# BYE: Build Your Encoder with One Sequence of Exploration Data for Long-Term Dynamic Scene Understanding and Navigation

**Chenguang Huang**
Department of Computer Science
University of Freiburg Germany
huang@informatik.uni-freiburg.de

**Wolfram Burgard**
Technical University of Nuremberg
Germany
wolfram.burgard@utn.de

**Abstract:** Dynamic scene understanding has long been a challenge in robotic applications. Earlier approaches to dynamic mapping focused on mitigating the impact of short-term dynamic objects in view, typically by removing or tracking masks for specific object categories while estimating camera motion. However, these methods often struggle to handle long-term scene changes. Recent efforts have addressed the object association problem in long-term dynamic environments using neural networks trained on synthetic datasets, though these approaches still rely on predefined object shapes and categories. Other methods leverage visual, geometric, or semantic clues as heuristics for association. In this work, we introduce BYE, a class-agnostic per-scene point cloud encoder that eliminates the need for predefined categories, shape priors, or large association datasets. It requires only a single sequence of exploration data for training and can efficiently perform object association in the face of dynamic changes.

**Keywords:** Representation Learning, Dynamic Mapping, Robot Navigation, Scene Understanding

## 1 Introduction

Interacting with the physical environment is a dynamic experience. On the one hand, numerous objects are moving in view in our daily lives such as humans, animals, cars, and machines and we need to reason, react, avoid, maneuver, or operate, processing the information instantly. On the other hand, beyond our sight, the environment is also evolving. Objects were moved to new places, drawers were pulled out, screens were turned off, and containers were refilled. When we revisit or explore, we constantly associate new observations with our past experiences and keep updating our knowledge base. We ask the question: can a robot have the ability to associate new observations with its previous knowledge base?

Despite the recent spike of robot generalist policy research [1, 2, 3, 4] which aims at training a reactive robot policy that handles dynamics through learning from a plethora of data, the majority of the robotic systems still rely on a knowledge base of the environment to operate within a certain scope. Such a knowledge base is usually a scene representation, namely, a map. Previous dynamic mapping techniques focused on reducing the impact of short-term dynamic objects in view by removing or tracking certain classes of semantic masks during camera pose estimation [5, 6, 7, 8, 9, 10, 11], which struggles to handle long-term dynamic scenes. Later approaches to long-term dynamic scene understanding used visual, geometric, or semantic clues as heuristics for associations between scenes [12, 13]. However, these methods require perfect registration of two scenes. Recently, researchers proposed to train an encoder-decoder relying on synthetic datasets to perform scene object association, registration, and reconstruction across object location changes [14]. However, the method still depends on a predefined set of object categories and shapes in ShapeNet [15].

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

To this end, we introduce BYE, a class-agnostic per-scene point cloud encoder that removes the need for predefined categories, shape priors, or large association datasets. It only requires a sequence of exploration data to train and can generalize the association ability to long-term dynamic changes in the environment. BYE highly resembles a human's ability to recall objects and their past locations when seeing objects with similar shapes and appearances in new places. The idea of BYE stems from the work LangSplat [16] which trains a per-scene autoencoder to encode CLIP features of all images collected in the same scene into a low dimensional latent space to accelerate the optimization of Gaussian Splatting [17]. In this work, our major contributions are:

1. We introduce BYE, a novel pipeline to train a per-scene point cloud encoder for object association in long-term dynamic environments. The training requires only one sequence of exploration data in the scene.
2. We propose to construct an object memory bank with BYE encoder and all partial point cloud observations in the exploration data for object association in dynamic scenes, resembling human memory of past experiences.
3. We evaluate our BYE in a photorealistic simulator AI2THOR [18] and our method outperforms heuristic baselines based on vision language foundation models.

## 2 Related Work

**Open-Vocabulary Semantic Mapping:** In recent years, the advancement of Vision Language Models and their fine-tuned counterparts [19, 20, 21] have innovated the mapping techniques in robotic applications such as navigation [22, 23, 24, 25, 26, 27, 28, 29, 30, 31], manipulation [32, 33, 34], and 3D semantic scene understanding [35, 36, 37, 16, 38, 39, 40]. By integrating visual-language features into sparse topological nodes [22, 30, 31], dense 3D voxels or 2D pixels [24, 36, 41], discrete 3D locations [25, 27, 26, 29], or implicit neural representations [35, 38, 16, 39], those created maps can be used to retrieve concepts with natural language descriptions, extending closed-set semantics retrieval [42, 43, 44, 11] to open-vocabulary level and enabling more flexible and efficient human-robot interaction in the downstream tasks. However, most of the open-vocabulary semantic mapping approaches assumes a static environment, struggling to readjust the contents to scene changes in a long-term dynamic environment. In this work, we propose training a scene-wise point cloud encoder to extract class-agnostic, instance-level features, which are stored in an object memory bank to manage future observations of dynamic scene changes.

