Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

From Local Matches to Global Masks: Template-Guided Instance Detection and Segmentation in Open-World Scenes

Qifan Zhang, Sai Haneesh Allu, Jikai Wang, Yangxiao Lu, Yu Xiang
Intelligent Robotics and Vision Lab, The University of Texas at Dallas
{qifan.zhang, saihaneesh.allu, jikai.wang, yangxiao.lu, yu.xiang}@utdallas.edu

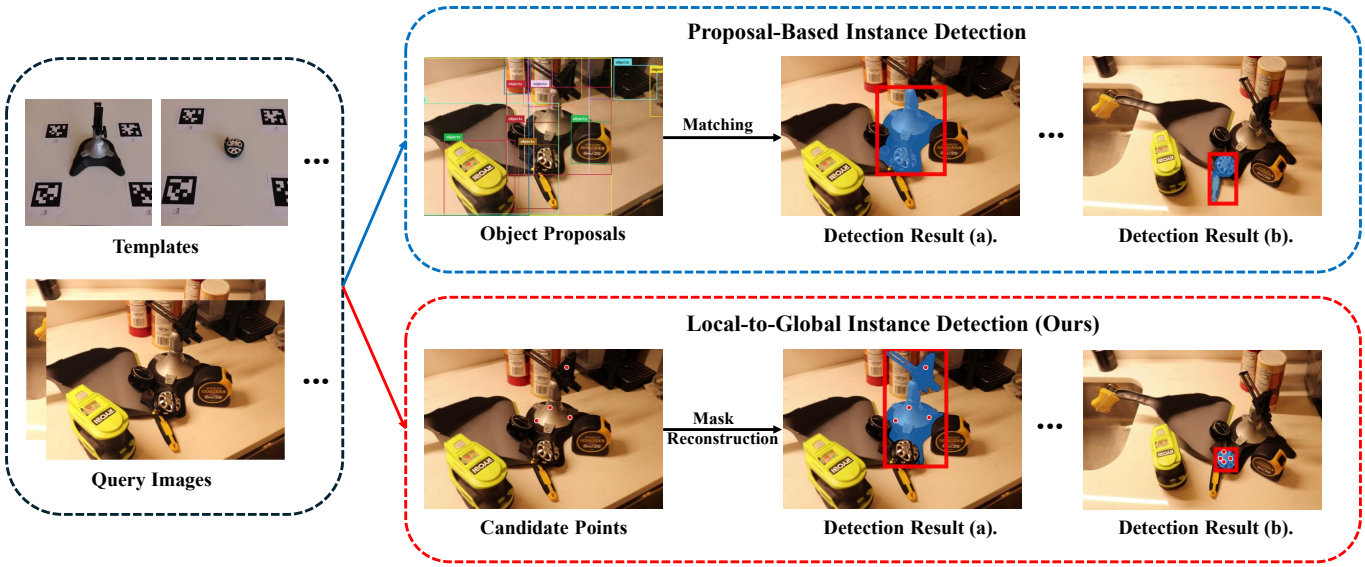


Fig. 1: Conceptual comparison between object proposal-based instance detection methods and our local-to-global instance detection framework. Top: Proposal-based approaches [28] first generate object proposals in the query image and then perform instance matching to obtain detection results. Bottom: Our method starts from dense local correspondences to identify candidate points and reconstructs complete instance masks through mask reconstruction, producing final detection results without explicit proposal generation.

***Abstract*—Detecting and segmenting novel object instances in open-world environments is a fundamental problem in robotic perception. Given only a small set of template images, a robot must locate and segment a specific object instance in a cluttered, previously unseen scene. Existing proposal-based approaches are highly sensitive to proposal quality and often fail under occlusion and background clutter. We propose L2G-Det, a local-to-global instance detection framework that bypasses explicit object proposals by leveraging dense patch-level matching between templates and the query image. Locally matched patches generate candidate points, which are refined through a candidate selection module to suppress false positives. The filtered points are then used to prompt an augmented Segment Anything Model (SAM) with instance-specific object tokens, enabling reliable reconstruction of complete instance masks. Experiments demonstrate improved performance over proposal-based methods in challenging open-world settings.** [1]

[1]Project website: https://irvlutd.github.io/L2G/

I. Introduction

Object instance detection and segmentation in open-world environments [17, 31] have become increasingly important in robotic perception. In this problem setting, a robot is given a small set of template images of a target object, often captured from multiple viewpoints and is required to locate that specific object instance in a novel, cluttered scene. The goal is not only to determine whether the object is present, but also to precisely localize it and recover an accurate instance segmentation mask.

This capability is fundamental to many real-world robotic applications. For example, a service robot may be instructed to retrieve a particular household item shown to it only once, or a warehouse robot may need to pick a newly introduced product for which no prior training data exists. As robots increasingly operate in open-world environments, the ability to find and segment novel object instances from a small number of visual templates becomes a core perception capability.

Recent approaches [17, 22, 28, 41] predominantly adopt an object proposal-based pipeline. As illustrated in Fig. 1(top), an object proposal generator such as [17, 25] is first applied to the query image to produce object-like regions, after which template embeddings are matched against these proposals to identify the target instance. However, the effectiveness of this pipeline is critically dependent on the quality of the proposal. Inaccurate proposals, such as regions covering only partial object parts or regions affected by background clutter, directly degrade the subsequent embedding matching stage and lead to degraded detection and segmentation performance. These limitations are particularly pronounced in real-world robotic environments, where objects frequently undergo occlusions, appear under diverse viewpoints, or are only partially visible. Under such conditions, proposal-based detectors struggle to produce high-quality proposals that align well with the complete object appearances provided by the templates, as illustrated in Fig. 1(top).

To address these issues, we propose a new local-to-global instance detection framework (L2G-Det) centered on dense local feature matching. As illustrated in Fig. 1(bottom), our approach first extracts dense patch-level features from template images using DINOv3 [42], where each patch represents a localized object cue. For each template patch within the object, we identify its best-matching location in the query image and treat the center of the corresponding patch as a candidate point. By aggregating candidate points across multiple template views, we obtain a rich set of object-specific local cues. Instead of relying on explicit object proposals, we leverage these locally matched candidate points to reconstruct the global target object mask.

However, dense local matching alone inevitably introduces false positives due to local appearance ambiguities, where background regions or distractor objects share similar local textures or patterns with the target instance. To suppress such erroneous matches, we introduce a candidate selection module that refines candidate points. By probing each candidate point as a prompt with SAM [19] to obtain a local mask, and then comparing masked region embeddings against template embeddings, the candidate selector effectively filters unreliable matches while preserving locally consistent object cues. We also introduce a lightweight adapter [13] to strengthen the ability of patch embeddings based on contrastive learning [44].

