not say "sport bag"; say "bag."

5. You must be specific enough: for example, if there is clearly an apple, say "apple," not "fruit." If there is a violin, say "violin," not "instrument." If there is a cat, say "cat," not "animal."

6. Avoid generic terms in the labels, such as "item" or "object." If there is not a clearly identified object, do not include it in your response.

7. If the image description shows uncertainty about a specific object (for example, if it says there is "what seems to be a bag or a jacket" on top of a bed), you must include all possible mentioned objects ("bag," "jacket").

8. If the description of the image indicates that no clear object has been detected, return an empty JSON list—that is completely fine.

These have been the 8 rules you must always consider for object labeling in JSON based on a textual description. Now, I will give you two examples of what you should do in real cases:

EXAMPLE 1

Imagine I send you the following description:

"The image shows an indoor scene featuring a small, low wooden table or shelf against a wall with beige tiles. On top of the table, there are several items including a red envelope, some papers, and possibly books or magazines. The bottom shelf of the table appears to contain more items, but they're not clearly visible from this angle.

In front of the table, on the floor, there is a pair of shoes with black soles and white laces, indicating that someone might have recently removed them. The room has curtains that are partially open, allowing light in from outside. There doesn't seem to be any text or distinctive branding visible in this im-

```
age. The style of the photograph is casual, capturing an everyday setting
without any particular artistic intent."

Your response should be like this, following the format and rules explained:

["table", "shelf", "wall", "envelope", "papers", "book", "magazine", "floor",
"shoe", "curtain"]

EXAMPLE 2

Imagine I send you the following description:

"The image shows an indoor scene with a focus on two main objects: a guitar
and a bed. The guitar is positioned to the left of the frame, leaning against
what appears to be a wall or a stand. It's a dreadnought-style acoustic gui-
tar with visible wood tones typical of such instruments.

To the right, there is a single bed with a dark-colored mattress and bedding.
The bed appears unoccupied, and it takes up most of the right side of the im-
age. There are no people in the picture, giving the impression of an empty
or quiet room.

In the foreground, there's a small pile of clothes, possibly discarded, which
adds a lived-in feel to the space. The floor includes various shades of brown
and beige. There are no texts visible in this image, and the style appears
to be a candid or unposed photograph, likely taken without much artistic in-
tent, perhaps as a snapshot of daily life."

Your response should be as follows, strictly following the format and rules above:

["guitar", "bed", "wall", "stand", "mattress", "pile of clothes", "floor"]

The explanation of your task has ended. Now, following all the instructions
above, you must produce the response for the following input text:
```

After this prompt, the actual image description would be provided as input so that the model can produce the corresponding JSON label list.

Appendix B. Integrated Models per TALOS Stage

Table A1. Models integrated in each TALOS stage.

Stage	Substage	Model
Direct LVLM Tagging	–	Qwen2.5-VL-Instruct [60]
		Gemma 3 [61]
		MiniCPM-o 2.6 [58]
Tagging via LVLM + LLM	LVLM image descriptor	Qwen2.5-VL-Instruct [60]
		LLaVA [62]
		MiniCPM-o 2.6 [58]
	LLM keyword extractor	DeepSeek-R1 [63]
		Qwen3 [64]
		MiniCPM-o 2.6 [58]
		Llama-3.1-Nemotron-Nano [65]
Direct Tagging	–	Recognize Anything Plus Model (RAM++) [55]
Location	–	Grounding DINO [29]
Segmentation	–	Segment Anything Model 2 (SAM2) [28]

Appendix C. Quantitative Ablation of Tagging Configurations

This appendix reports the controlled component ablation used to select the Tagging realization employed in the Voxeland experiments. The 16 initial configurations comprise the three Direct LVLM Tagging models, the Cartesian product of three LVLM image descriptors and four LLM keyword extractors (12 two-stage configurations), and RAM++ as the Direct Tagging configuration listed in Appendix B. Throughout the ablation, the remaining TALOS stages were held fixed: Grounding DINO Base was used for Location and SAM2 Hiera Large for Segmentation, with identical settings. Consequently, any difference within this study is attributable to the selected Tagging method/model and its propagation through the otherwise unchanged pipeline.