**Dynamic Scene Understanding:** Understanding dynamic environments has posed ongoing challenges in both academia and industry. One key challenge is the constant motion within the scene, which complicates tracking and mapping. Early approaches used semantic segmentation masks to filter or track specific object classes during SLAM optimization [5, 6, 7, 8, 9]. However, these methods rely on assumptions about static and dynamic categories that are not always valid. For instance, in outdoor environments, a "car" is classified as dynamic whether it is parked or moving. Other methods exploited dense tracking like optical flow to model the motion of objects [45], even achieving non-rigid object tracking [46]. In recent years, advances in neural implicit scene representation such as NeRF [47] and Gaussian Splatting [17] allows us to render novel views in a dynamic scene at any historical timestep [10, 48, 49] and even simulate the dynamics based on physical rules [50]. Another challenge in dynamic environments is handling long-term scene changes. Key tasks in this area include change detection and change association, which involve tracking changes such as object displacement, addition, or removal across several sequences of observations. Several datasets have been developed to support research in this field [51, 52, 53, 54]. Some methods tackled change detection by registering two reconstructed scenes, subtracting them, and exploiting visual, geometric, or semantic clues for matching [12, 55, 13]. Zhu et al. [14] proposed to train an encoder-decoder network to learn the association, registration, and reconstruction of objects across changes, while Qiu et al. [56] trained a model to take observations from two sequences and generate scene change captions. Other approaches has used proximity graphs [55] or object relationship scene graphs [57] to generate features for association. Recently, probabilistic and optimization-based ap-

proaches have been applied to build systems addressing change detection, association, and SLAM in a pipeline [58, 59, 60]. In our work, we tackle long-term change association based on an encoder trained with a sequence of observations data, and create an object memory bank storing all object latent representations generated by the encoder for association of new observations.

**Shape Representation Learning:** A key technique to enhance the understanding of 3D scenes or objects is 3D shape representation learning. Early approaches focusing on 3D semantic understanding learned global or point-wise representations of point clouds [61, 62, 63], which nowadays serve as strong backbones for advanced methods. Other works learned implicit functions resembling traditional 3D shape representations like SDFs and occupancy grids with neural networks [64, 65]. Stemming from the techniques above, other methods explore learning the neural descriptor fields that map spatial locations relative to a shape to latent features for shape completion [66, 67], registration [68, 69], manipulation [70], and object-level SLAM [71]. While the methods in those applications above have shown promising results, most of them require a curated 3D shape dataset containing a variety of complete shapes spanning different categories. In this work, our method only needs the observation data during the exploration of a scene to train a scene-wise encoder which can be used for object association in a long-term dynamic environment. By following the strategy introduced in SimCLR [72], we train an efficient encoder that attracts partial point cloud observations of the same object while repelling those of different instances.

## 3 Problem Definition

To formalize the problem, we first define a long-term dynamic environment where the locations of objects can change between different trials of exploration, but each object remains static during a single trial. For each exploration trial, at every time step, we assume access to the following data: an RGB image $\mathbf{I}_t$, a depth image $\mathbf{D}_t$, 2D instance segmentation masks $\mathcal{M}_t = \{\mathbf{M}_{tk}\}_{k=1,2,\dots,K}$, and the camera pose $\mathbf{T}_t$. From these data, we can construct an instance-level map where each object is represented as an independent point cloud. Let the set of point clouds be denoted as $\{\mathcal{P}_i\}_{i=1,2,\dots,M}$, where $\mathcal{P}_i$ represents the point cloud of the $i$-th object, and $M$ is the total number of objects (instances) in the scene.

The problem of long-term object change detection and association is defined as follows: given two sets of exploration data, collected before and after several object location changes, find the bijective mapping of object IDs between the two trials. That is, for each object in the second trial, we must identify the corresponding object in the first trial. Formally, given the two sets of point clouds $\{\mathcal{P}_i^{ref}\}_{i=1,2,\dots,M}$ from the first trial and $\{\mathcal{P}_j^{new}\}_{j=1,2,\dots,M'}$ from the second trial (after changes), the goal is to find a bijection $f : 1, 2, \dots, M' \to 1, 2, \dots, M$ such that if $f(j) = i$, $\mathcal{P}_j^{new}$ and $\mathcal{P}_i^{ref}$ correspond to the same object in both trials, where $M = M'$ after excluding any added or removed objects.

## 4 Method

This work aims to train a per-scene point cloud encoder that extracts latent features of partial object point cloud observations. With contrastive learning, we ensure that observations from the same object have similar embeddings while those from different objects are distinct from one another. By using the encoder to generate latent embeddings of all partial point cloud observations of all instances in the reference trial of the exploration, we can build a memory bank for the scene objects. Later, during the new trial of exploration after object relocations have happened, we can use the same encoder to get the latent embeddings of new point cloud observations, find nearest neighbors in the memory bank, and thus associate to an instance in the reference trial.

The idea of training a "per-scene" encoder is inspired by LangSplat [16] that trained an autoencoder mapping high dimensional CLIP features [19] for images collected in a single scene to low dimensional ones, which accelerates the optimization of CLIP-enriched Gaussian Splatting. In our work, we follow the idea of training a network only for data collected in a single scene during one trial of

