Navigating to Objects Specified by Images

Jacob Krantz[1*] Theophile Gervet[2] Karmesh Yadav[3] Austin Wang[3]
Chris Paxton[3] Roozbeh Mottaghi[3,4] Dhruv Batra[3,5] Jitendra Malik[3,6]
Stefan Lee[1] Devendra Singh Chaplot[3]
[1]Oregon State [2]Carnegie Mellon [3]Meta AI [4]University of Washington [5]Georgia Tech [6]UC Berkeley
Project Page & Videos: jacobkrantz.github.io/modular_iin

Abstract

Images are a convenient way to specify which particular object instance an embodied agent should navigate to. Solving this task requires semantic visual reasoning and exploration of unknown environments. We present a system that can perform this task in both simulation and the real world. Our modular method solves sub-tasks of exploration, goal instance re-identification, goal localization, and local navigation. We re-identify the goal instance in egocentric vision using feature-matching and localize the goal instance by projecting matched features to a map. Each sub-task is solved using off-the-shelf components requiring zero fine-tuning. On the HM3D InstanceImageNav benchmark, this system outperforms a baseline end-to-end RL policy 7x and a state-of-the-art ImageNav model 2.3x (56% vs. 25% success). We deploy this system to a mobile robot platform and demonstrate effective real-world performance, achieving an 88% success rate across a home and an office environment.

1. Introduction

Consider instructing a last-mile delivery agent on where to deliver a package. Specifying a particular porch receptacle can be conveniently done via image, provided you are local to the environment prior to delivery and have foresight to capture the image. This example motivates the fundamental embodied skill we study in this paper: navigating to an object instance specified by an image. As depicted in Fig. 1 (Top), the agent is provided with egocentric vision and a goal image (in this case, a bed) and must navigate to that particular bed. This Image Goal Navigation task requires reasoning over the relation of objects in the scene (*e.g.*, disambiguating between instances of similar appearance) and exploring efficiently (*e.g.*, entering bedrooms while searching for the bed).

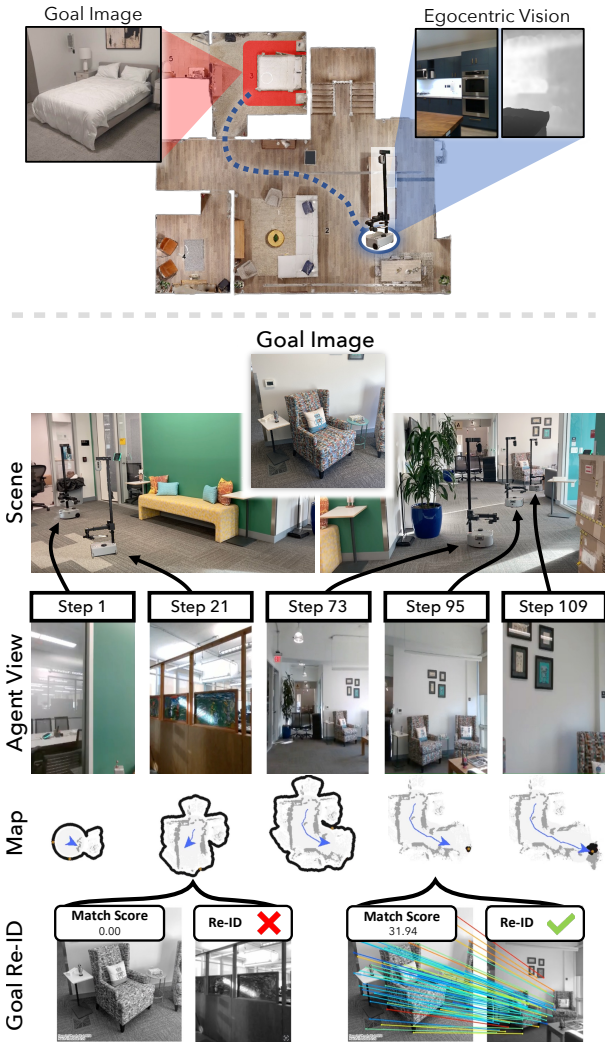


Figure 1: **Top: InstanceImageNav** tasks an agent with navigating to an object instance described by a goal image. **Bottom: Real-World Deployment.** Our method achieves leading performance in sim and transfers to reality. Here, we find a chair 10m away. Videos are on the project page.

*Work done while interning at Meta AI's FAIR Labs.
Correspondence: krantzja@oregonstate.edu