Appendix C.1. Evaluation Protocol

The ablation was organized into three progressively more expensive phases. The first two phases evaluated the complete TALOS pipeline on individual images from LVIS v1 [66], before any 3D-map construction. This design served two purposes. First, reconstructing every ScanNet sequence with all 16 configurations would have entailed a prohibitive computational cost. Second, evaluating TALOS separately from Voxeland avoids treating the modular perception pipeline as a black box: it measures the semantic tags produced by Tagging, the boxes produced by Location, and the masks produced by Segmentation. The third phase then evaluated one representative of each Tagging method in the complete Voxeland mapping system. All experiments in this evaluation were conducted using a single NVIDIA RTX 6000 Ada Generation GPU.

For the LVIS phases, predicted labels were matched only by exact normalized correspondence to an official LVIS category name or synonym. On each image, unique mapped labels belonging to annotated positive categories were counted as true positives, mapped labels explicitly listed by LVIS as negative categories were counted as verified false positives, and missing positive categories were counted as false negatives. Mapped but unverified labels and labels outside or ambiguous within the LVIS taxonomy were reported separately and were not automatically treated as false positives, in accordance with the federated, non-exhaustive LVIS annotations. The reported semantic-label F1 and Jaccard values are macro-averages of the per-image category-set scores. Bounding-box AP and mask AP were computed with the official LVIS API, averaging AP over IoU thresholds from 0.50 to 0.95; AP50, AP75, size/frequency-stratified AP, and average recall were also retained as supplementary diagnostics. Per-stage and end-to-end times were measured after CUDA synchronization and exclude model loading, image-file decoding, and metric export.

Seed 2026 was used to select and order the LVIS images, and each stage made a single inference attempt per image. To make Ollama outputs reproducible, it used temperature $= 0$ and an image-specific seed computed as $(2026 \times 1{,}000{,}003 + \text{LVIS image ID}) \bmod 2{,}147{,}483{,}647$; therefore, its value differed between images but remained reproducible. Transformer-based models used greedy decoding. An invalid format or ordinary inference exception was recorded as an empty prediction: precision, recall, F1, and Jaccard were set to zero, all positive categories became false negatives, no boxes or masks were emitted, and processing continued according to the phase-specific stopping rule. This single-attempt treatment is appropriate under deterministic decoding because repeating an unchanged image, prompt, model, and seed would reproduce the same invalid response.

Appendix C.2. Phase 1: Feasibility Screening on 50 LVIS Images

The same 50 images, randomly selected without replacement from the LVIS validation set, were presented in the same order to all 16 configurations. The sample was fixed in an immutable manifest using seed 2026 and contains 750 annotated instances from

127 categories. This phase was intended only to reject configurations that were clearly unsuitable for repeated use with Voxeland, rather than to rank their final detection quality.

After each image, two independent counters were updated. A successful complete TALOS execution taking at least 60 s incremented the slow-image counter; an invalid response or ordinary inference exception incremented the scored-failure counter. A configuration stopped immediately after the third event of either type, while the triggering image and all preceding measurements were retained. One slow image plus two scored failures, for example, did not cause termination. Configurations that completed all 50 images before either counter reached three advanced only if their macro per-image semantic-label F1 was strictly greater than 0.25. All 16 reported pilots were rerun from the beginning under this common, fixed single-attempt protocol.

Appendix C.2.1. Phase 2: Quantitative Evaluation on 2500 Held-Out LVIS Images

Every Phase 1 survivor was then evaluated on the same 2500 unique LVIS images. This held-out manifest contains 26,680 annotated instances from 667 categories and has zero image overlap with Phase 1. Unlike Phase 1, no early-stopping screen was applied: all 2500 image records were retained, including inference/format failures as zero-scored semantic predictions with no detections. Per-image macro averages excluded only empty-denominator cases for which a metric was mathematically undefined, while the official AP evaluation used the complete manifest.