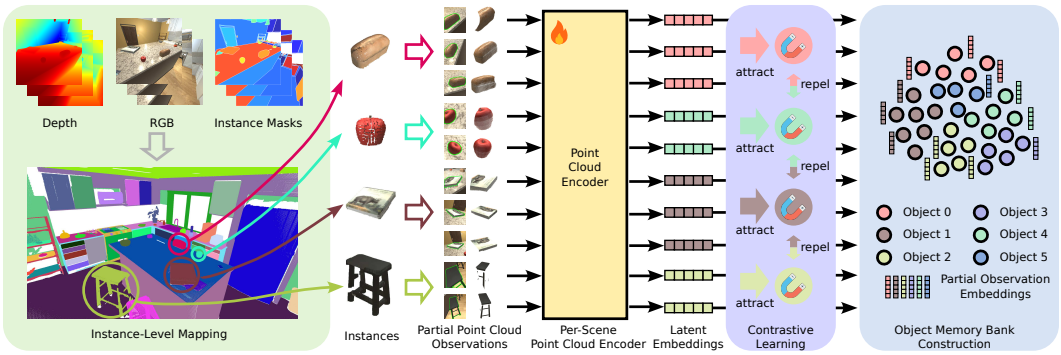


Figure 1: Overview of the pipeline of BYE for long-term dynamic environment understanding. With the reference trial of exploration data, we first build an instance-level map using the RGB, depth, instance masks, and odometry inputs, from which we generate a partial object point cloud observations dataset. Later, we exploit the principles of contrastive learning to train a point cloud encoder from scratch. Finally, we encode all the partial observations in the dataset into latent embeddings and associate them with instance labels in the reference exploration trial as the object memory bank.

exploration which inherently closes the door of generalization in novel scenes. However, this could facilitate object change detection and association in long-term dynamic environments.

In the following, we introduce (i) the construction of an instance-level map which is the prerequisite of generating training data (Sec. 4.1), (ii) the generation of partial point cloud observation dataset (Sec. 4.2), (iii) the training method of a per-scene object point cloud encoder with contrastive learning (Sec. 4.3), (iv) the generation of memory bank (Sec. 4.4), and lastly (v) object association with the memory bank (Sec. 4.5). The overview of the pipeline is shown in Fig 1.

### 4.1 Instance-Level Map Construction

We construct the instance-level map with the reference trial of exploration before object relocation. Given the instance segmentation masks $\mathcal{M}_t = \{\mathbf{M}_{tk}\}_{k=1,2,...,K}$, depth image $\mathbf{D}_t$, and camera pose $\mathbf{T}_t$ of each frame $t$ in a trial of exploration, we can easily back-project the instance masks into 3D through the depth image, transform them to the global coordinate frame, and either fuse them with existing global instance point clouds with the instance IDs of those masks or initialize new global instances. After iterating the process in each frame, we can obtain a list of point clouds $\{\mathcal{P}_i\}_{i=1,2,...,M}$ with instance IDs $i = 1, 2, ..., M$ in this trial of exploration. For simplicity, we assume known instance masks and odometry as input to emphasize the effectiveness of our trained encoder and mitigate the impact of the quality of segmentation and odometry results. However, this can be easily extended with any existing instance-level mapping techniques such as ConceptGraphs [30], HOV-SG [31], and so on.

### 4.2 Partial Point Cloud Observation Data Generation

After obtaining the instance-level map $\{\mathcal{P}_i\}_{i=1,2,...,M}$, for each instance point cloud $\mathcal{P}_i$, we find all the instance masks used to generate it and back-project them into camera coordinate frame to get a list of partial point cloud observations of the instance $\{\mathcal{P}_{ir}^{cam}\}_{r=1,2,...,R_i}$ where $R_i$ is the total number of masks for object $i$ during this trial of exploration. Each $\mathcal{P}_{ir}^{cam}$ contains a list of 6-dimensional vectors each storing the 3D position and the RGB values of a point. We then subtract each point cloud's 3D coordinate with its mean to obtain a zero-center point cloud $\bar{\mathcal{P}}_{ir}$ where $\mathbf{x}_{\bar{\mathcal{P}}} = \mathbf{x}_{\mathcal{P}} - \frac{1}{|\mathcal{P}|}\sum_{\mathbf{x}\in\mathcal{P}}\mathbf{x}$ where $\mathbf{x}_{\bar{\mathcal{P}}} \in \mathbb{R}^3$ and $\mathbf{x}_{\mathcal{P}} \in \mathbb{R}^3$. For each zero-center point cloud $\bar{\mathcal{P}}_{ir}$, we take its instance id $i$ as label, forming one data sample as a tuple $(\bar{\mathcal{P}}_{ir}, i)$. For simplicity, we denote all point clouds and their object ID labels as $\{(\bar{\mathcal{P}}_k, y_k)\}_{k=1,2,...,L}$ where $y_k$ is the object ID, and $L = \sum_{i=1}^{M} R_i$ is the total masks number of all objects.

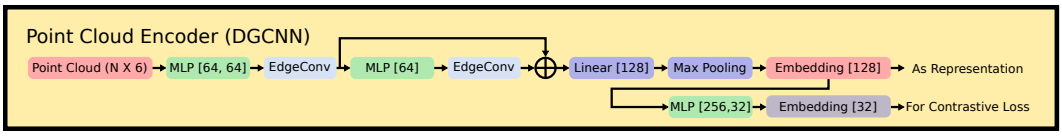


Figure 2: The architecture of the point cloud encoder. We first follow the architecture of DGCNN [63] and the training scheme of SimCLR [72], which add one more MLP layer without normalization following of the embedding output layer and project the representation to low dimensional space for more efficient contrastive learning.

### 4.3 Training Scene-wise Object Point Cloud Encoder

The main goal of the encoder is to extract a latent representation for a point cloud in a scene so that the point clouds belonging to the same object have similar embeddings while the point clouds from different objects are far away from one another in the embedding space. In Sec. 4.2, we obtain a dataset of point cloud and object label pairs. In this section, we will walk through the pipeline of training the encoder with contrastive learning. Following the idea of SimCLR [72], the training pipeline is comprised of five major components: (i) a data preprocessing step, (ii) a stochastic data augmentation module, (iii) a neural network base encoder $\mathcal{E}(\cdot)$, (iv) a small neural network projection head $\mathbf{g}(\cdot)$, and (v) a contrastive loss function. The architecture of the encoder is shown in Fig. 2. In the following, we will walk through these components.