During mask generation, we use the filtered candidate points as prompts for SAM. However, these candidate points may not cover the entire object, which can result in incomplete masks. To address this limitation, we propose an *Augmented SAM* module that incorporates a learnable, instance-specific object token [21, 16] into the mask decoder. This object token guides the frozen decoder to complete missing object parts and recover coherent global masks. The object tokens are learned using template-based synthetic images and stored in a memory pool, enabling incremental addition of instance-specific object tokens for new object instances without affecting previously learned ones.

In general, our approach achieves the state-of-the-art performance on two challenging instance detection benchmarks [40, 22] and demonstrates strong real-world performance in robotic experiments involving object search and navigation in cluttered environments.

Our main contributions are summarized as follows:

- **Local-to-global novel instance detection.** We propose a local-to-global framework that bypasses object proposals by reconstructing global instance masks from dense local correspondences, enabling robust novel instance detection in cluttered scenes.
- **Candidate selection via dense local matching.** We introduce a candidate selector that leverages multi-view templates to filter unreliable local matches and suppress false positives caused by appearance ambiguities.
- **Template-based instance-specific object tokens.** We propose an instance-specific object token memory that supports incremental learning of novel objects without interfering with previously learned instances.

II. Related Work

Instance Detection. Instance detection [5, 29, 22, 41], aims to localize and segment specific object instances that are unseen during training, typically given only a small set of template images. Most existing approaches adopt a proposal-based [28, 40] instance matching pipeline, where object-proposals are first generated and then matched against template embeddings. In addition to generating 2D representations [2, 29, 26], VoxDet [22] further leverages multi-view observations to construct 3D representations. While effective in relatively clean settings, these methods are highly sensitive to proposal quality. In contrast, our method avoids explicit proposal generation and instead reconstructs global object masks by aggregating locally matched features.

Dense Local Feature Matching. Recent pretrained visual encoders [7, 4] produce semantically meaningful patch-level features [10] that generalize across objects and viewpoints, enabling dense matching between template and query images. Several works exploit such correspondences for localization [39, 43] or pose estimation [46]. However, they typically focus on geometric alignment [47] or sparse keypoints [27, 3] rather than instance-level segmentation. Our approach builds upon dense patch-level matching [42] but extends it to novel instance detection by aggregating locally consistent matches across multiple template views and reconstructing global object masks [19] from these local cues.

Foundation Models. Foundation models refer to general-purpose visual models pretrained on large-scale and diverse data, which provide highly transferable representations for a wide range of downstream tasks. Pretrained visual encoders [4, 7, 35] have demonstrated strong capability in extracting semantically consistent and instance-discriminative features, making them well suited for feature matching and instance-level perception. Proposal-based instance detection methods typically rely on detectors such as GroundingDINO [25], which extend foundation models to open-vocabulary object localization by generating object proposals. In contrast, our

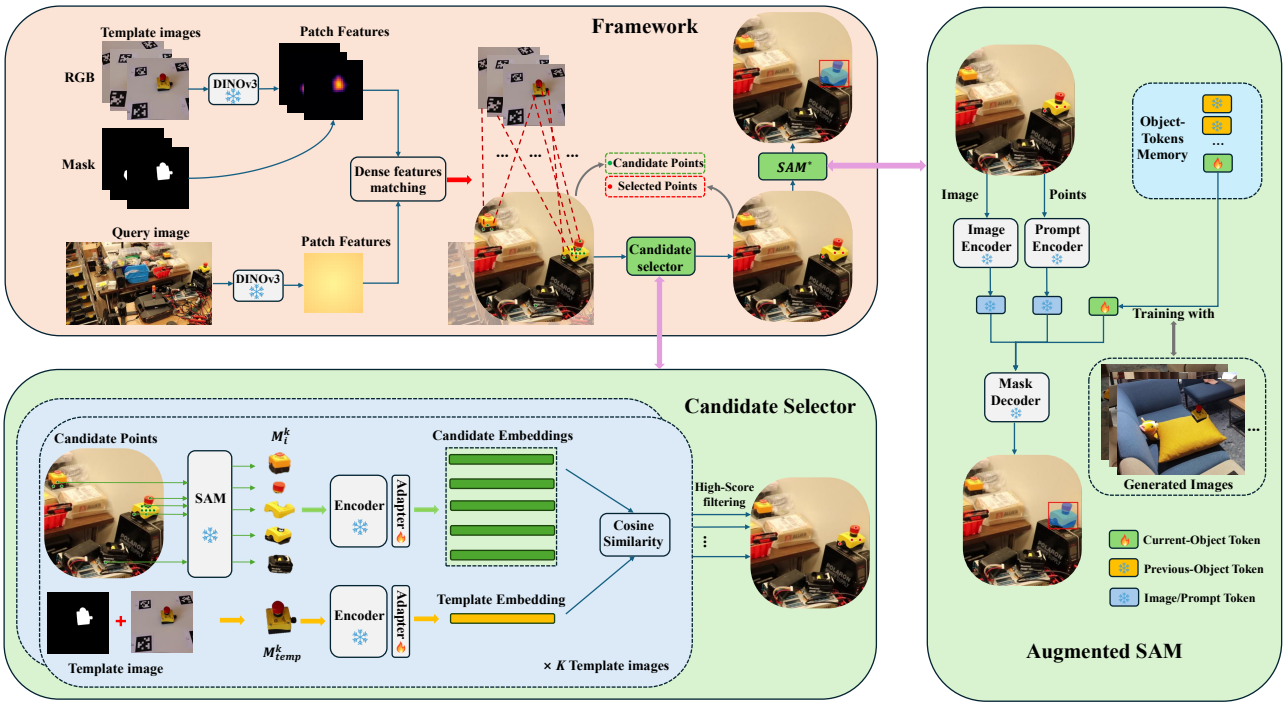


Fig. 2: Overview of our L2G-Det framework for novel instance detection. It consists of a candidate selection module and an augmented SAM module (SAM*). Only the adapters and object-tokens are learnable, while all other components are frozen.

method leverages Segment Anything (SAM) [19, 37, 6] to reconstruct instance masks directly from locally matched features obtained via dense feature matching, without relying on proposal generation. Unlike existing approaches [8, 48] that fine-tune SAM for task-specific adaptation, we target the novel instance detection setting and introduce instance-specific object tokens to guide mask reconstruction, making our approach better suited for open-world, real-world environments.

III. Method

Most prior instance detection methods rely on an explicit object proposal detector. However, in cluttered open-world scenes, the quality of the proposal can be unreliable (partial boxes, background leakage, or missed instances), directly degrading template matching and final segmentation. To avoid this dependency, we adopt a local-to-global strategy that starts from dense local correspondences rather than global proposals (Fig. 2). Specifically, we first perform dense patch-level matching between each template view and the query image to obtain candidate points that likely lie on the target instance. Since dense matching can introduce false positives due to local appearance ambiguities, we then design a Candidate Selector to filter candidate points. Finally, using the filtered points as prompts, we reconstruct the full-instance mask with an augmented SAM decoder that is guided by a learnable instance-specific object token. We describe these components in this section.