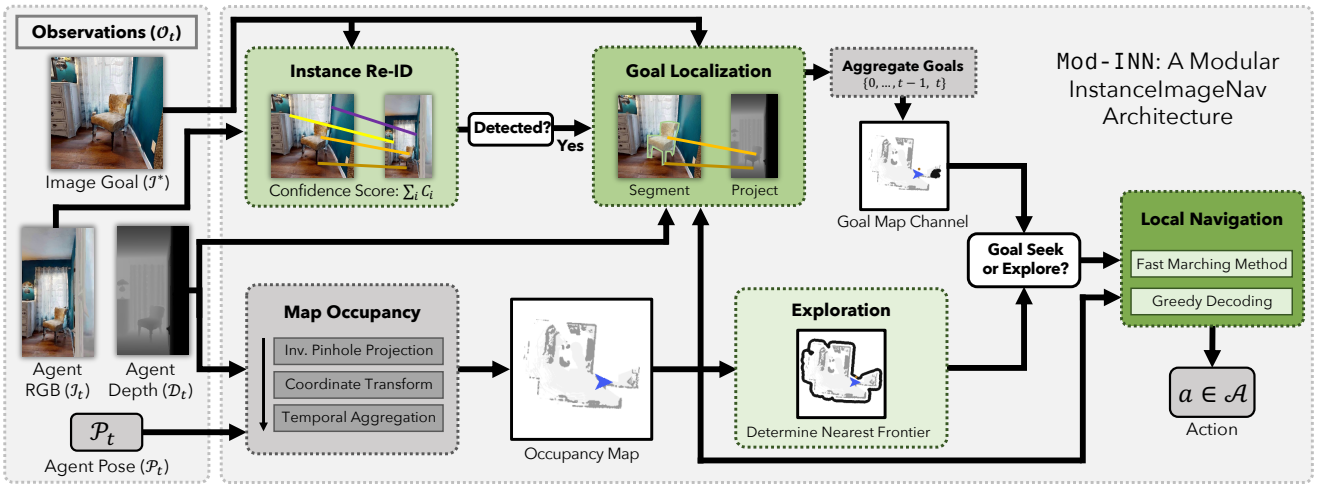


Figure 2: **Model Overview.** We instantiate *exploration* with frontier-based exploration, *instance re-identification* with feature matching, *goal localization* with masked feature projection, and *local navigation* with analytical planning. Sub-task modules are green with supporting components in gray.

In this paper we present a navigation system that can reliably perform this instance-based Image Goal Navigation task in the real-world. Specifically, we propose a modular framework consisting of Exploration, Instance Re-Identification, Goal Localization, and Local Navigation. We instantiate this framework using a simple combination of off-the-shelf components requiring zero fine-tuning. Depicted in Fig. 2, our system uses a frontier-based exploration policy, re-identifies goals with feature matching, localizes goals with projected feature matches, and path plans with an analytical planner. On the challenging Habitat-Matterport3D (HM3D) [32] InstanceImageNav benchmark [24], we achieve a success rate of 56% *vs*. 25% for the best baseline. We deploy our system on a mobile robot platform in two real-world environments where it achieves a success rate of 88% (*e.g.*, Fig. 1 Bottom).

Most prior work tackling Image Goal Navigation (ImageNav)[53, 10, 19, 29, 1, 28, 49] assumes that goal images were captured at random poses in the environment and always match the camera parameters of the agent. As argued in Krantz, *et al*. [24], this formulation may result in ambiguous image goals (*e.g.*, captures of nondescript walls) and is detached from potential user applications. To overcome these issues, the instance-based ImageNav task (InstanceImageNav) proposed by Krantz, *et al*. [24] has two key properties: (1) goal images depict an object instance, and (2) goal images are independent of agent embodiment. A baseline end-to-end reinforcement learning policy achieves just 8.3% success. Our system outperforms this 7-fold. To further compare to prior work, we evaluate a state-of-the-art ImageNav method [48] on InstanceImageNav. While it outperforms the baseline, our system outperforms it 2.3x.

Many prior navigation approaches, such as the two mentioned above, involve training sensors-to-action policies *end-to-end* using reinforcement or imitation learning. While end-to-end methods can be easy to implement and applicable to multiple tasks, they suffer from limitations of high sample complexity [24], overfitting [46], and poor sim-to-real transfer [18]. Alternatively, navigation can be decomposed into sub-tasks solved with more constrained skills [8, 9, 17, 18]. This *modular* paradigm provides benefits of increased sample efficiency and improved real-world execution [31, 18, 14]. Our method demonstrates these benefits with efficient sample complexity (zero fine-tuning) and effective real-world performance (Fig. 1 Bottom), all while achieving top performance on the simulation benchmark.

Altogether, we tackle the relevant and challenging Instance-specific Image Goal Navigation task. We propose a modular method that tops the HM3D InstanceImageNav benchmark and outperforms a state-of-the-art ImageNav model. We deploy our system on a mobile robot platform and demonstrate that it can operate reliably in indoor real-world environments.

2. Related Work

Image Goal Navigation. In embodied navigation, targets can be specified via coordinates [2] or through various forms of semantic description [6, 53, 11, 3, 25]. Image-goal navigation (ImageNav) is one form of the latter; agents navigate in response to a visual description provided by an image. ImageNav is commonly studied in previously-unseen environments [53] where goal images are sampled randomly throughout the scene [10]. While many works study indoor environments, some study outdoor [38, 39, 40]. Many approaches to solving ImageNav adopt deep reinforcement

learning (DRL) to learn end-to-end policies that map egocentric vision to action [53, 1, 28, 49]. However, skills relating to visual scene understanding, semantic exploration, and long-term memory tend to be difficult to learn end-to-end. Thus, these methods tend to adopt a combination of careful reward shaping [12], pre-training routines [49], and advanced memory modules [34, 29, 46]. In opposition to end-to-end DRL, some approaches carve out sub-tasks that can be learned in a supervised manner, such as graph prediction via topological SLAM [10], graph-based distance learning [19, 39], and camera pose estimation for last-mile navigation [44]. In this work we address the ImageNav task where goal images depict object instances (InstanceImageNav [24]). We study how far we can drive performance on this benchmark using a purely modular method with no fine-tuning and demonstrate superior performance to a state-of-the-art end-to-end policy.