**Data Preprocessing:** Due to the variation in the size of different objects, we need to sample the point clouds to ensure the balance of the training workload. In this work, we use a mixed sampling strategy. For point clouds with more than 1024 points, we first apply voxel-downsampling to a resolution of 0.01 meter. If the points number still exceeds 1024, we apply farthest point sampling to select 1024 points. In this way, we guarantee that the points number is below 1024 and ensure an efficient training process.

**Data Augmentation:** In this work, we sequentially apply random jittering of the point positions with a range of 0 to 0.03 meter, rotation of the whole point cloud around the X-axis, around the Y-axis, and around the Z-axis of the point cloud $\bar{\mathcal{P}}_k$. The rotation around each axis has a range of 0 to 30 degrees.

**The Neural Network Base Encoder.** We choose the architecture of DGCNN [63] as the backbone of our method. Thanks to its ability to dynamically capture local geometric relationships, the dynamic edge convolutional network can efficiently scale to handle complex structures or sparse point clouds. Furthermore, the pipeline is compatible with other base encoder backbones [61, 62], allowing for potential improvements when better point cloud processing architectures are discovered. In this work, we sequentially pass the point cloud into two MLP layers, one dynamic edge CNN, one MLP layer, one dynamic edge CNN, and one linear layer to generate the embeddings $\mathbf{h}(\bar{\mathcal{P}})$ of the point cloud as is shown in Fig. 2.

**Embedding projection head:** As in SimCLR [72], we project the embeddings $\mathbf{h}(\bar{\mathcal{P}})$ generated by the backbone above into a low dimensional space $\mathbf{g}(\bar{\mathcal{P}}) = MLP(\mathbf{h}(\bar{\mathcal{P}}))$ with an MLP layer for the ease of contrastive loss computation. During inference, we don't apply the final MLP projection head and directly use $\mathbf{h}(\bar{\mathcal{P}})$ as the point cloud representation.

**Contrastive Learning Loss:** In this work, we use the NT-Xent loss proposed in [73]:

$$\mathbb{L}_{i,j} = -\log \frac{\exp(\mathbf{g}_i, \mathbf{g}_+)/\tau}{\sum_{k=1}^{K} \exp(\mathbf{g}_i, \mathbf{g}_k/\tau)} \tag{1}$$

where $\mathbf{g}_i$ is the anchor embedding, $\mathbf{g}_+$ is the positive sample embedding which has the same object ID as $\mathbf{g}_i$ while all other $\{\mathbf{g}_k\}_{k=1,2,\ldots,K,k\neq i}$ are negative samples in the batch that have different object IDs from $\mathbf{g}_i$.

**Training Details.** During each training iteration, we randomly load a batch of 64 data samples, along with one additional positive sample for each instance (i.e., a sample with the same object ID),

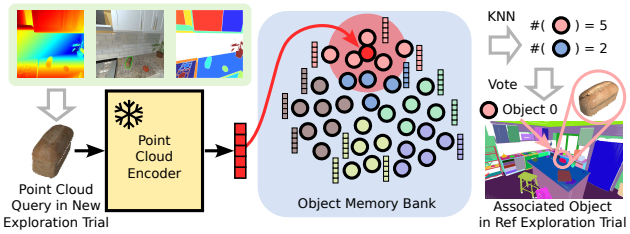


Figure 3: The process of querying the object memory bank with new exploration trial data. Given the RGB, depth, and instance masks in the new exploration trial, we extract the partial point cloud observation, encode the point cloud with the pre-trained per-scene point cloud encoder as in Sec. 4.3, and obtain a latent embedding which we use to look up the object memory bank (see Sec. 4.4) and find the K nearest neighboring embeddings. After counting the neighboring embeddings' instance labels, we can associate the partial observation to an instance in the reference trial of exploration.

resulting in 128 samples in total and ensuring at least 64 positive pairs. For each instance, all other instances in the batch, except its positive counterpart, are treated as negative samples. The model is trained with a learning rate of 0.003 for 300 epochs, using a 90/10 training-validation split. We evaluate the validation loss every 300 iterations and save the checkpoint with the lowest validation loss for use in experiments. In the dynamic edge convolutional network, we set $k = 10$ for the k-NN search.

### 4.4 Object Memory Bank Generation

After training the encoder for the scene with the reference trial of exploration data, you can encode all the partial point cloud observations $\{\bar{\mathcal{P}}_l\}_{l=1,2,...,L}$ in your dataset in Sec. 4.2 into the latent embeddings $\{\mathbf{h}(\bar{\mathcal{P}}_l)\}_{l=1,2,...,L}$ and associate those embeddings with their corresponding instance ID labels $\{y_l\}_{l=1,2,...,L}$, forming embedding-ID pairs $\{(\mathbf{h}_l^{ref}, y_l^{ref})\}_{l=1,2,...,L}$ (for simplicity, we write $\mathbf{h}(\bar{\mathcal{P}}_l)$ as $\mathbf{h}_l^{ref}$). Since the embeddings and labels are for the reference exploration trial, we add a superscript of $ref$ to their symbols. We treat these embeddings and labels as the object memory bank of the scene. Since the object embeddings are translation-invariant, we can easily look up the memory bank when new observations come after object location changes.