Our goal is to detect and segment a target object instance given a set of template images of that object. We assume that each object instance is associated with K template images $\{\mathcal{I}^k_{\text{temp}}\}_{k=1}^{K}$ with corresponding object masks $\{\mathcal{M}^k_{\text{temp}}\}_{k=1}^{K}$. These template images characterize the appearance of the instance from multiple viewpoints. Given a query image $\mathcal{I}_{\text{query}}$, our objective is to localize the target instance and predict its segmentation mask by using the visual correspondence between the template images and the query image.

A. Dense Feature Matching

As illustrated in the **Framework** of Fig. 2, we first perform dense feature matching between the template images and the query image to obtain an initial set of candidate points. Our model employs a frozen DINOv3 [42] backbone to extract dense patch-level features.

For a given template image $\mathcal{I}^k_{\text{temp}}$, we uniformly sample S patches inside the corresponding object mask $\mathcal{M}^k_{\text{temp}}$. Let $\mathbf{f}^k_i \in \mathbb{R}^D$ denote the feature embedding of the i-th sampled patch from the k-th template image, where $i = 1, \ldots, S$ and D is the dimension of the feature embedding. Similarly, we extract dense patch features $\{\mathbf{f}_j\}_{j=1}^{N}$ from the query image $\mathcal{I}_{\text{query}}$ using the same DINOv3 encoder, where N denotes the number of feature embeddings from the query image.

To match the feature embeddings, for each template patch feature $\mathbf{f}^k_i$, we compute its cosine similarity with all patch features $\{\mathbf{f}_j\}_{j=1}^{N}$ in the query image. Then for each template patch, we select the query patch with the highest cosine similarity score:

$$j^* = \arg\max_{j \in \{1,\ldots,N\}} \text{sim}\left(\mathbf{f}^k_i, \mathbf{f}_j\right). \tag{1}$$

The spatial center of the selected query patch j^* is then taken as a candidate point $\mathbf{p}^k_i$ corresponding to the i-th patch from the k-th template. Repeating this process for all S

sampled patches yields a set of candidate points for the k-th template:

$$\mathcal{C}^k = \left\{ \mathbf{p}_i^k \mid i = 1, \ldots, S \right\}. \tag{2}$$

Finally, we apply this procedure independently to each template image to generate K candidate point sets $\{\mathcal{C}^k\}_{k=1}^K$, which serve as the initial candidate points for subsequent refinement and selection.

B. Candidate Selector

After performing dense feature matching using patch features extracted by DINOv3, we obtain an initial set of candidate points. However, many of these candidates do not fall within the target object region and are falsely matched. This is mainly caused by local regions in the query image that exhibit similar appearance to certain local parts of the target instance. As illustrated in the **Framework** of Fig. 2, red dashed lines indicate dense feature correspondences, while the green points denote the initial candidate points, among which some lie outside the true object region. Therefore, an additional filtering step is required to refine the candidate set.

The **Candidate Selector** in Fig. 2 depicts the filtering process applied in a template-wise manner. For each template image, we perform the same candidate selection pipeline independently, and finally aggregate the selected points from all templates, corresponding to the red *Selected Points* shown in the figure.

a) Single-point SAM probing: Given a template image $\mathcal{I}_{\text{temp}}^k$, we first consider its associated candidate point set $\mathcal{C}^k = \{\mathbf{p}_i^k\}_{i=1}^S$ obtained from dense feature matching. Each candidate point $\mathbf{p}_i^k$ is individually used as a point prompt for the SAM model, which produces a corresponding mask region $\mathcal{M}_i^k, i = 1, \ldots, S$ in the query image. Since only a single point is provided as the prompt, the resulting mask typically focuses on a local region around the queried point, serving as a local feature probe.

b) Candidate and template embeddings: The generated local mask $\mathcal{M}_i^k$ is applied to the query image to extract a masked local region, which is then processed by a frozen feature encoder $E(\cdot)$ followed by a learnable adapter to obtain the candidate embedding. Similarly, using the template image together with its object mask, we extract the full object region and compute the corresponding template embedding:

$$\mathbf{z}_i^k = \mathcal{A}\big(E\big(\mathcal{I}_{\text{query}} \odot \mathcal{M}_i^k\big)\big), \tag{3}$$

$$\mathbf{z}_{\text{temp}}^k = \mathcal{A}\big(E\big(\mathcal{I}_{\text{temp}}^k \odot \mathcal{M}_{\text{temp}}^k\big)\big), \tag{4}$$

where $\odot$ denotes element-wise masking, $E(\cdot)$ is a pretrained encoder for visual feature extraction, and $\mathcal{A}(\cdot)$ is a lightweight residual MLP adapter.

c) Learnable residual MLP adapter: The adapter $\mathcal{A}(\cdot)$ is implemented as a residual MLP:

$$\mathcal{A}(\mathbf{x}) = \mathbf{x} + \alpha \cdot \mathrm{MLP}(\mathbf{x}), \tag{5}$$

where $\mathbf{x}$ denotes the input to the adapter, $\mathrm{MLP}(\cdot)$ consists of two linear layers with a non-linear activation in between, and α is a scaling factor. All parameters of the backbone encoder $E(\cdot)$ are frozen, and only the adapter parameters are learnable.

The adapter is introduced to further enhance instance-level discrimination via contrastive learning on the given template images. Given embeddings from the same object instance as positives and embeddings from different instances as negatives, we employ an InfoNCE-style contrastive loss [32, 9] to learn the adapter:

$$\mathcal{L}_{\text{con}} = -\log \frac{e^{\mathrm{sim}(\mathbf{z}, \mathbf{z}^+)/\tau}}{e^{\mathrm{sim}(\mathbf{z}, \mathbf{z}^+)/\tau} + \sum_{\mathbf{z}^-} e^{\mathrm{sim}(\mathbf{z}, \mathbf{z}^-)/\tau}}, \tag{6}$$

where $\mathbf{z}$ and $\mathbf{z}^+$ denote embeddings from the same object instance, $\mathbf{z}^-$ denotes embeddings from different instances, τ is a temperature parameter, and $\mathrm{sim}(\cdot, \cdot)$ denotes cosine similarity. We freeze the visual encoder and train the learnable adapter using the given template images and template-based synthetic images, as described in Sec. III-C0b.