The purpose of this phase was first to choose one representative configuration for each of the three Tagging methods, so that the methods themselves could subsequently be compared within Voxeland. For each method, the following decision procedure was applied:

1. If no configuration of that method had survived Phase 1, a new compatible state-of-the-art model (or LVLM–LLM pair) would be implemented and evaluated again from Phase 1 until at least one candidate reached Phase 2.
2. Candidates with semantic-label F1 ≤ 0.25 or an inference/format-failure rate $\geq 5\%$ were rejected. If this left no candidate, the model-search procedure in the previous step would be repeated.
3. Among the remaining candidates, configurations dominated by another candidate of the same Tagging method were removed. Configuration a dominates b when a has a lower mean end-to-end time and is at least as good in all four selected quality measures—F1, Jaccard, bounding-box AP, and mask AP—with at least one strict quality improvement.
4. The non-dominated configurations were considered in increasing order of mean end-to-end time. A slower candidate s replaced the current faster candidate f only if its relative improvement was at least proportional to its relative time increase for every quality metric M_k:

$$\frac{M_{k,s}}{M_{k,f}} \geq \frac{T_s}{T_f} \qquad \forall M_k \in \{\text{F1}, \text{Jaccard}, \text{BBox AP}, \text{Mask AP}\}. \tag{A1}$$

Equation (A1) is an explicit engineering selection rule, rather than a hypothesis test: it requires that accepting a slower configuration does not reduce any measured quality-per-second ratio. It avoids a post hoc weighted sum of heterogeneous metrics and reflects the importance of throughput when TALOS must process long RGB-D sequences.

Appendix C.2.2. Phase 3: Voxeland Evaluation

The three representatives selected in Phase 2—one per Tagging method—were each used once to reconstruct the same 11 ScanNet v2 scenes under the conditions and 3D evaluation protocol described in Section 4. The evaluation procedure, metrics, and scene

subset were originally selected for this ablation and were subsequently reused in the comparison with Detectron2 and YOLOE. The representative with the greatest global mAP@0.5 was selected as the TALOS configuration used in the quantitative and qualitative comparison described in Section 4. In Phase 3, a single run was performed for each of the three configurations representing the different Tagging methods. The two methods based on generative models used identical temperature, top-p, and top-k settings. Unlike the 2D phases, failed generative inferences were retried on the same image without a fixed seed.

Appendix C.3. Ablation Results

Appendix C.3.1. Phase 1 Results

Tables A2–A4 report the feasibility screen separately for the three Tagging methods. For compactness in some of the result tables, MiniCPM-o and Llama denote MiniCPM-o 2.6 and Llama-3.1-Nemotron-Nano, respectively. For early-terminated configurations, F1 and timing are descriptive statistics over the inclusive terminal prefix shown in the "Images" column and are not equal-sample quality comparisons.

Table A2. Phase 1 results for Direct LVLM Tagging. "Slow" counts complete TALOS executions taking at least 60 s; failures are zero-scored inference or format errors.

Configuration	Images	Failures	Slow	F1	Mean Time (s)	Max Time (s)	Decision
Qwen2.5-VL-Instruct	5/50	0	3	0.5467	62.568	87.744	Rejected: three slow images
Gemma 3	3/50	0	3	0.4444	129.775	173.890	Rejected: three slow images
MiniCPM-o 2.6	50/50	1	0	0.3332	1.140	2.677	**Advanced**

Bold text identifies configurations advancing to Phase 2.

Table A3. Phase 1 results for Tagging via LVLM Captioning + LLM Keyword Extraction. Arrow notation denotes LVLM image descriptor → LLM keyword extractor.

Configuration	Images	Failures	Slow	F1	Mean Time (s)	Max Time (s)	Decision
Qwen2.5-VL → DeepSeek-R1	3/50	0	3	0.6508	819.766	1096.637	Rejected: three slow images
Qwen2.5-VL → Qwen3	3/50	0	3	0.6508	2552.942	2788.814	Rejected: three slow images
Qwen2.5-VL → MiniCPM-o	3/50	0	3	0.3175	721.449	976.148	Rejected: three slow images
Qwen2.5-VL → Llama	3/50	0	3	0.6508	4067.617	5655.379	Rejected: three slow images
LLaVA → DeepSeek-R1	50/50	2	1	0.5042	29.368	623.010	**Advanced**
LLaVA → Qwen3	9/50	0	3	0.4651	56.371	91.355	Rejected: three slow images
LLaVA → MiniCPM-o	50/50	0	0	0.2938	5.213	7.981	**Advanced**
LLaVA → Llama	6/50	0	3	0.4167	152.802	481.379	Rejected: three slow images
MiniCPM-o → DeepSeek-R1	50/50	0	0	0.5364	13.601	35.053	**Advanced**
MiniCPM-o → Qwen3	9/50	0	3	0.3704	53.145	87.098	Rejected: three slow images
MiniCPM-o → MiniCPM-o	50/50	2	0	0.3226	2.410	3.372	**Advanced**
MiniCPM-o → Llama	4/50	0	3	0.4048	86.077	133.531	Rejected: three slow images

Bold text identifies configurations advancing to Phase 2.