Modular Methods for Semantic Navigation. Classical approaches to navigating previously-unseen environments involve building a geometric map and localizing the agent (SLAM [16]). Modular methods to semantic navigation decompose high-level tasks into components that can either leverage the classical navigation pipeline or be solved with modern vision systems, such as object detectors [20]. One related task is Object Goal Navigation (ObjectNav [6]), in which an agent is given an object category (*e.g.*, *potted plant*) and must navigate to any instance of that category. Chaplot *et al*. [9] decomposed the ObjectNav task to exploration, object detection, and local navigation. Expanding on this, CLIP on Wheels (CoW [17]) employed a decomposition of exploration and object localization to address an open-set object vocabulary. Modular methods are also promising for effective simulation-to-reality transfer (Sim2Real). Gervet *et al*. [18] performed Sim2Real transfer of modular and end-to-end ObjectNav systems, finding that modularity avoided the visual Sim2Real gap that degraded the end-to-end policy. In this work, we expand the capability of modular navigation agents to include image-specified navigation targets. We propose a factorization of the problem that can be solved via modular components and demonstrate its effectiveness in both simulation and reality.

Instance Re-Identification. Object instance re-identification (OIRe-ID) is the task of determining if a given image depicts the same object in an anchor image. OIRe-ID is commonly operationalized as image retrieval [5, 4]. Prior to the deep learning revolution, image retrieval and related visual recognition tasks were based primarily on local feature descriptors [26] such as SIFT [27] or HOG [13]. A foundational object retrieval method involved applying text retrieval methods to these local features, resulting in a bag-of-visual-words model (BoVW [41]). Models relying on local features are limited by the expressivity of feature representation. As such, some modern approaches update this stack with deep networks for description (*e.g.* SuperPoint [15]) and matching (*e.g.* SuperGlue [33]). Alternatively, re-ranking methods using transformers [43] and epipolar-guided transformers [7] have been proposed in the OIRe-ID space. In this work, we propose solving an embodied OIRe-ID problem where egomotion guides the acquisition of novel instance views. We show that our SuperGlue-based keypoint method not only enables re-identification but also informs localizing the instance.

3. InstanceImageNav Task Setup

We follow the task setup proposed by Krantz *et al*. [24]. The instance-specific image goal navigation task (InstanceImageNav) places an agent at a random pose in an unexplored indoor environment and tasks the agent with navigating to a particular object instance depicted as the primary subject of an RGB image. We match our task setup with the ImageNav track of the 2023 Habitat Navigation Challenge[1] [47] to enable both clear comparisons and a smooth transfer from simulation to reality.

Observation Space. At each time step t, the agent's observation $\mathcal{O}_t$ consists of the RGB goal image $\mathcal{I}^*$, an egocentric RGB image $\mathcal{I}_t$, an egocentric depth image $\mathcal{D}_t$, and the agent's pose $\mathcal{P}_t = (x, y, \theta)$ relative to the starting pose $\mathcal{P}_0 = (0, 0, 0)$. Collectively, $\mathcal{O}_t = (\mathcal{I}^*, \mathcal{I}_t, \mathcal{D}_t, \mathcal{P}_t)$. The camera capturing $\mathcal{I}^*$ is independent of the camera capturing $\mathcal{I}_t$. Camera parameters for $\mathcal{I}^*$ are sampled to reflect realistic deployments, both extrinsic (height, look-at-angle, distance-to-object) and intrinsic (field of view). See Krantz, *et al*. [24] for more details. $\mathcal{I}^*$ is a 512×512 RGB image. The agent's egocentric perception matches the Intel RealSense camera: both $\mathcal{I}_t$ and $\mathcal{D}_t$ are aligned to 640×360 with a $42°$ HFOV.

Action Space. We adopt a discrete action space $\mathcal{A} = \{\texttt{MOVE-FORWARD}, \texttt{TURN-LEFT}, \texttt{TURN-RIGHT}, \texttt{STOP}\}$ where forward movement translates the agent by 0.25m and turn commands rotate the agent by $30°$.

Success Criteria. An InstanceImageNav episode is successful if the agent issues the `STOP` action while within 1.0m Euclidean distance of the object instance depicted by $\mathcal{I}^*$. The target instance must also be oracle-viewable by any combination of turning the agent and looking up or down.

Embodiment. Our real-world execution is performed using Stretch by Hello Robot[2]. We model this embodiment in our simulation experiments with a rigid-body, zero-turn-radius cylinder of height 1.41m and radius 0.17m. The forward-facing RGBD camera is mounted at a height of 1.31m.