After obtaining the candidate embeddings and the corresponding template embedding, we compute the cosine similarity between each candidate embedding and the template embedding from Eq. (3) and Eq. (4):

$$s_i^k = \frac{\mathbf{z}_i^k \cdot \mathbf{z}_{\text{temp}}^k}{\|\mathbf{z}_i^k\|_2 \, \|\mathbf{z}_{\text{temp}}^k\|_2}, \tag{7}$$

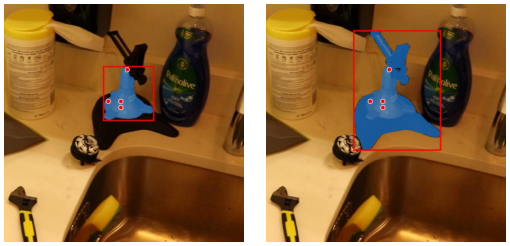
where s_i^k denotes the similarity score between the i-th candidate embedding and the template embedding for the k-th template image. The candidates are then ranked according to their similarity scores. We first identify the candidate point with the maximum similarity score: $s_{\max}^k = \max_i \; s_i^k$. We then apply a filtering threshold δ and retain not only the candidate point with the highest score, but also all candidate points whose similarity scores differ from the maximum by less than δ:

$$\mathcal{P}^k = \left\{ \mathbf{p}_i^k \mid s_{\max}^k - s_i^k < \delta \right\}, \tag{8}$$

where $\mathcal{P}^k$ denotes the set of selected candidate points retained for the k-th template image after filtering. This strategy ensures that the most similar regions to the target instance are preserved, while also retaining secondary yet relevant local regions. Such regions often correspond to meaningful object parts (e.g., the head of the target object in the local mask $\mathcal{M}_i^k$ shown in the Candidate Selector of Fig. 2), which also exhibit high similarity scores. At the same time, low-confidence candidates originating from non-target regions are effectively filtered out. The above procedure is applied independently to all K template images. Finally, the selected points from all templates are aggregated to form the final set of selected candidate points: $\mathcal{P} = \bigcup_{k=1}^K \mathcal{P}^k$.

C. Augmented SAM

The selected points $\mathcal{P}$ obtained from the Candidate Selector (red points in Fig. 2) are designed to lie on the target instance as much as possible. However, sparse point prompts cannot guarantee coverage of all informative object parts. As illustrated in Fig. 3, the selected points $\mathcal{P}$ may concentrate



SAM SAM*

Fig. 3: Comparison between the original SAM and our augmented **SAM*** under candidate points. With all parameters frozen, SAM tends to produce incomplete masks focused on local regions around the prompts. In contrast, **SAM*** incorporates a learnable instance-specific object token, which guides the decoder to produce more complete object masks improving detection performance.

on the central region of the target instance, while other local regions remain uncovered. This phenomenon arises because certain object parts exhibit local features that differ significantly from the overall appearance of the instance, causing the corresponding candidate points to be filtered out by the *Candidate Selector*. As a result, using these selected points with frozen SAM typically leads to partial masks that cover only local regions of the target object. Therefore, we seek to further enhance SAM's ability to reconstruct complete and accurate masks to address this limitation.

a) Object token for mask completion: To this end, we introduce a learnable *object token* whose purpose is to guide SAM toward predicting the full object mask. The object token is injected into the SAM mask decoder together with the image tokens and prompt tokens, as illustrated in the **Augmented SAM** module of Fig. 2. The image encoder and prompt encoder are kept frozen, and therefore the corresponding image tokens and prompt tokens are also fixed during training.

Formally, given the query image $\mathcal{I}$ and a set of prompt points $\mathcal{P}$ after the *Candidate Selector*, the image tokens and prompt tokens are obtained as

$$\mathbf{T}_I = E_I(\mathcal{I}), \qquad \mathbf{T}_P = E_P(\mathcal{P}), \tag{9}$$

where $E_I(\cdot)$ and $E_P(\cdot)$ denote the frozen SAM image encoder and prompt encoder, respectively. For each object instance, we introduce an instance-specific learnable object token $\mathbf{t}_O$. The predicted mask is then produced by the frozen mask decoder conditioned on all three types of tokens:

$$\hat{\mathcal{M}} = D_{\text{mask}}(\mathbf{T}_I, \mathbf{T}_P, \mathbf{t}_O), \tag{10}$$

where $D_{\text{mask}}(\cdot)$ denotes the SAM mask decoder. During training, only the object token $\mathbf{t}_O$ is learnable, while all other components remain frozen.

b) Template-based synthetic training data: Based on the given template images and their corresponding mask images, we generate a set of synthetic training data to train both the adapter in the Candidate Selector and the object token in the Augmented SAM. Specifically, we composite the masked object regions from the template images onto publicly available open-world background images, which are provided in [40]. We consider three types of synthesized scenes:

1) **Single-object composition:** only the target object is pasted onto the background image.
2) **Multi-object composition without overlap:** the target object and several other objects are pasted onto the background, with additional objects placed around the target object without overlapping it.
3) **Multi-object composition with overlap:** both the target object and other objects are pasted onto the background, where objects may partially overlap. When the target object is in the frontmost position, it remains unoccluded; otherwise, it may be partially occluded by surrounding objects, simulating challenging real-world conditions.

These template-based synthetic images (shown in the supplementary material) enrich the training data and facilitate effective learning of both the adapter in the Candidate Selector and the object token in the Augmented SAM. Moreover, this template-based compositing strategy significantly reduces the computational cost and time required for data generation. Experimental results further demonstrate that such a simple synthesis approach can effectively improve model performance, without the need for additional overhead from complex generative models. This simplicity makes the proposed method particularly suitable for novel instance recognition in real-world environments.

c) Training Loss: Given a synthesized training image $\tilde{\mathcal{I}}$ and its ground-truth target mask $\tilde{\mathcal{M}}$, we randomly sample between 1 and m pixel locations within the target object mask. These sampled points are constrained to lie inside the object mask and are used as point prompts for SAM. The ground-truth object mask $\tilde{\mathcal{M}}$ serves as the supervision signal.

The object token is optimized using a hybrid segmentation loss composed of the binary cross-entropy loss [15], the Dice loss [30], and the IoU loss [36]:

$$\mathcal{L}_{\text{seg}} = \mathcal{L}_{\text{BCE}} + \lambda_{\text{Dice}}\,\mathcal{L}_{\text{Dice}} + \lambda_{\text{IoU}}\,\mathcal{L}_{\text{IoU}}, \tag{11}$$

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2|\hat{\mathcal{M}} \cap \tilde{\mathcal{M}}| + \epsilon}{|\hat{\mathcal{M}}| + |\tilde{\mathcal{M}}| + \epsilon}, \tag{12}$$

$$\mathcal{L}_{\text{IoU}} = 1 - \frac{|\hat{\mathcal{M}} \cap \tilde{\mathcal{M}}| + \epsilon}{|\hat{\mathcal{M}} \cup \tilde{\mathcal{M}}| + \epsilon}. \tag{13}$$

In the above equations, λ_{Dice} and λ_{IoU} denote the weighting coefficients for the Dice loss and the IoU loss, respectively. Here, $\tilde{\mathcal{M}}$ represents the ground-truth mask, and $\hat{\mathcal{M}}$ denotes the predicted mask.

d) Object token memory for incremental instance addition: To better address the challenge of novel instance detection in a template-based setting, we design an improved training strategy for SAM by introducing an object token memory that supports incremental and scalable instance addition. Each learned object token corresponds to a specific object instance and is stored in the memory after training. During inference, each object is associated with its own dedicated object token, ensuring that detecting or segmenting one object does not interfere with other instances. Since object tokens

are instance-specific, the appropriate object token at inference time is directly determined by the provided template images of the target instance.