### 4.5 Object Association in Dynamic Environment

Now we switch to the new trial of exploration data, after a random number of object locations changes without adding new objects or removing old ones. Given the instance segmentation masks $\mathcal{M}_t = \{\mathbf{M}_{tk}\}_{k=1,2,...,K}$, depth image $\mathbf{D}_t$, and camera pose $\mathbf{T}_t$ of each frame $t$ in a new trial of exploration, we can build a new instance-level map as in Sec. 4.1. In addition, for each mask observation $\mathbf{M}_{tk}$, we can back-project them into the camera coordinate, center them at their means, and apply voxel-downsampling to a resolution of 0.01 meter to obtain partial point cloud observations $\bar{\mathcal{P}}_{tk}^{new}$ as in Sec. 4.2. We can use the trained encoder (see Sec. 4.3) to generate a latent embedding $\mathbf{h}(\bar{\mathcal{P}}_{tk}^{new})$ (we write it as $\mathbf{h}_{tk}^{new}$ for simplicity) for each partial point cloud observation $\bar{\mathcal{P}}_{tk}^{new}$. In the following step, we can find the distance of $\mathbf{h}_{tk}^{new}$ to all the embeddings $\{\mathbf{h}_l^{ref}\}_{l=1,2,...,L}$ in the object memory bank created in Sec. 4.4. We take the 10 nearest embeddings and store their object ID labels in the reference trial. These reference labels are associated with the global object ID in the new trial in a dictionary $\{new\ object\ ID:\ reference\ object\ IDs\}$. Whenever there are new observations for the same new object ID later, the reference object IDs list will be extended with new labels. During this process, we can use frequency to approximate the probability of association such that $P(f(j) = i|\mathbf{z}_{t=1:n}) = \frac{\#(y^{ref}=i)}{\sum_{m=1}^{M}\#(y^{ref}=m)}$ where $f(\cdot)$ is the mapping from new trial object ID $j$ to reference trial object ID $i$, $\mathbf{z}_t$ denotes the observations at timestep $t$, $\#(y^{ref} = i)$ represents the count of reference labels $i$ associated with new object $j$, and $M$ denotes the total number of objects in reference trial. To determine the final association for a new object ID, we simply retrieve the reference object ID with the highest probability.

# 5 Experimental Results

## 5.1 Object Association in Long-Term Dynamic Environments

**Experiment Setup:** To evaluate the association effectiveness of BYE, we collected 10 reference scenes (kitchens and bedrooms, more details can be found in Appendix Sec. C) and 10 corresponding change scenes in AI2THOR [18]. A robot was manually controlled to gather data, including RGB images, depth images, instance masks, and camera poses. For the change scenes, we initialized them to match the reference scenes, randomly moved some objects, and then collected exploration data after the changes. For each reference scene, we built an instance-level map, generated a dataset of partial instance point clouds with labels, trained the encoder, and used the checkpoint with the best validation loss to create an object memory bank. During the association process, we iterate through all observations of the new exploration, extracting masked point clouds, back-projecting, centering them, and generating latent embeddings using the same checkpoint. We then found the 10 nearest neighbors in the object memory bank for each new observation, keeping a count for each reference instance ID. This created a mapping from new instance IDs to reference IDs which evolves over time when more observations come. Finally, we used majority voting to assign the most likely reference instance ID to each new instance.

**Baselines:** The goal of the baseline methods is to integrate semantic-rich visual language features into the segment-level map, namely, associate one feature with each instance. We exploit the open-vocabulary mapping scheme introduced in HOV-SG [31] as follows. First, we build a voxel feature map as in VLMaps [24]. This requires a visual encoder that can generate dense pixel-level visual language features. Here we use LSeg [74], OVSeg [20], and DINOv2 [75] as the encoders in our baselines. Then we can back-project depth pixels into 3D space, find the voxel they belong to, and integrate the pixel features into the voxel with mean operation. At the same time, construct an instance-level map with the instance masks. Finally, after the voxel map and the instance map are completed, we need to assign one visual language feature to each instance. We first search for the nearest neighboring voxel for each point in an instance, then collect the voxel's associated feature. After collecting all features for all points in one instance, we apply DBSCAN [76] to the features and find the major cluster's mean. Then we find the feature with the closest distance to the cluster mean as the instance's feature. We build such instance-level feature maps for reference and new exploration trials for each scene. During association, we simply collect all instance features in the reference and the new scenes and compute the cosine similarity between each pair of objects. Finally, we use the Hungarian algorithm to determine the association.

**Metrics:** We use the association success rate as our metric, which can be defined as the number of correctly associated objects divided by the total number of objects. Since we are using KNN to retrieve nearby embeddings' instance IDs for each observation, after accumulating all observations' results for one instance in the new trial, there might be multiple IDs associated with it. We can sort these labels with their count in descending order. We use the object ID with the highest frequency as the predicted object in our method. In this experiment, we want to test the capability of associating all objects including the static objects. In total, there are 531 association objects, including 80 object categories among which 62 categories are movable. We further listed per-class success rates for movable categories with high frequencies.

**Results:** The results are shown in Table 1. The first two columns show the methods and the overall success rates. The rest of the columns show the per-class success rates for movable objects with high frequencies of occurrence. We observe that BYE outperforms all other baselines by a large margin with at least 5% improvement. We also plotted the success rate of each tested scene in AI2THOR as in Fig. 5. In most of the case, BYE achieve the highest recalls.