[1] aihabitat.org/challenge/2023
[2] hello-robot.com/stretch-2

4. Method

We propose factorizing the InstanceImageNav task into sub-tasks that can be individually addressed. Specifically, we carve out exploration, goal instance re-identification, goal localization, and local navigation to solve InstanceImageNav in aggregate. We describe these sub-tasks as follows.

Exploration. Finding an object instance in a previously-unknown environment requires exploration — both the location of the goal and the map of the environment are unknown. In InstanceImageNav, the goal is described by $\mathcal{I}^*$, where a successful navigation entails observing an $\mathcal{I}_t$ at some time t with a high semantic similarity to $\mathcal{I}^*$. Efficiently visiting locations in the environment that may afford this view, *i.e.*, maximizing *coverage* of the observable space, can lead to a successful navigation.

Goal Instance Re-Identification. If an exploration policy results in 100% coverage of the observable space, then there exists at least one time step t such that the associated image $\mathcal{I}_t$ depicts the goal instance described by $\mathcal{I}^*$. Goal instance re-identification is thus the binary classifier f_{ID} that answers *"is the object depicted in $\mathcal{I}^*$ visible in $\mathcal{I}_t$?"* Concretely,

$$\hat{y}_t = f_{ID}(\mathcal{I}^*, \mathcal{I}_t). \tag{1}$$

This task leverages foreground and background to re-identify an object from novel views and is studied extensively in the object instance re-identification (OIRe-ID) literature.

Goal Localization. Agents may be far from the goal instance when it is identified, so simply calling `STOP` is insufficient for success. Thus, it is essential to use the pairing between $\mathcal{I}^*$ and $\mathcal{I}_t$ to localize the goal. Goal localization, f_{GL}, maps the paired RGB images and egocentric depth to the position of the goal instance relative to the agent's current pose:

$$\left(\mathcal{P}_G^{(x,y,\cdot)} - \mathcal{P}_t^{(x,y,\theta)}\right) = f_{GL}(\mathcal{I}^*, \mathcal{I}_t, \mathcal{D}_t). \tag{2}$$

Local Navigation. Local Navigation is the foundation that enables both exploration and navigation to the goal instance. We consider a local navigation policy π that maps a relative polar coordinate goal (r, θ) to a sequence of actions $\{a_0, a_1, \ldots, a_n\} \in \mathcal{A}$. π can be conditioned on a map and/or egocentric vision.

4.1. Proposed Modules

We instantiate a system that operationalizes the above factorization: `Mod-IIN`. Specifically, we propose a purely modular method that can perform InstanceImageNav without *any* re-trained or fine-tuned components. This method is in stark contrast to the prevailing paradigm of learned end-to-end ImageNav policies, yet demonstrates compelling results in simulation and reality. We visualize this model in Fig. 2.

Exploration. We adopt a frontier-based exploration (FBE [51]) policy that operates on a top-down 2D map tracking occupancy, free-space, and frontiers, where frontiers delineate the boundary between explored and unexplored regions. This map is updated each time step using an inverse perspective projection of egocentric depth ($\mathcal{D}_t$) and pose ($\mathcal{P}_t$). FBE greedily selects the nearest frontier in the map to navigate to. Upon reaching that frontier, the process repeats with the selection of the next nearest frontier. This policy enables an efficient expansion of coverage with demonstrated effectiveness in both simulation and reality [18].

Goal Instance Re-Identification. We employ a keypoint-based re-identification method that performs binary classification conditioned on the goal image and egocentric image. First, we extract the pixel-wise (x, y) location of keypoints $K^* \in \mathbb{R}^{n\times 2}$ in the goal image and their associated vector descriptions $V^* \in \mathbb{R}^{n\times 256}$. We extract these features using SuperPoint [15], a single-pass convolutional neural network trained using homographic adaptation, a self-supervised consistency method. We repeat this for the egocentric image, resulting in $K_t \in \mathbb{R}^{m\times 2}$ and $V_t \in \mathbb{R}^{m\times 256}$. Concretely:

$$(K^*, V^*) = \text{SuperPoint}(\mathcal{I}^*) \tag{3}$$

$$(K_t, V_t) = \text{SuperPoint}(\mathcal{I}_t) \tag{4}$$

We then compute correspondences between (K^*, V^*) and (K_t, V_t), resulting in a matched subset $(\hat{K}^* \in \mathbb{R}^{w\times 2}, \hat{K}_t \in \mathbb{R}^{w\times 2})$ such that the i^{th} keypoint of $\hat{K}^*$ corresponds to the i^{th} keypoint of $\hat{K}_t$ with $w \leq \min(n, m)$. Each keypoint pair has a match confidence score $0 \leq C \in \mathbb{R}^w \leq 1$. We use SuperGlue [33], a graph neural network (GNN) that optimizes a partial match assignment via optimal transport:

$$(\hat{K}^*, \hat{K}_t), C = \text{SuperGlue}((K^*, V^*), (K_t, V_t)) \tag{5}$$

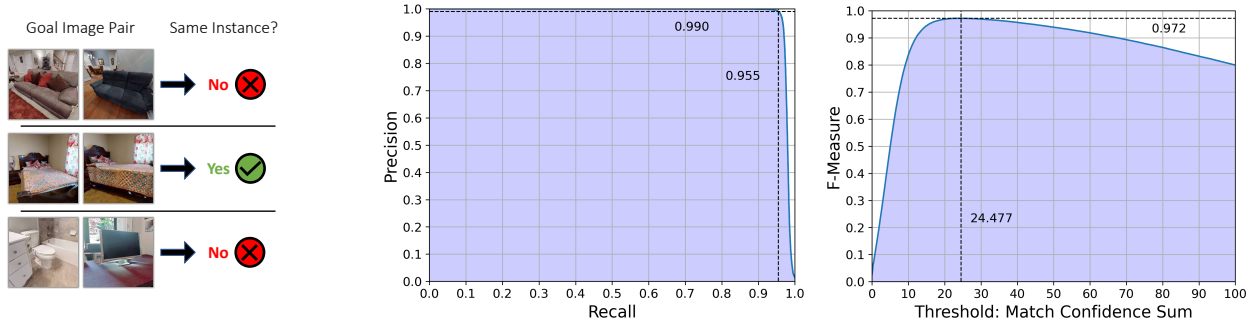
We can then turn this output into a binary classifier by thresholding the sum of the confidence scores:

$$\text{sum}(C) \geq \tau \tag{6}$$

where a positive truth value indicates re-identification of the goal instance and τ is a chosen threshold. This classifier is evaluated on $(\mathcal{I}^*, \mathcal{I}_t)$ at each step t. For both feature extraction (SuperPoint) and feature matching (SuperGlue), we use off-the-shelf models with zero downstream fine-tuning.

Determining a Detection Threshold τ. If τ is too low, egocentric images that do not observe the goal will be incorrectly passed on to goal localization (too many false positives). If τ is too high, steps during exploration that observe the goal will be missed (too many false negatives). To strike a balance between these error modes, we collect a dataset of pairs of goal images and pick the threshold that maximizes our classifier's F-measure. This method is as follows.

We sample a set of object instances that are represented in the training split of the InstanceImageNav episode dataset. Each object has between 1 and 50 goal images that depict it. We extract 3% of the object instances and their associated



(a) Classification Dataset (b) Precision-Recall Curve (c) F-Measure Curve

Figure 3: To determine an instance re-identification (Re-ID) threshold τ, we collect a dataset of goal image pairs (a). Our Re-ID method computes the sum of feature matching scores for each image pair which can then be used to compute a PR curve (b) and an f-measure curve (c). We select the τ with maximal f-measure ($\tau = 24.5$).

goal images, resulting in 121 objects and 2270 goal images (an average of 18.8 images per object). We use this data to construct a pairwise dataset where the input consists of two images $(\mathcal{I}_a, \mathcal{I}_b)$ and the output y consists of whether those images observe the same object: $((\mathcal{I}_a, \mathcal{I}_b), y) \in D$. We then run our classifier on all image pairs to produce a set of confidence scores. This enables us to compute precision-recall (PR) and F-measure curves for our classifier (Fig. 3). In the end, we select the threshold that maximizes F-measure (the harmonic mean of precision and recall). We verify that this produces an empirically optimal threshold in Sup. A.

Goal Localization. Upon observing the goal instance, we need to localize it in the world (Eq. 2). From the previous re-identification step, we have correspondences $(\hat{K}^*, \hat{K}_t)$ between the goal image and the egocentric image. Using the depth map $\mathcal{D}_t$ aligned to our egocentric image, we can project these points into the world via an inverse perspective projection. However, not all matched keypoints lie on the goal instance we seek to navigate to; visual features associated with the background and adjacent objects can cause the agent to stop at the wrong location. The question becomes, *"which matched keypoints should we project as goal targets?"*.

We perform instance segmentation of the goal image to determine a mask of the goal instance — keypoints inside the mask can be mapped to the egocentric frame and then be projected. We perform instance segmentation using an off-the-shelf pre-trained network (Detic [52]). We select Detic as it is a high-performance detector that allows for an open vocabulary by encoding concepts through CLIP [30]. In using Detic, our method is not constrained to fixed object categories and can be extended to image goals depicting arbitrary objects. To determine a goal instance mask, we select the mask containing the image center point with highest confidence.

All keypoints in $\hat{K}^*$ that lie within this mask are mapped to their corresponding keypoint in $\hat{K}_t$ and projected to the world. The point cloud of goal points is coalesced along the height dimension and voxelized into a 2D map channel. This channel is then concatenated with the map constructed during exploration and treated as a navigation target.

Local Navigation. The local navigator must navigate to frontier points (exploration) and goal points (goal localization). Both can be addressed as follows; given a partial occupancy map and a set of goal points, plan a path in the agent's action space $\mathcal{A}$ that conveys the agent to the nearest reachable point. We solve this using an incremental path planner based on the fast marching method (FMM [37]) as proposed by [8] and used in subsequent works [9, 19, 23].

5. Experiments

We evaluate our model in simulation and reality. In simulation, we compare to prior art and alternate sub-task modules (Sec. 5.2). We analyze failure modes (Sec. 5.3) and discuss a qualitative example (Sec. 5.4). Finally, we demonstrate successful Sim2Real transfer (Sec. 5.5).