As new object instances are introduced, new object tokens are appended to the memory, allowing the memory size to grow dynamically. Importantly, training a new object token does not modify previously stored tokens, thereby preventing catastrophic forgetting [20] of earlier instances. This parameter-isolated design enables continual learning [23] and provides a scalable framework for incorporating new novel instances over time, which is particularly beneficial for long-term deployment in open-world environments.

IV. EXPERIMENTS

A. Datasets and Settings

Datasets. We evaluate our method on two datasets designed for novel instance detection. The High-Resolution Instance Detection (HR-InsDet) dataset [40] contains 100 object instances and a total of 160 test images captured across 14 different indoor scenes. Each test image has a high resolution of 8192×6144. The test images are further categorized into *easy* and *hard* subsets according to object clutter and scene complexity. We follow the experimental settings in [40]. Specifically, FasterRCNN [38], RetinaNet [24], CenterNet [50], FCOS [45], and the transformer-based DINO [49] detector are trained using synthetic images generated by the Cut-Paste-Learn [11] strategy. This approach constructs training data by pasting object instances onto background images, enabling supervised training for instance detection. For non-learned methods, we directly employ **SAM** [19] and DINO [7, 33] to generate object proposals and visual features. In addition, we include NIDS-Net [28], a recent state-of-the-art method, which introduces a trainable adapter to enhance performance.

The RoboTools dataset [22] consists of 20 object instances evaluated in 24 complex scenes, with each test image having a resolution of 1920×1080. We evaluate methods with different proposal detectors. Specifically, $\text{OLN}_{\text{Corr.}}$ employs its own detection module [17], where a matching head based on correlation [26]. Several other methods utilize GroundingDINO as the proposal model.

Evaluation metrics. We adopt **Average Precision (AP)** as the primary evaluation metric for instance detection. AP is computed by averaging precision over multiple Intersection over Union (IoU) thresholds ranging from 0.50 to 0.95 with a step size of 0.05. We additionally report AP_{50} and AP_{75}, which correspond to IoU thresholds of 0.50 and 0.75, respectively.

Implementation details. All experiments are conducted using two NVIDIA A5000 GPUs. In the Candidate Selector, the residual adapter ratio α in Eq. (5) is set to 0.2. The filtering threshold δ in Eq. (8) is set to 0.01. We employ the Adam optimizer [18] with a learning rate of 5×10^{-4}, a batch size of 96, and train the adapter for 20 epochs. For the InfoNCE-style contrastive learning [32] objective, the temperature parameter τ is set to 0.07, and the ratio between positive and negative samples is 1:2. In the Augmented SAM module, all components of SAM are frozen except for the instance-specific object tokens. In the training loss Eq. (11), we set $\lambda_{\text{Dice}} = 0.5$ and $\lambda_{\text{IoU}} = 1.0$, respectively. We use the Adam optimizer with a learning rate of 5×10^{-3}, a batch size of 1, and train for 12 epochs.

For template-based synthetic training images, three types of synthesized scenes (Sec. III-C0b) are used with an equal ratio of 1:1:1. For data augmentation, we apply random scaling, random rotation, and random blur to the target object. The background images are sampled from the open-world background dataset provided in [40]. Ultimately, we generate 500 synthesized images for each target object.

For dense feature extraction, we employ DINOv3-large [42]. All SAM components are built upon SAM2-large [37]. Within the Candidate Selector, a pre-trained Perception Encoder (PE) [4] is used to extract visual embeddings for both candidate regions and template objects. For dense feature matching, we uniformly sample $S = 10$ patches inside the object mask of each template image. After the Candidate Selector obtains the selected candidates, we further choose the final prompt points using a farthest-point-first strategy, which prevents the prompts from being concentrated in a small local region and encourages spatial diversity for mask reconstruction. For the RoboTools benchmark and real-world robotic experiments, we directly perform inference on the full image. For the HR-InsDet benchmark, due to its high image resolution, we adopt a local-window inference strategy. Each local window has a resolution of 2048×1536, corresponding to one quarter of the original image resolution along each spatial dimension.

B. Benchmarking Results

HR-InsDet. Table I summarizes the results on the High-Resolution dataset. Our L2G-Det achieves a substantial improvement of **12.3 AP** over the top-performing baseline. Notably, on the hard subset, which contains severe occlusion and heavy clutter, L2G-Det yields an even larger gain of **17.6 AP**. These results demonstrate the effectiveness and robustness of our approach in complex real-world scenarios.

RoboTools. Table II reports the detection performance on the RoboTools dataset. Unlike proposal-based pipelines, our L2G-Det does not rely on explicit proposal generation; instead, it performs dense patch-level matching to obtain candidate points for subsequent mask reconstruction. As a result, L2G-Det outperforms the top-performing baseline, NIDS-Net [28], by **7.0 AP**. Qualitative comparisons on RoboTools are shown in Fig. 4, where our method produces more complete and accurate detections in cluttered scenes. More results can be found in the supplementary material.

C. Ablation Studies

We analyze the importance and effectiveness of each component in our model through a series of ablation studies. Unless otherwise specified or explicitly varied in a particular experiment, all ablation experiments follow the same settings described in the implementation details.

Effect of the Adapter and Augmented SAM. Across both datasets, we observe that enabling either the adapter

TABLE I: Comparison of detection performance on the HR-InsDet dataset [40].

Method	AP						AP_{50}	AP_{75}
	avg	hard	easy	small	medium	large		
FasterRCNN [38]	19.5	10.3	23.8	5.0	22.2	38.0	29.2	23.3
RetinaNet [24]	22.2	14.9	26.5	5.5	25.8	42.7	31.2	25.0
CenterNet [50]	21.1	11.8	25.7	5.9	24.1	40.4	32.7	23.6
FCOS [45]	22.4	13.2	28.7	6.2	26.5	38.1	32.8	25.5
DINO [49]	28.0	17.9	32.6	11.5	31.6	48.3	39.6	32.2
SAM + $DINO_f$ [19, 7]	37.0	22.4	43.9	11.9	40.9	62.7	44.1	40.4
SAM + $DINOv2_f$ [19, 33]	41.6	28.0	47.6	14.6	45.8	69.1	49.1	46.0
NIDS-Net [28]	63.9	43.4	72.7	18.1	62.5	84.0	76.6	70.6
L2G-Det (Ours)	**76.2**	**61.0**	**81.2**	**24.1**	**75.4**	**92.6**	**80.6**	**77.9**

TABLE II: Comparison of detection performance on RoboTools dataset [22]. The *Proposal* column indicates the proposal generation strategy. Specifically, OLN denotes Object Localization Network [17], GD denotes GroundingDINO [25].