Table A4. Phase 1 result for Direct Tagging with a discriminative open-vocabulary model.

Configuration	Images	Failures	Slow	F1	Mean Time (s)	Max Time (s)	Decision
RAM++	50/50	0	0	0.4836	0.709	2.858	**Advanced**

Bold text identifies configurations advancing to Phase 2.

The six configurations marked as advanced completed all 50 images, did not reach either three-event stopping condition, and exceeded the strict F1 threshold of 0.25. All other configurations were excluded because their third slow image occurred after only 3–9 pilot images; none was rejected on the basis of a selectively truncated F1 estimate.

Appendix C.3.2. Phase 2 Results

Tables A5–A7 present the held-out 2500-image results used for within-method selection. The failure rate is the percentage of images assigned an empty, zero-quality prediction; those images remain included in all reported means and AP calculations.

Table A5. Phase 2 results for the surviving Direct LVLM Tagging configuration.

Configuration	Failure (%)	F1	Jaccard	BBox AP	Mask AP	Mean Time (s)	Decision
MiniCPM-o 2.6	0.36	0.3708	0.3083	0.0670	0.0608	1.070	**Method representative**

Bold text identifies the selected representative of each Tagging method, which therefore advances to Phase 3.

Table A6. Phase 2 results for Tagging via LVLM Captioning + LLM Keyword Extraction. Arrow notation denotes LVLM image descriptor → LLM keyword extractor.

Configuration	Failure (%)	F1	Jaccard	BBox AP	Mask AP	Mean Time (s)	Decision
MiniCPM-o → MiniCPM-o	1.88	0.3481	0.2872	0.0726	0.0684	2.465	**Method representative**
MiniCPM-o → DeepSeek-R1	0.32	0.5211	0.4413	0.1154	0.1066	14.927	Rejected by proportional-cost rule
LLaVA → DeepSeek-R1	0.36	0.4611	0.3847	0.0905	0.0834	19.835	Pareto dominated
LLaVA → MiniCPM-o	1.84	0.3084	0.2544	0.0572	0.0523	5.115	Pareto dominated

Bold text identifies the selected representative of each Tagging method, which therefore advances to Phase 3.

Table A7. Phase 2 result for Direct Tagging with a discriminative open-vocabulary model.

Configuration	Failure (%)	F1	Jaccard	BBox AP	Mask AP	Mean Time (s)	Decision
RAM++	0.00	0.5072	0.4296	0.0714	0.0657	0.667	**Method representative**

Bold text identifies the selected representative of each Tagging method, which therefore advances to Phase 3.

All six configurations remained above the F1 floor and below the 5% failure-rate ceiling. MiniCPM-o 2.6 and RAM++ were therefore selected directly as the sole survivors of their respective methods. Within the two-stage method, MiniCPM-o → MiniCPM-o dominated LLaVA → MiniCPM-o, while MiniCPM-o → DeepSeek-R1 dominated LLaVA → DeepSeek-R1. The remaining comparison illustrates the efficiency rule: relative to MiniCPM-o → MiniCPM-o, MiniCPM-o → DeepSeek-R1 required 6.06× the mean end-to-end time, whereas its gains were only 1.50× in F1, 1.54× in Jaccard, 1.59× in bounding-box AP, and 1.56× in mask AP. It therefore did not satisfy Equation (A1), and MiniCPM-o → MiniCPM-o was selected as the two-stage representative.

Appendix C.3.3. Phase 3 Results

Table A8 reports the global 3D map results for the three method representatives. The accounting follows Table 1: OOV/Ign. denotes predictions outside the global 119-class evaluation vocabulary and excluded from AP.