We further analyzed why BYE performs better than other foundation-model-based embeddings. Therefore, we also report the per-class success rate of several common categories in the dataset. In Fig. 4 in the Appendix, we further show the qualitative association results for some categories. When we first look at objects that are commonly occurring in various datasets like "Garbage Can",

"Bowl", "Stool", "Bed", "Desk", and "Laptop", fine-tuned VLMs like OVSeg and LSeg perform well in the association tasks, sometimes even better than BYE. The major reason behind this might be that the pre-training and fine-tuning processes allow the model to learn reliable and robust features for those categories. When there are no duplicate objects in the scene (like "Bed" and "Desk"), the semantic features are representative for those objects and therefore help with association. However, when we look at less common categories like "Cell Phone", "Credit Card", "Key Chain", and "Pan", foundation models struggle to correctly associate them due to their long-tailed characteristics in the dataset. The benefits of BYE emerge under these circumstances. Since BYE is trained only on the reference exploration data, which is not restricted by the data distribution that pre-trained models are accustomed to. BYE only focuses on the geometric and visual characteristics of the objects in the scenes and learns to differentiate them in the process of contrastive learning.

Table 1: OBJECT ASSOCIATION SUCCESS RATE IN LONG-TERM DYNAMIC ENVIRONMENTS

| Method | Success Rate (%) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Overall (531) | Garbage Can (10) | Bowl (9) | Cell Phone (6) | Credit Card (6) | Stool (6) | Bed (5) | Bread (5) | Desk (5) | Key Chain (5) | Alarm Clock (5) | Laptop (5) | Pan (5) |
| DINOv2 [75] | 68.2 | 70 | 55.6 | 16.7 | 50 | 83.3 | 80 | 20 | 80 | 60 | 20 | 60 | 50 |
| LSeg [74] | 69.9 | 88.9 | 55.6 | 16.7 | 0 | **100** | **100** | 40 | 40 | 40 | 40 | **100** | 20 |
| OVSeg [20] | 84.7 | **100** | **77.8** | 50 | 50 | **100** | **100** | 60 | 80 | 60 | 60 | **100** | 40 |
| BYE(ours) | **89.6** | 90 | 66.7 | **100** | **100** | **100** | **100** | **100** | **100** | **100** | **80** | 60 | **100** |

The overall and per-class object association success rates in 10 AI2THOR scenes. The number in the parenthesis denotes the total occurrence number of a specific kind of object.

### 5.2 Runtime Analysis

**Experiment Setup:** We use a machine with an AMD EPYC 7543 32-Core Processor CPU and an NVIDIA A40 GPU with 40 GB VRAM. We load the BYE (DGCNN) encoder in evaluation mode. We define a dataloader with batch size 1 and use only a single thread for dataloading. Then we iterate through a dataset of each scene and let the encoder generate latent embeddings in inference mode. We count the time for preprocessing (downsampling) and generating all embeddings with partial point cloud observations in the scene and divide the time with the total partial observations number to get the average runtime of the encoder.

**Results:** We observe in Table 2 that the average runtime of the encoder without considering the batch processing is around 20ms which is highly efficient to run at 50 Hz compared to other methods based on foundation models.

Table 2: RUNTIME ANALYSIS OF BYE

| Scene | Total Samples number | Total Runtime (s) | Average Runtime per Sample (ms) |
|---|---|---|---|
| Scene 1 | 5444 | 100.0 | 18.4 |
| Scene 2 | 7472 | 150.3 | 20.1 |
| Scene 3 | 5010 | 101.8 | 20.3 |
| Scene 4 | 5544 | 106.4 | 19.2 |
| Scene 5 | 6207 | 125.3 | 20.2 |
| Scene 6 | 3149 | 68.3 | 21.7 |
| Scene 7 | 1368 | 37.0 | 27.0 |
| Scene 8 | 2749 | 62.5 | 22.7 |
| Scene 9 | 1644 | 48.4 | 29.4 |
| Scene 10 | 2680 | 66.6 | 24.9 |
| Total | 41267 | 866.3 | 21.0 |

## 6 Conclusion

In this work, we investigate a novel way of tackling the object association problem in long-term dynamic scenes. By training a per-scene encoder without using category priors, shape priors, and large datasets, we manage to associate objects across dynamic scenes with high efficiency and accuracy. However, BYE has the limitations of struggling to detect newly inserted or removed objects. In the future, we will investigate how such an encoder can be combined with an open-vocabulary map, and integrated into a real-world robotic navigation system, achieving spatio-temporal open-vocabulary navigation or object search navigation.

## References


[1] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[3] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[4] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[5] M. Rünz and L. Agapito. Co-fusion: Real-time segmentation, tracking and fusion of multiple objects. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4471–4478. IEEE, 2017.

[6] B. Bescos, J. M. Fácil, J. Civera, and J. Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, 3(4):4076–4083, 2018.

[7] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei. Ds-slam: A semantic visual slam towards dynamic environments. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1168–1174. IEEE, 2018.

[8] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou. Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment. *Robotics and Autonomous Systems*, 117:1–16, 2019.

[9] M. Henein, J. Zhang, R. Mahony, and V. Ila. Dynamic slam: The need for speed. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2123–2129. IEEE, 2020.

[10] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.

[11] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger. Mid-fusion: Octree-based object-level multi-instance dynamic slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5231–5237. IEEE, 2019.