Method	Proposal	AP	AP_{50}	AP_{75}
OS2D [34]	N/A	2.9	6.5	2.0
DTOID [29]	N/A	3.6	9.0	2.0
$OLN_{Corr.}$ [17]	OLN	14.4	18.1	15.7
VoxDet [22]	OLN	18.7	23.6	20.5
OTS-FM [41, 40]	GD	56.7	64.8	59.0
IDOW [41]	GD	59.4	67.8	61.8
NIDS-Net [28]	GD	64.9	79.4	70.8
L2G-Det (Ours)	**N/A**	**71.9**	**84.6**	**77.2**

or Augmented SAM (i.e., introducing instance-specific object tokens) individually leads to consistent improvements over the base L2G-Det framework. As summarized in Table. III, when both components are jointly applied, the detection performance is further enhanced, achieving the best results. Notably, the performance gains are more pronounced on the High-resolution dataset. This can be attributed to the presence of many visually similar objects, as well as increased object overlap and occlusion in query images. In such challenging scenarios, the adapter improves instance-level feature

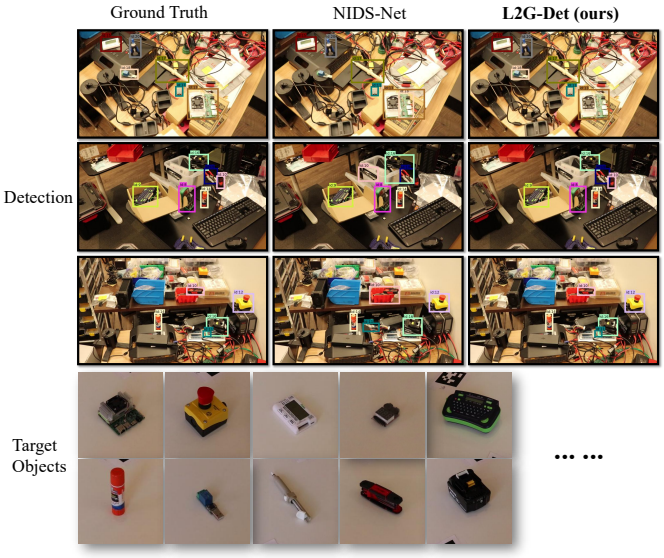


Fig. 4: Qualitative results on RoboTools benchmark. From left to right, we show the ground-truth annotations, results produced by NIDS-Net [28], and results of our method **L2G-Det**.

TABLE III: Ablation study on the effects of the Adapter in Candidate Selector and Augmented SAM on RoboTools [22] and High-resolution datasets [40].

Method	**Adapter**	**Augmented SAM**	**AP**
RoboTools Dataset			
w/o Adapter + SAM	✗	✗	64.8
Adapter + SAM	✓	✗	67.5
w/o Adapter + SAM*	✗	✓	69
Adapter + SAM*	✓	✓	**71.9**
High-resolution Dataset			
w/o Adapter + SAM	✗	✗	58.9
Adapter + SAM	✓	✗	66.5
w/o Adapter + SAM*	✗	✓	65.4
Adapter + SAM*	✓	✓	**76.2**

discrimination during candidate selection, while Augmented SAM effectively recovers complete object masks from sparse prompts. Overall, these results demonstrate the effectiveness and complementarity of the two components in improving the detection capability of L2G-Det.

Candidate selector design. We analyze the design choices of the proposed Candidate Selector on the RoboTools dataset, as summarized in Table IV. Here, we do not consider instance-specific object tokens and use the basic SAM model for mask generation, in order to isolate the effect of the Candidate Selector. When the Candidate Selector is removed, the *w/o Filtering* setting directly uses all candidate points produced by dense feature matching as prompts for SAM to generate masks. *Filtering*, where candidate points are selected based on their dense matching scores following Eq. (8), leads to a certain performance improvement compared to using all candidate points. This indicates that filtering low-confidence matches can partially suppress noisy prompts. Despite this improvement, the performance of score-based filtering remains significantly inferior to that achieved with the full Candidate Selector, demonstrating the importance of the Candidate Selector, that suppresses false positives caused by local appearance ambiguities beyond simple score thresholding.

We further evaluate the complete Candidate Selector by comparing different visual encoders used for candidate embedding extraction. The *DINO-CLS* encoder uses the class token generated by DINOv3 [42] as the candidate embedding. In contrast, the *Perception Encoder (PE)* [4] consistently achieves better performance. This suggests that PE provides stronger semantic representations, enabling better association between local object parts and the overall instance appearance, which leads to improved novel instance detection performance.

TABLE IV: Ablation study of the Candidate Selector design.

(1) w/o Candidate Selector

Configuration	**AP**	**AP_{50}**	**AP_{75}**
w/o Filtering [43]	48.7	60.5	51.6
With Filtering [33]	53.4	64.6	58.1

(2) With Candidate Selector

Encoder	**AP**	**AP_{50}**	**AP_{75}**
DINO-CLS [42]	62.7	75.4	68.0
PE [4]	**64.8**	**77.6**	**70.2**

Effect of the number of template images. We study the impact of the number of template images K on detection performance on the RoboTools dataset. As shown in Fig. 5,

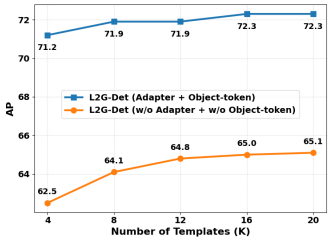


Fig. 5: Effect of the number of template images K.

TABLE V: Comparison of different training strategies.

Training Strategy	AP	AP_{50}	AP_{75}
CL-Joint	69.5	82.7	74.8
Joint	69.9	82.6	75.3
SAM* (Ours)	**71.9**	**84.6**	**77.2**

increasing K consistently improves performance for both the basic L2G-Det framework and the full model. This trend indicates that incorporating more template observations provides richer appearance coverage of the target instance, leading to more reliable local matching and candidate generation.

For the baseline L2G-Det framework, we observe that performance gains become marginal once the number of template images exceeds $K = 12$. The full L2G-Det model equipped with the adapter and instance-specific object tokens reaches a high performance level at $K = 8$, after which the improvement gradually saturates. Overall, considering the trade-off between detection accuracy and the cost of collecting template images, we adopt $K = 12$ as the default number of templates for L2G-Det. This choice achieves near-peak AP while significantly reducing the reliance on additional template observations.