Table A8. Phase 3 results for the three Tagging method representatives on 11 ScanNet v2 scenes. MiniCPM-o 2.6 represents Direct LVLM Tagging; MiniCPM-o → MiniCPM-o represents Tagging via LVLM Captioning + LLM Keyword Extraction, using MiniCPM-o 2.6 in both substages; and RAM++ represents Direct Tagging with a discriminative open-vocabulary model. OOV/Ign. denotes predictions outside the global 119-class evaluation vocabulary and excluded from AP. The best mAP@0.5 is highlighted in bold.

Configuration	mAP@0.5	GT Inst.	Pred Inst.	TP	FP	OOV/Ign.	FN
MiniCPM-o 2.6	**0.1799**	530	1926	110	1423	393	420
MiniCPM-o → MiniCPM-o	0.1525	530	3876	133	2581	1162	397
RAM++	0.0685	530	3130	94	1746	1290	436

Among the three Phase 3 representatives, Direct LVLM Tagging with MiniCPM-o 2.6 achieved the highest global mAP@0.5: 0.1799, compared with 0.1525 for MiniCPM-o → MiniCPM-o and 0.0685 for RAM++. The two-stage representative was thus the closest alternative to the final winner; however, in addition to its lower 3D mAP, Phase 2 showed that its mean end-to-end time was 2.30 times that of Direct LVLM Tagging (2.465 versus 1.070 s per image), consistent with the cost of its two successive generative inference stages. Although the two-stage configuration produced more true positives than Direct LVLM Tagging (133 versus 110), it also produced substantially more evaluable false positives (2581 versus 1423), resulting in a lower mAP@0.5. Direct LVLM Tagging with MiniCPM-o 2.6 was therefore selected for the quantitative and qualitative experiments in Section 4. After this selection, the Direct LVLM configuration was run four additional times, for a total of five runs; its Phase 3 value of 0.1799 corresponds to the first run, while Table 1 reports the resulting mean of 0.1607 ± 0.0255 across all five runs.

References

1. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
2. Kostavelis, I.; Gasteratos, A. Semantic Mapping for Mobile Robotics Tasks: A Survey. *Robot. Auton. Syst.* **2015**, *66*, 86–103. [CrossRef]
3. Moncada-Ramirez, J.; Matez-Bandera, J.-L.; Gonzalez-Jimenez, J.; Ruiz-Sarmiento, J.-R. Agentic Workflows for Improving Large Language Model Reasoning in Robotic Object-Centered Planning. *Robotics* **2025**, *14*, 24. [CrossRef]
4. Liu, K.; Tang, Z.; Wang, D.; Wang, Z.; Li, X.; Zhao, B. COHERENT: Collaboration of Heterogeneous Multi-Robot System with Large Language Models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Atlanta, GA, USA, 19–23 May 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 10208–10214.
5. Zhou, E.; An, J.; Chi, C.; Han, Y.; Rong, S.; Zhang, C.; Wang, P.; Wang, Z.; Huang, T.; Sheng, L.; et al. RoboRefer: Towards Spatial Referring with Reasoning in Vision-Language Models for Robotics. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*; Neural Information Processing Systems Foundation, Inc.: San Diego, CA, USA, 2025; pp. 32209–32286.
6. Ahn, M.; Brohan, A.; Brown, N.; Chebotar, Y.; Cortes, O.; David, B.; Finn, C.; Fu, C.; Gopalakrishnan, K.; Hausman, K.; et al. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. *arXiv* **2022**, arXiv:2204.01691.
7. Peng, S.; Genova, K.; Jiang, C.; Tagliasacchi, A.; Pollefeys, M.; Funkhouser, T. OpenScene: 3D Scene Understanding with Open Vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 815–824.
8. Gu, Q.; Kuwajerwala, A.; Morin, S.; Jatavallabhula, K.M.; Sen, B.; Agarwal, A.; Rivera, C.; Paul, W.; Ellis, K.; Chellappa, R.; et al. ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 13–17 May 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 5021–5028.
9. Ruiz-Sarmiento, J.-R.; Galindo, C.; Gonzalez-Jimenez, J. Building Multiversal Semantic Maps for Mobile Robot Operation. *Knowl.-Based Syst.* **2017**, *119*, 257–272. [CrossRef]
10. McCormac, J.; Handa, A.; Davison, A.; Leutenegger, S. SemanticFusion: Dense 3D Semantic Mapping with Convolutional Neural Networks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 4628–4635.
11. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L. Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020*; IEEE: Piscataway, NJ, USA, 2020; pp. 1689–1696.
12. Narita, G.; Seno, T.; Ishikawa, T.; Kaji, Y. PanopticFusion: Online Volumetric Semantic Mapping at the Level of Stuff and Things. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 4205–4212.
13. Matez-Bandera, J.-L.; Ojeda, P.; Monroy, J.; Gonzalez-Jimenez, J.; Ruiz-Sarmiento, J.-R. Voxeland: Probabilistic Instance-Aware Semantic Mapping with Evidence-based Uncertainty Quantification. *arXiv* **2024**, arXiv:2411.08727.
14. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 2961–2969.
15. Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *Computer Vision—ECCV 2018*; Springer: Cham, Switzerland, 2018; pp. 801–818.

16. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 2881–2890.
17. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Springer: Cham, Switzerland, 2014; pp. 740–755.
18. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
19. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016*; IEEE: Piscataway, NJ, USA, 2016; pp. 3213–3223.
20. Silberman, N.; Hoiem, D.; Kohli, P.; Fergus, R. Indoor Segmentation and Support Inference from RGBD Images. In *Computer Vision—ECCV 2012*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 746–760.
21. Liu, C.; Yuen, J.; Torralba, A. SIFT Flow: Dense Correspondence across Scenes and Its Applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 978–994. [CrossRef] [PubMed]
22. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning Transferable Visual Models from Natural Language Supervision. In *Proceedings of the International Conference on Machine Learning (ICML), Virtual Event, 18–24 July 2021*; PMLR: Cambridge, MA, USA, 2021; Volume 139, pp. 8748–8763.
23. Minderer, M.; Gritsenko, A.; Stone, A.; Neumann, M.; Weissenborn, D.; Dosovitskiy, A.; Mahendran, A.; Arnab, A.; Dehghani, M.; Shen, Z.; et al. Simple Open-Vocabulary Object Detection with Vision Transformers. In *Computer Vision—ECCV 2022*; Springer: Cham, Switzerland, 2022.
24. Gu, X.; Lin, T.-Y.; Kuo, W.; Cui, Y. Open-Vocabulary Object Detection via Vision and Language Knowledge Distillation. In *Proceedings of the International Conference on Learning Representations (ICLR), Virtual Event, 25–29 April 2022*; OpenReview: Amherst, MA, USA, 2022.
25. Ghiasi, G.; Gu, X.; Cui, Y.; Lin, T.-Y. Scaling Open-Vocabulary Image Segmentation with Image-Level Labels. In *Computer Vision—ECCV 2022*; Springer: Cham, Switzerland, 2022.
26. Jatavallabhula, K.M.; Kuwajerwala, A.; Gu, Q.; Omama, M.; Chen, T.; Maalouf, A.; Li, S.; Iyer, G.; Saryazdi, S.; Keetha, N.; et al. ConceptFusion: Open-Set Multimodal 3D Mapping. *arXiv* **2023**, arXiv:2302.07241.
27. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.-Y.; et al. Segment Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 4015–4026.
28. Ravi, N.; Gabeur, V.; Hu, Y.-T.; Hu, R.; Ryali, C.; Ma, T.; Khedr, H.; Rädle, R.; Rolland, C.; Gustafson, L.; et al. SAM 2: Segment Anything in Images and Videos. *arXiv* **2024**, arXiv:2408.00714.
29. Liu, S.; Zeng, Z.; Ren, T.; Li, F.; Zhang, H.; Yang, J.; Jiang, Q.; Li, C.; Yang, J.; Su, H.; et al. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. In *Computer Vision—ECCV 2024*; Springer: Cham, Switzerland, 2024; pp. 38–55.
30. Decena-Giménez, M.; Moncada-Ramírez, J.; Ruiz-Sarmiento, J.R.; González-Jiménez, A.J. Segmentación Semántica de Instancias de Objetos Empleando un Vocabulario Abierto. In *Simposio CEA de Robótica, Bioingeniería, Visión Artificial y Automática Marina*; Comité Español de Automática (CEA): Barcelona, Spain, 2025; Volume 1. [CrossRef]
31. Wang, A.; Liu, L.; Chen, H.; Lin, Z.; Han, J.; Ding, G. YOLOE: Real-Time Seeing Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Honolulu, HI, USA, 19–23 October 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 24591–24602.
32. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 5828–5839.
33. Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.-Y.; Girshick, R. Detectron2. GitHub. 2019. Available online: https://github.com/facebookresearch/detectron2 (accessed on 10 September 2026).
34. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid Task Cascade for Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 4974–4983.
35. Cheng, B.; Misra, I.; Schwing, A.G.; Kirillov, A.; Girdhar, R. Masked-Attention Mask Transformer for Universal Image Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022*; IEEE: Piscataway, NJ, USA, 2022; pp. 1290–1299.
36. Zou, X.; Dou, Z.-Y.; Yang, J.; Gan, Z.; Li, L.; Li, C.; Dai, X.; Behl, H.S.; Wang, J.; Yuan, L.; et al. Generalized Decoding for Pixel, Image, and Language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 15116–15127.