5.1. Experimental Setup

Simulation. Our simulated experiments follow the task definition and dataset proposed for the ImageNav track of the 2023 Habitat Navigation Challenge [24]. We build and evaluate our agent on top of the Habitat Simulator [35, 42]. The episode dataset follows the generation procedure proposed by Krantz, *et al*. [24] for InstanceImageNav with the additional constraint that multi-floor navigation is not required of any episode. Scenes supporting this dataset come from the Habitat-Matterport3D Dataset [32] with semantic annotations (HM3D-SEM [50]). The 216 scenes in HM3D-SEM are split Train/Val/Test on 145/36/35 and InstanceImageNav episodes are split Train/Val/Test-Standard/Test-Challenge on 7,056K/1K/1K/1K. The object instances depicted by the goal images have a category belonging to one of the following: {

#	Model	Validation		
		NE ↓	SR ↑	SPL ↑
1	IIN RL Baseline	6.3	0.083	0.035
2	OVRL-v2-ImageNav	6.9	0.006	0.002
3	OVRL-v2-IIN	5.0	0.248	0.118
4	Mod-IIN (Ours)	**3.1**	**0.561**	**0.233**

Table 1: We compare our Modular InstanceImageNav method (Mod-IIN) against prior methods on InstanceImageNav. Mod-IIN outperforms the baseline model (row 1) with a 6.8x increase in SR and outperforms a state-of-the-art ImageNav model (OVRL-v2, row 3) by 2.3x.

chair, couch, plant, bed, toilet, television }. We use the Val split which is composed of 795 unique object instances.

Metrics. We evaluate models on success and efficiency. Particularly, we report success rate (SR), success weighted by inverse path length (SPL), navigation error (NE), and, in some analyses, maximum steps taken (Max-ST). Success is true if upon calling STOP, the agent is within 1.0m Euclidean distance of the goal and the object is oracle-visible by turning or looking up/down. SPL is an efficiency measure defined in [2] and modified for goal viewpoints in [6]. NE is the geodesic distance between the agent's stopping location and the nearest goal viewpoint. Max-ST is the percentage of episodes that use the full step budget (1000).

Baselines. We compare our agent to the following models:

- IIN RL Baseline: The InstanceImageNav (IIN) baseline model [24] is an end-to-end sensors-to-action recurrent policy consisting of unimodal encoders for the goal image, egocentric image, egocentric depth, and pose. This model was trained from scratch using reinforcement learning for 3.5 billion steps of experience using proximal policy optimization (PPO [36]) with variable experience rollout (VER [45]).
- OVRL-v2: Offline Visual Representation learning V2 (OVRL-v2 [48]) is a model-free, end-to-end semantic navigation policy that employs a ViT+LSTM architecture and self-supervised visual pre-training. At the time of writing, OVRL-v2 achieves state-of-the-art performance on a prior ImageNav benchmark and near state-of-the-art on an ObjectNav benchmark. We evaluate the pre-trained model zero-shot on InstanceImageNav (OVRL-v2-ImageNav). To make a fair comparison, we then take the pre-trained model and fine-tune it for InstanceImageNav (OVRL-v2-IIN) following the same training routine and parameters used in [48] for their ImageNav experiments.

Ablations: Goal Instance Re-Identification. We compare our goal instance re-identification method against baseline approaches. We consider:

- *Keypoint-Conf:* This is our primary method as described in Sec. 4.1. In summary, keypoints are extracted from both goal and egocentric images and matched. This is converted to binary classification by thresholding the sum of correspondence confidence scores.
- *Keypoint-Match:* This method is the same as above but with a different classification strategy; the number of matched keypoint pairs are thresholded instead of the confidence sum. This allows us to test if match confidence affords additional discriminative value.
- *ResNet:* This method ablates keypoints entirely by encoding both the goal and the egocentric images with a ResNet-50 [21] pre-trained on ImageNet. We compute and threshold the cosine similarity between the resulting feature vectors such that a cosine similarity above τ implies re-identification of the goal instance.
- *CLIP:* This method is the same as ResNet but encodes images using a model trained contrastively: CLIP [30].
- *Oracle:* This method provides an upper bound to the instance re-identification sub-task by querying the simulator for an oracle instance mask in the agent's egocentric frame. A positive detection is made if any pixel in the mask matches the ID of the goal object.

All methods above (except for *Oracle*) threshold some scalar value to perform detection. Each threshold is determined via the maximal F-measure method presented in Sec. 4.1. We include PR and F-measure curves for each in Sup. B.