Training strategies. We compare different training paradigms on RoboTools in Table V. *Joint* represents the conventional strategy that mixes training data from all 20 objects and optimizes a single model jointly. *CL-Joint* evaluates a continual-learning setting [23] by splitting the 20 objects into four sequential tasks (objects 1–5, 6–10, 11–15, and 16–20). Training proceeds task-by-task in order, while objects within the same task are still jointly trained. In an additional per-task analysis, earlier objects show a slight performance drop after learning subsequent tasks; for example, the AP of objects 1–5 decreases from 70.5 to 69.9. Finally, SAM* (*Augmented SAM*) corresponds to our proposed instance-specific object-token training, where each object is associated with its own learnable token. As shown in Table V, SAM* achieves the best performance, indicating that instance-specific tokenization provides a more effective and scalable training strategy than joint-training baselines.

Effect of dense feature extractors. We evaluate the impact of different dense feature extractors used in the dense matching stage on the RoboTools dataset, summarized in Table VI. We compare LoFTR [43] with large-scale foundation models, including DINOv2-Large [33] and DINOv3-Large [42]. For the base L2G-Det framework, replacing LoFTR with DINO-based dense features leads to substantial performance improvement, demonstrating the importance of strong semantic representations for reliable local matching. Particularly, DINOv3-Large slightly outperforms DINOv2-Large. These results further indicate that the final instance detection performance is positively correlated with the quality of the dense feature extractor, and validate our choice of DINOv3-Large as the default dense feature extractor in L2G-Det.

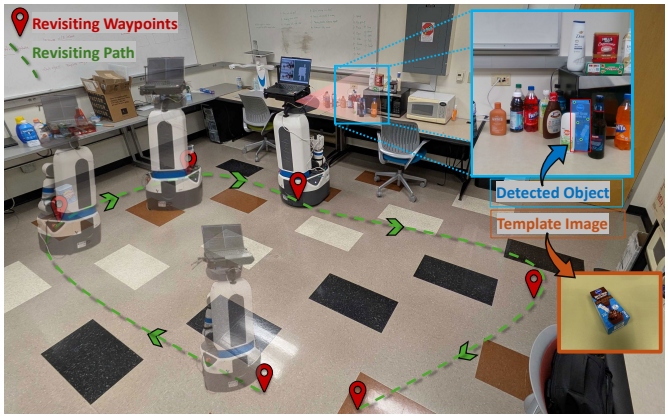


Fig. 6: Overview of the robot experiment process showing the waypoint tracking and target object identification.

TABLE VI: Effect of different dense feature extractors used in the dense matching stage.

(1) L2G-Det w/o Adapter + SAM

Dense Backbone	AP	AP_{50}	AP_{75}
LoFTR [43]	41.3	49.2	44.4
DINOv2-Large [33]	64.4	76.3	69.8
DINOv3-Large [42]	**64.8**	**77.6**	**70.2**

(2) L2G-Det with Adapter + SAM*

Dense Backbone	AP	AP_{50}	AP_{75}
LoFTR [43]	48.3	56.8	51.7
DINOv2-Large [33]	71.4	83.8	76.6
DINOv3-Large [42]	**71.9**	**84.6**	**77.2**

Effect of backbone choice. To ensure a fair comparison with other instance detection methods, we use the same SAM ViT-Large backbone as the compared baselines and vary the DINO backbone. Specifically, we compare L2G-Det with the same or smaller DINOv2 [33] backbones. As shown in Table VII, L2G-Det consistently outperforms the compared methods even when using the same or smaller DINOv2 backbones. In particular, L2G-Det with DINOv2-Small already achieves higher performance than NIDS-Net [28] with DINOv2-Large. This indicates that, although stronger dense features can further improve performance, the main improvement comes from the proposed local-to-global pipeline design rather than simply using a larger pretrained backbone.

TABLE VII: Effect of backbone choice on RoboTools benchmark with the adapter and augmented SAM.

Method	Backbone	AP	AP_{50}	AP_{75}
IDOW [41]	DINOv2-Small	51.9	63.8	56.5
NIDS-Net [28]	DINOv2-Large	64.9	79.4	70.8
L2G-Det (Ours)	DINOv2-Small	69.2	81.1	74.3
L2G-Det (Ours)	**DINOv2-Base**	**70.8**	**82.9**	**76.1**

D. Real-World Robotic Experiments

To evaluate the applicability of our approach in open-world settings, we deploy our system on an RTX 4090 laptop, mounted on a Fetch robot, connected via Ethernet. Given the template images of novel target objects, the robot searches for and localizes the objects in cluttered indoor environment. For inference, we mount a Realsense D415 camera on the robot head and stream the RGB images (1280×720, 15fps).

TABLE VIII: Real-world robotic detection results on 8 objects. $N_{\text{IoU}>\tau}$ denotes the number of detections whose predicted masks achieve IoU $> \tau$ (threshold) w.r.t ground-truth annotations. N_{Success} denotes the number of trials in which the robot successfully detects the target object and stops. There are 8 trials in total.

Method	$\mathbf{N}_{\text{IoU}>0.5}$	$\mathbf{N}_{\text{IoU}>0.75}$	$\mathbf{N}_{\text{IoU}>0.95}$	$\mathbf{N}_{\text{Success}}$
L2G-Det with SAM	7	7	7	8
L2G-Det with SAM*	**8**	**8**	**7**	**8**

We first map the environment with ROS Gmapping [14], record the robot trajectory and extract a revisit path of 6 waypoints using the procedure mentioned in [1]. Next, the robot localizes using AMCL ROS [12] and revisits these waypoints, while we run real-time RGB inference (Fig. 6). We tested 8 objects with 2 settings: 1) Augmented SAM with instance-specific object tokens; 2) using the pretrained SAM. Based on the predicted mask, we compute the matching score using cosine similarity following Eq. (7). If the matching score exceeds the threshold (here 0.7), the corresponding detection is accepted, and the trial stops at that waypoint. Table VIII shows that **L2G-Det** robustly detects all 8 objects in real-world environments. The augmented SAM yields more accurate and complete masks, improving performance under stricter IoU thresholds. Experimental videos are included in the supplementary material.

E. Computational Cost

We further report the computational cost of L2G-Det in Table IX. The real-world robotic experiments are deployed on a laptop with an NVIDIA RTX 4090 GPU, while the benchmark experiments on RoboTools and HR-InsDet are conducted using two NVIDIA RTX A5000 GPUs for training and inference. The reported inference runtime is the average end-to-end per-object cost measured over the entire benchmark. Since we sample only $S = 10$ patches per template image and use a small number of templates K, the number of dense matching features remains limited. While the current implementation still has non-trivial computational overhead, the improved detection accuracy suggests that the proposed local-to-global formulation provides a practical and effective solution for open-world robotic perception.

TABLE IX: Summary of training-related and inference cost. The inference runtime is reported as the end-to-end per-object cost with $K = 4$ template images.