37. Ren, T.; Liu, S.; Zeng, A.; Lin, J.; Li, K.; Cao, H.; Chen, J.; Huang, X.; Chen, Y.; Yan, F.; et al. Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks. *arXiv* **2024**, arXiv:2401.14159.
38. Carion, N.; Gustafson, L.; Hu, Y.-T.; Debnath, S.; Hu, R.; Surís, D.; Ryali, C.K.; Alwala, K.V.; Khedr, H.; Huang, A.; et al. SAM 3: Segment Anything with Concepts. *arXiv* **2025**, arXiv:2511.16719.
39. Kawano, Y.; Aoki, Y. TAG: Guidance-Free Open-Vocabulary Semantic Segmentation. *arXiv* **2024**, arXiv:2403.11197.
40. Chen, J.; Zhu, D.; Qian, G.; Ghanem, B. Exploring Open-Vocabulary Semantic Segmentation from CLIP Vision Encoder Distillation Only. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 699–710.
41. Zhang, H.; Li, F.; Zou, X.; Liu, S.; Li, C.; Yang, J.; Zhang, L. A Simple Framework for Open-Vocabulary Segmentation and Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 1020–1031.
42. Yu, Q.; He, J.; Deng, X.; Shen, X.; Chen, L.-C. Convolutions Die Hard: Open-Vocabulary Segmentation with Single Frozen Convolutional CLIP. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; pp. 32215–32234.
43. Hong, Y.; Zhen, H.; Chen, P.; Zheng, S.; Du, Y.; Chen, Z.; Gan, C. 3D-LLM: Injecting the 3D World into Large Language Models. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; pp. 20482–20494.
44. Chen, S.; Chen, X.; Zhang, C.; Li, M.; Yu, G.; Fei, H.; Zhu, H.; Fan, J.; Chen, T. LL3DA: Visual Interactive Instruction Tuning for Omni-3D Understanding, Reasoning, and Planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 26428–26438.
45. Rana, K.; Haviland, J.; Garg, S.; Abou-Chakra, J.; Reid, I.; Suenderhauf, N. SayPlan: Grounding Large Language Models using 3D Scene Graphs for Scalable Robot Task Planning. In *Proceedings of the 7th Conference on Robot Learning (CoRL), Atlanta, GA, USA, 6–9 November 2023*; PMLR: Cambridge, MA, USA, 2023; Volume 229, pp. 23–72.
46. Huang, W.; Wang, C.; Zhang, R.; Li, Y.; Wu, J.; Fei-Fei, L. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models. In *Proceedings of the 7th Conference on Robot Learning (CoRL), Atlanta, GA, USA, 6–9 November 2023*; PMLR: Cambridge, MA, USA, 2023; Volume 229, pp. 540–562.
47. Yin, H.; Xu, X.; Wu, Z.; Zhou, J.; Lu, J. SG-Nav: Online 3D Scene Graph Prompting for LLM-based Zero-shot Object Navigation. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*; Curran Associates, Inc.: Red Hook, NY, USA, 2024; pp. 5285–5307.
48. Grinvald, M.; Furrer, F.; Novkovic, T.; Chung, J.J.; Cadena, C.; Siegwart, R.; Nieto, J. Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3037–3044. [CrossRef]
49. Wang, L.; Li, R.; Sun, J.; Liu, X.; Zhao, L.; Seah, H.S.; Quah, C.K.; Tandianus, B. Multi-View Fusion-Based 3D Object Detection for Robot Indoor Scene Perception. *Sensors* **2019**, *19*, 4092. [CrossRef] [PubMed]
50. Li, W.; Gu, J.; Chen, B.; Han, J. Incremental Instance-Oriented 3D Semantic Mapping via RGB-D Cameras for Unknown Indoor Scene. *Discret. Dyn. Nat. Soc.* **2020**, *2020*, 2528954. [CrossRef]