Ablations: Goal Localization. We compare our goal localization method against baseline approaches. We consider:

- *Detic-Projected:* This is our primary method as described in Sec. 4.1. In summary, matched goal image keypoints that lie within the goal instance mask (computed by Detic) are projected to a goal map channel.
- *Crop-Projected:* This method ablates instance segmentation via Detic. Matched keypoints that lie within a static central crop of the goal image are projected. We provide further details of this method in Sup. C.
- *Detic-ObjectNav:* This method ablates keypoints. Detic infers the object category of the goal instance. Upon positive Re-ID, Detic identifies all pixels in the egocentric image that belong to the inferred category. These pixels are projected and the agent navigates to the closest object of that class. This method takes the semantic segmentation form of modular ObjectNav [9, 18].
- *Oracle:* This method provides an upper bound to the goal localization sub-task. Upon positive detection, all

#	Model Variation: Instance Re-ID	Goal Localization	Validation: NE ↓	Max-ST ↓	SR ↑	SPL ↑
1	Keypoint-Conf	Detic-Projected	**3.096**	**0.310**	**0.561**	**0.233**
2	Keypoint-Match	Detic-Projected	3.261	0.339	0.539	0.221
3	Keypoint-Conf	Crop-Projected	3.128	0.301	0.523	0.224
4	Keypoint-Conf	Detic-ObjectNav	3.235	0.327	0.488	0.205
5	ResNet	Detic-ObjectNav	6.264	0.857	0.138	0.044
6	CLIP	Detic-ObjectNav	6.570	0.917	0.097	0.029
7	Oracle	Oracle	1.300	0.089	0.845	0.453
8	Oracle	Detic-Projected	2.551	0.116	0.498	0.250
9	Keypoint-Conf	Oracle	2.914	0.348	0.647	0.266

Table 2: Our Modular InstanceImageNav method (Mod-IIN: row 1) with variations to instance re-identification and goal localization.

egocentric pixels that observe the goal object instance (according to an oracle instance mask) are projected.

5.2. Simulation Results

We center our results discussion on key observations.

Mod-IIN Outperforms Prior Art. Our method outperforms the baseline InstanceImageNav method (Tab. 1 row 1 *vs*. 4) with a nearly 7-fold increase in success (0.561 SR *vs*. 0.083 SR). Mod-IIN requires zero fine-tuning, while the end-to-end IIN RL Baseline was trained for 3.5 billion steps of experience distributed across 64 GPUs.

Evaluating OVRL-v2 zero-shot on InstanceImageNav (Tab. 1 row 2) results in very low performance (0.006 SR). Contributing factors include a different scene dataset (Gibson *vs*. HM3D), a change in embodiment (LocoBot *vs*. Stretch), and a different goal destination (image source *vs*. image subject). OVRL-v2 trained for InstanceImageNav (OVRL-v2-IIN) performs better with a 0.248 SR (Tab. 1 row 3), yet despite its visual pre-training, demonstrates significant overfitting to the Train split (0.850 SR). Our method performs 2.3x better in Val (Tab. 1 row 4 *vs*. 3).

Keypoint Confidence Is More Discriminative Than Match Count. Thresholding the sum of match confidence scores (*Keypoint-Conf*) leads to a downstream improvement of 0.022 SR over thresholding the count of matched keypoints (*Keypoint-Match*) (Tab. 2 row 1 *vs*. 2). This result is supported by comparing the maximal F-measure of the two classifiers on the pair-wise goal image dataset; *Keypoint-Match* produces 0.972 while *Keypoint-Conf* produces 0.962.

Goal Localization Benefits From Instance Segmentation. Replacing the instance mask (*Detic-Projected*) with a central crop (*Crop-Projected*) reduces success by 0.038 (Tab. 2 row 1 *vs*. 3). Beyond worse performance, the central crop method exploits a size bias in the dataset that may not generalize.

Keypoint-Based Localization Outperforms Class-Based Localization. Both methods of matched keypoint projection, *Detic-Projected* and *Crop-Projected*, outperform class-based localization (*Detic-ObjectNav*) (Tab. 2 rows 1,3 *vs*. 4). Semantic segmentation methods common for goal localization in ObjectNav cannot be trivially applied to InstanceImageNav. This validates that navigating to object instances in indoor environments fundamentally requires the ability to disambiguate between object instances of the same class.

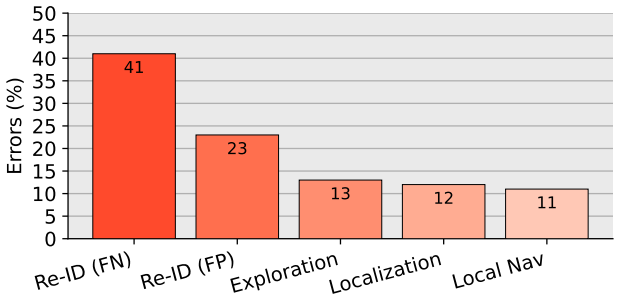


Figure 4: A distribution of Mod-IIN failure modes.

Off-The-Shelf Image Encoders Fail to Re-ID Instances. Both *ResNet* and *CLIP* fail to discriminate views of object instances. This is evident in Tab. 2 rows 5&6 where Max-ST is 86% and 92%, leading to success rates of 14% and 10%, respectively. We found reducing the *ResNet* detection threshold failed to improve performance. We suspect that the low performance of these methods relates to a poor alignment between the training objective and the OIRe-ID application.

5.3. Failure Analysis.

In Tab. 2 row 7, we evaluate our agent with perfect instance Re-ID and goal localization, meaning that any failures can be attributed to exploration via FBE with local navigation. The resulting performance of 0.845 SR acts as an upper bound when addressing instance Re-ID and goal localization within our system. In row 8, we use oracle instance Re-ID with predicted localization. Performance does not improve over our complete model. We suspect this to be caused by the shared reliance on keypoint matching between our detection and localization systems; assuming detection with low match confidence leads to poor localization. Finally, row 9 demonstrates a 0.086 SR gap between our goal localization method and perfect goal localization (row 9 *vs*. 1).