Training-related Cost	**Runtime**	**GPU Memory**
Create synthetic images (per object)	3 min	∅
Train object token (per object)	25 min	2400 MiB
Train Adapter (all objects)	2 h	6900 MiB
Inference Cost ($K = 4$)	**Runtime**	**GPU Memory**
RoboTools (1920×1080)	1.10 s	8300 MiB
HR-InsDet (8192×6144)	4.10 s	20000 MiB

Table X further reports the inference cost under different numbers of template images K on RoboTools. This reflects the trade-off between efficiency and performance: using fewer templates reduces inference time, while using more templates can improve robustness in the zero-shot setting at the cost of additional runtime.

TABLE X: Inference runtime and GPU memory with different numbers of template images K on RoboTools.

K	**Runtime**	**GPU Memory**
4	1.1 s	8300 MiB
8	1.6 s	9000 MiB
12	1.9 s	9800 MiB
16	2.1 s	10500 MiB

V. Conclusion

We introduced L2G-Det, a local-to-global framework for novel object instance detection and segmentation in open-world environments. By replacing proposal-based pipelines with dense patch-level matching and mask reconstruction, our approach robustly detects novel objects under occlusion, viewpoint variation, and background clutter. We further proposed a candidate selection module to suppress false positives and an augmented SAM with instance-specific object tokens to recover coherent global masks from sparse prompts. Experimental results on benchmark datasets and real-world robotic scenarios demonstrate the effectiveness and robustness of the proposed approach, highlighting its potential for scalable instance-level perception in open-world robotics.

Limitations. First, our framework integrates multiple pretrained models for dense feature extraction, candidate selection, and mask generation, rather than employing a fully end-to-end detector. As a result, it requires higher computational resources compared to conventional end-to-end detection pipelines. Second, the learning of instance-specific object tokens relies on template-based synthetic training images, where target objects are generated by simple copy-and-paste operations onto background images, which may not fully capture complex real-world interactions. Exploring generative models to produce more realistic training images is an important direction for future work.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under Grant Nos. 2346528 and 2520553, and the NVIDIA Academic Grant Program Award, and a gift funding from XPeng.

References

[1] Sai Haneesh Allu, Itay Kadosh, Tyler Summers, and Yu Xiang. A modular robotic system for autonomous exploration and semantic updating in large-scale indoor environments, 2025. URL https://arxiv.org/abs/2409.15493.

[2] Phil Ammirato, Cheng-Yang Fu, Mykhailo Shvets, Jana Kosecka, and Alexander C Berg. Target driven instance detection. *arXiv preprint arXiv:1803.04610*, 2018.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision (ECCV)*, pages 404–417, 2006.

[4] Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. *arXiv:2504.13181*, 2025.

[5] Richard Bormann, Xinjie Wang, Markus Völk, Kilian Kleeberger, and Jochen Lindermayr. Real-time instance detection with fast incremental learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021. doi: 10.1109/ICRA48506.2021.9561202.

[6] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts, 2025. URL https://arxiv.org/abs/2511.16719.

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[8] Tianrun Chen, Lanyun Zhu, Chaotao Deng, Runlong Cao, Yan Wang, Shangzhan Zhang, Zejian Li, Lingyun Sun, Ying Zang, and Papa Mao. Sam-adapter: Adapting segment anything in underperformed scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3367–3375, 2023.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020.

[11] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[12] Eitan Marder-Eppstein, David V. Lu, Michael Ferguson, and Aaron Hoy. Ros navigation stack. URL https://github.com/ros-planning/navigation.

[13] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv 2110.04544*, 2021.

[14] Brian Gerkey. slam_gmapping, 2013. URL https://github.com/ros-perception/slam_gmapping.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[16] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, and Serge Belongie. Visual prompt tuning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 709–727, 2022.

[17] Dahun Kim, Tsung-Yi Lin, Anelia Angelova, In So Kweon, and Weicheng Kuo. Learning open-world object proposals without learning to classify. *IEEE Robotics and Automation Letters*, 7(2):5453–5460, 2022.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[20] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the National Academy of Sciences (PNAS)*, volume 114, pages 3521–3526, 2017.

[21] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059, 2021.

[22] Bowen Li, Jiashun Wang, Yaoyu Hu, Chen Wang, and Sebastian Scherer. Voxdet: Voxel learning for novel instance detection. *Advances in Neural Information Processing Systems*, 36, 2024.

[23] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1. IEEE, 2017.

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[25] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[26] Yuan Liu, Yilin Wen, Sida Peng, Cheng Lin, Xiaoxiao Long, Taku Komura, and Wenping Wang. Gen6d: Generalizable model-free 6-dof object pose estimation from rgb images. In *European Conference on Computer Vision*,

pages 298–315. Springer, 2022.

[27] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[28] Yangxiao Lu, Jishnu Jaykumar P, Yunhui Guo, Nicholas Ruozzi, and Yu Xiang. Adapting pre-trained vision models for novel instance detection and segmentation, 2024.

[29] Jean-Philippe Mercier, Mathieu Garon, Philippe Giguere, and Jean-Francois Lalonde. Deep template-based object instance detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1507–1516, 2021.

[30] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, 2016.

[31] Van Nguyen Nguyen, Thibault Groueix, Georgy Ponimatkin, Vincent Lepetit, and Tomas Hodan. Cnos: A strong baseline for cad-based novel object segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2134–2140, 2023.

[32] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[33] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv:2304.07193*, 2023.

[34] Anton Osokin, Denis Sumin, and Vasily Lomakin. Os2d: One-stage one-shot object detection by matching anchor features. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 635–652. Springer, 2020.

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[36] Md. Asiful Islam Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International Symposium on Visual Computing (ISVC)*, 2016.

[37] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL https://arxiv.org/abs/2408.00714.

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[39] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[40] Qianqian Shen, Yunhan Zhao, Nahyun Kwon, Jeeeun Kim, Yanan Li, and Shu Kong. A high-resolution dataset for instance detection with multi-view instance capture. In *NeurIPS Datasets and Benchmarks Track*, 2023.

[41] Qianqian Shen, Yunhan Zhao, Nahyun Kwon, Jeeeun Kim, Yanan Li, and Shu Kong. Solving instance detection from an open-world perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[42] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3, 2025. URL https://arxiv.org/abs/2508.10104.

[43] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021.

[44] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in neural information processing systems*, 33:6827–6839, 2020.

[45] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9626–9635, 2019. doi: 10.1109/ICCV.2019.00972.

[46] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[47] Yifan Wang, Xingyi He, Sida Peng, Dongli Tan, and Xiaowei Zhou. Efficient LoFTR: Semi-dense local feature matching with sparse-like speed. In *CVPR*, 2024.

[48] Yang Yu, Chen Xu, and Kai Wang. Ts-sam: Fine-tuning segment-anything model for downstream tasks. *arXiv preprint arXiv:2408.01835*, 2024.

[49] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection, 2022.

[50] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.