51. Mascaro, R.; Teixeira, L.; Chli, M. Volumetric Instance-Level Semantic Mapping via Multi-View 2D-to-3D Label Diffusion. *IEEE Robot. Autom. Lett.* **2022**, *7*, 3531–3538. [CrossRef]
52. Miao, Y.; Armeni, I.; Pollefeys, M.; Barath, D. Volumetric Semantically Consistent 3D Panoptic Mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Abu Dhabi, United Arab Emirates, 14–18 October 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 12924–12931.
53. Feng, X.; Li, Y.; Chen, D.; Gao, M.; Liu, M.; Yuan, J.; Qiao, C. Benchmarking Large and Small MLLMs. *Mach. Vis. Appl.* **2025**, *36*, 137. [CrossRef]
54. Ding, S.; Wu, S.; Zhao, X.; Zang, Y.; Duan, H.; Dong, X.; Zhang, P.; Cao, Y.; Lin, D.; Wang, J. MM-IFEngine: Towards Multimodal Instruction Following. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Honolulu, HI, USA, 19–23 October 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1099–1109.
55. Huang, X.; Huang, Y.-J.; Zhang, Y.; Tian, W.; Feng, R.; Zhang, Y.; Xie, Y.; Li, Y.; Zhang, L. Open-Set Image Tagging with Multi-Grained Text Supervision. *arXiv* **2023**, arXiv:2310.15200.
56. Jøsang, A. *Subjective Logic*; Springer International Publishing: Cham, Switzerland, 2016.
57. Johnson, S.J.; Murty, M.R.; Navakanth, I. A detailed review on word embedding techniques with emphasis on word2vec. *Multimed. Tools Appl.* **2024**, *83*, 37979–38007 [CrossRef]
58. OpenBMB. MiniCPM-o 2.6. Hugging Face Model Card. 2025. Available online: https://huggingface.co/openbmb/MiniCPM-o-2_6 (accessed on 10 September 2026).
59. Ambrosio-Cestero, G.; Matez-Bandera, J.-L.; Ruiz-Sarmiento, J.-R.; Gonzalez-Jimenez, J. Linux Container-Based Environment for the Development of Robotic Applications. In *XLV Jornadas de Automática*; Servizo de Publicacións da Universidade da Coruña: A Coruña, Spain; Comité Español de Automática: Barcelona, Spain, 2024; Volume 45. [CrossRef]
60. Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. Qwen2.5-VL Technical Report. *arXiv* **2025**, arXiv:2502.13923.

61. Gemma Team; Kamath, A.; Ferret, J.; Pathak, S.; Vieillard, N.; Merhej, R.; Perrin, S.; Matejovicova, T.; Ramé, A.; Rivère, M.; et al. Gemma 3 Technical Report. *arXiv* **2025**, arXiv:2503.19786.
62. Liu, H.; Li, C.; Wu, Q.; Lee, Y.J. Visual Instruction Tuning. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; pp. 34892–34916.
63. DeepSeek-AI; Guo, D.; Yang, D.; Zhang, H.; Song, J.; Wang, P.; Zhu, Q.; Xu, R.; Zhang, R.; Ma, S.; et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv* **2025**, arXiv:2501.12948.
64. Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. Qwen3 Technical Report. *arXiv* **2025**, arXiv:2505.09388.
65. Bercovich, A.; Levy, I.; Golan, I.; Dabbah, M.; El-Yaniv, R.; Puny, O.; Galil, I.; Moshe, Z.; Ronen, T.; Nabwani, N.; et al. Llama-Nemotron: Efficient Reasoning Models. *arXiv* **2025**, arXiv:2505.00949.
66. Gupta, A.; Dollár, P.; Girshick, R. LVIS: A Dataset for Large Vocabulary Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 5356–5364.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.