[12] A. Adam, T. Sattler, K. Karantzalos, and T. Pajdla. Objects can move: 3d change detection by geometric transformation consistency. In *European Conference on Computer Vision*, pages 108–124. Springer, 2022.

[13] A. Adam, K. Karantzalos, L. Grammatikopoulos, and T. Sattler. Has anything changed? 3d change detection by 2d segmentation masks. *arXiv preprint arXiv:2312.01148*, 2023.

[14] L. Zhu, S. Huang, K. Schindler, and I. Armeni. Living scenes: Multi-object relocalization and reconstruction in changing 3d environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28014–28024, 2024.

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

[16] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024.

[17] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[18] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[20] F. Liang, B. Wu, X. Dai, K. Li, Y. Zhao, H. Zhang, P. Zhang, P. Vajda, and D. Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7061–7070, 2023.

[21] G. Ghiasi, X. Gu, Y. Cui, and T.-Y. Lin. Scaling open-vocabulary image segmentation with image-level labels. In *European Conference on Computer Vision*, pages 540–557. Springer, 2022.

[22] D. Shah, B. Osiński, S. Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on Robot Learning*, pages 492–504. PMLR, 2023.

[23] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[24] C. Huang, O. Mees, A. Zeng, and W. Burgard. Visual language maps for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023.

[25] C. Huang, O. Mees, A. Zeng, and W. Burgard. Audio visual language maps for robot navigation. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*, Chiang Mai, Thailand, 2023.

[26] N. M. (Mahi)Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam. CLIP-Fields: Weakly Supervised Semantic Fields for Robotic Memory. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi:10.15607/RSS.2023.XIX.074.

[27] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo, A. Stone, and D. Kappler. Open-vocabulary queryable scene representations for real world planning. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11509–11522. IEEE, 2023.

[28] D. Maggio, Y. Chang, N. Hughes, M. Trang, D. Griffith, C. Dougherty, E. Cristofalo, L. Schmid, and L. Carlone. Clio: Real-time task-driven open-set 3d scene graphs. *arXiv preprint arXiv:2404.13696*, 2024.

[29] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, S. Li, G. Iyer, S. Saryazdi, N. Keetha, A. Tewari, et al. Conceptfusion: Open-set multimodal 3d mapping. 2023.

[30] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028. IEEE, 2024.

[31] A. Werby, C. Huang, M. Büchner, A. Valada, and W. Burgard. Hierarchical Open-Vocabulary 3D Scene Graphs for Language-Grounded Robot Navigation. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.077.

[32] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[33] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot language-guided manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[34] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on Robot Learning*, pages 284–301. PMLR, 2023.

[35] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023.

[36] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser. Openscene: 3d scene understanding with open vocabularies. 2023.

[37] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann. OpenMask3D: Open-Vocabulary 3D Instance Segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[38] F. Engelmann, F. Manhardt, M. Niemeyer, K. Tateno, M. Pollefeys, and F. Tombari. OpenNeRF: Open Set 3D Neural Scene Segmentation with Pixel-Wise Features and Rendered Novel Views. In *International Conference on Learning Representations*, 2024.

[39] C. M. Kim, M. Wu, J. Kerr, K. Goldberg, M. Tancik, and A. Kanazawa. Garfield: Group anything with radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21530–21539, 2024.

[40] X. Zuo, P. Samangouei, Y. Zhou, Y. Di, and M. Li. Fmgs: Foundation model embedded 3d gaussian splatting for holistic 3d scene understanding. *International Journal of Computer Vision*, pages 1–17, 2024.

[41] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024.

[42] Ó. M. Mozos, C. Stachniss, A. Rottmann, and W. Burgard. Using adaboost for place labeling and topological map building. In *Robotics Research: Results of the 12th International Symposium ISRR*, pages 453–472. Springer, 2007.

[43] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[44] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 international conference on 3D vision (3DV)*, pages 32–41. IEEE, 2018.

[45] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang. Flowfusion: Dynamic dense rgb-d slam based on optical flow. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 7322–7328. IEEE, 2020.

[46] R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

[47] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[48] Z. Yang, H. Yang, Z. Pan, X. Zhu, and L. Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023.

[49] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024.

[50] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024.

[51] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[52] M. Halber, Y. Shi, K. Xu, and T. Funkhouser. Rescan: Inductive instance segmentation for indoor rgbd scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2541–2550, 2019.

[53] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7658–7667, 2019.

[54] T. Sun, Y. Hao, S. Huang, S. Savarese, K. Schindler, M. Pollefeys, and I. Armeni. Nothing stands still: A spatiotemporal benchmark on 3d point cloud registration under large geometric and temporal change. *arXiv preprint arXiv:2311.09346*, 2023.

[55] J. Fu, Y. Du, K. Singh, J. B. Tenenbaum, and J. J. Leonard. Robust change detection based on neural descriptor fields. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2817–2824. IEEE, 2022.

[56] Y. Qiu, Y. Satoh, R. Suzuki, K. Iwata, and H. Kataoka. Indoor scene change captioning based on multimodality data. *Sensors*, 20(17):4761, 2020.

[57] S. Looper, J. Rodriguez-Puigvert, R. Siegwart, C. Cadena, and L. Schmid. 3d vsg: Long-term semantic scene change prediction through 3d variable scene graphs. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8179–8186. IEEE, 2023.

[58] J. Qian, V. Chatrath, J. Yang, J. Servos, A. P. Schoellig, and S. L. Waslander. Pocd: Probabilistic object-level change detection and volumetric mapping in semi-static scenes. *arXiv preprint arXiv:2205.01202*, 2022.

[59] J. Qian, V. Chatrath, J. Servos, A. Mavrinac, W. Burgard, S. L. Waslander, and A. P. Schoellig. Pov-slam: Probabilistic object-aware variational slam in semi-static environments. *arXiv preprint arXiv:2307.00488*, 2023.

[60] L. Schmid, M. Abate, Y. Chang, and L. Carlone. Khronos: A unified approach for spatio-temporal metric-semantic slam in dynamic environments. In *Proc. of Robotics: Science and Systems (RSS)*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.081.

[61] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[62] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[63] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[64] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.

[65] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.

[66] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12200–12209, 2021.

[67] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5939–5948, 2019.

[68] C.-W. Lin, T.-I. Chen, H.-Y. Lee, W.-C. Chen, and W. H. Hsu. Coarse-to-fine point cloud registration with se (3)-equivariant representations. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 2833–2840. IEEE, 2023.

[69] A. Misik, D. Salihu, X. Su, H. Brock, and E. Steinbach. Hegn: Hierarchical equivariant graph neural network for 9dof point cloud registration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6981–6988. IEEE, 2024.

[70] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022.

[71] J. Fu, Y. Du, K. Singh, J. B. Tenenbaum, and J. J. Leonard. Neuse: Neural se (3)-equivariant embedding for consistent spatial understanding with objects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[72] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[73] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

[74] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RriDjddCLN.

[75] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[76] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. Density-based spatial clustering of applications with noise. In *Int. Conf. knowledge discovery and data mining*, volume 240, 1996.

# APPENDIX

## A Per-Class Object Association Results Visualization

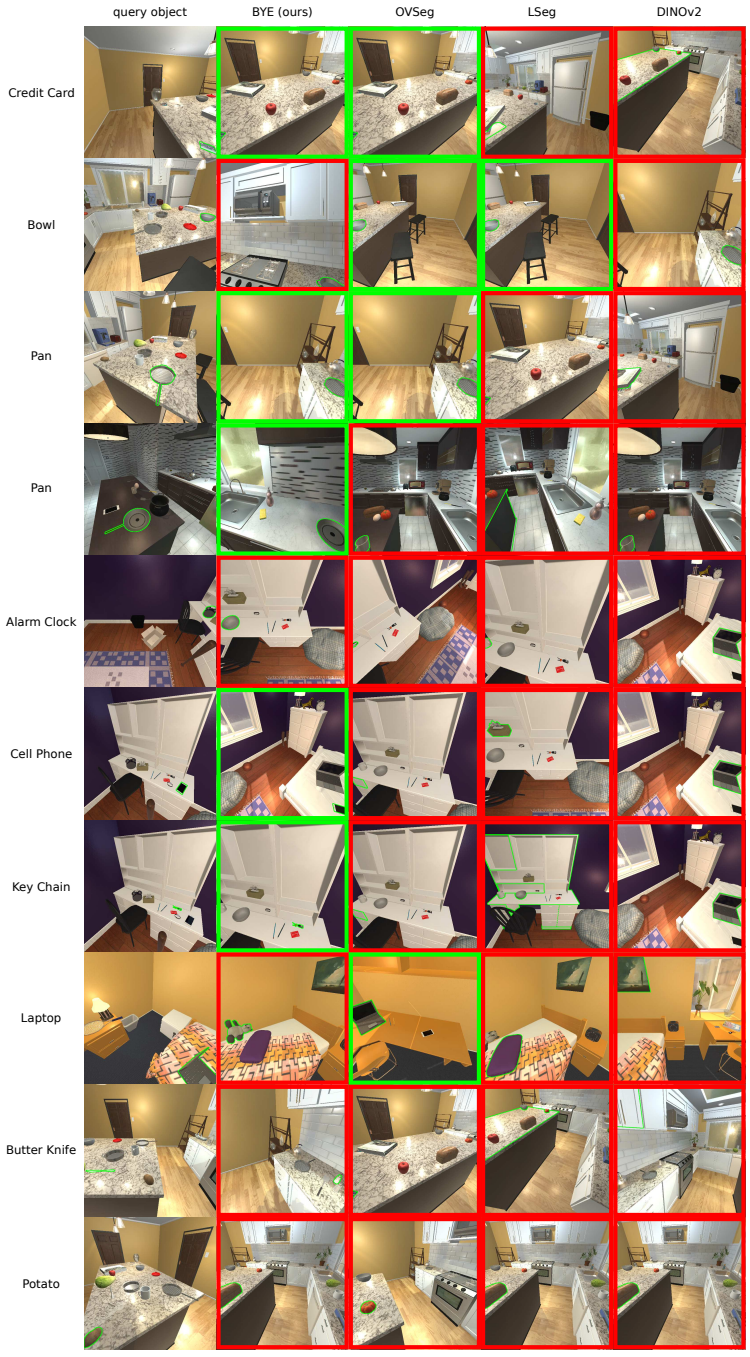


Figure 4: Per-class object association qualitative results in AI2THOR.

## B Per-Scene Association Success Rates

We show the per-scene association success rates in Fig. 5.


Figure 5: The association success rate in each tested scene in AI2THOR.

## C Evaluated Scenes in AI2THOR

We evaluated FloorPlan 1, 2, 4, 5, 6, which are kitchens, and FloorPlan 301-305, which are bedrooms.