We present qualitative analysis of Mod-IIN failure modes in Fig. 4. For a random subset of 100 failed episodes, we label the cause of failure as one of the following:

Instance Re-ID: False Negative (41%). Most commonly, the agent observes and fails to detect the goal. Improved Re-ID methods may mitigate this, but not all novel instance views are equally challenging; Re-ID would be simplified if exploration produces views more similar to the goal image.

Instance Re-ID: False Positive (23%). Incorrect detections

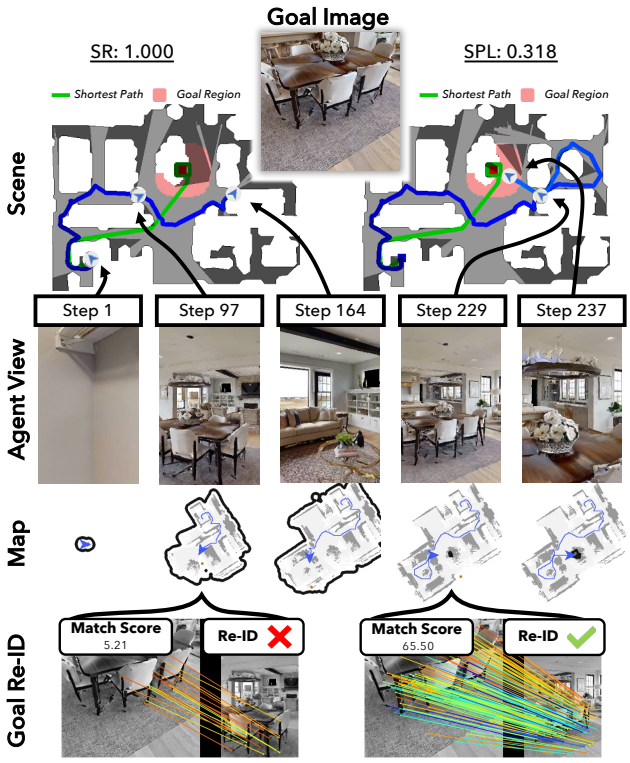


Figure 5: Qualitative example of our `Mod-IIN` agent performing the InstanceImageNav task in the Habitat Simulator.

often come from visual features correctly matched to the goal image background. This failure mode could be mitigated with additional background *vs*. foreground reasoning.

Exploration Error (13%)**.** Exploration fails to produce a view of the goal instance within the allotted time budget.

Localization Error (12%)**.** Goal localization errors result from projecting points not belonging to the goal instance, caused either by incorrect correspondences or goal masking.

Local Navigation Error (11%)**.** Local navigation occasionally fails to plan a path to the goal.

5.4. Qualitative Example

We present a qualitative example of our agent performing an episode in simulation in Fig. 5. During Steps 1, 97, and 164, the agent is exploring the environment. A detection is made at Step 229, and keypoints are projected as a goal target. Upon reaching the goal at Step 237 (a dining chair), the agent calls `STOP`. This episode demonstrates a long exploration horizon with a challenging goal instance; multiple identical objects exist in the scene, yet the agent navigated to the correct one. This is due to our instance Re-ID method matching all keypoints in the frame and our localization method only projecting those depicting the goal. We provide videos of our agent in simulation on our project page.

Figure 6: We deploy `Mod-INN` to a Hello Robot Stretch and evaluate in both an office (Env A) and an apartment (Env B). Our system achieves an 88% success rate across 8 episodes.

5.5. Real-World Deployment

We evaluate our method in the real world by deploying to a Hello Robot Stretch [22]. We evaluate in two different environments for a total of eight episodes using five unique image goals (Fig. 6). Environment A (Env A) is a furnished office space with a hallway and lounge. Multiple potted plants, couches, chairs and other confounding objects exist in the scene. We experiment with image goals depicting a couch, chair, and potted plant. Environment B (Env B) as depicted in Fig. 1 is a furnished apartment with a kitchen, living area, bedroom, bathroom, and office. We experiment with image goals depicting a bed and a potted plant. Image goals are captured using a mobile phone. All episodes start without viewing the goal such that exploration is required. The shortest-path geodesic distance of episodes range from 3-10m. Altogether, our agent is successful in $7/8 = 88\%$ of episodes. Videos of each episode are on our project page.

6. Discussion

We decompose InstanceImageNav into exploration, goal re-identification, goal localization, and local navigation. We craft a system within this framework that performs InstanceImageNav with zero fine-tuning. Our experiments show that this system outperforms existing state-of-the-art end-to-end learned policies and transfers to real-world execution.

Limitations and Impact. One limitation of our system is that detection and localization are memory-less; performing sequential tasks in persistent environments would benefit from grounding goal images in compressed memory. Our method can serve as a strong and robust baseline for evaluating trained navigation policies and our framework can serve as a catalyst for developing modular semantic navigators.

Acknowledgements The Oregon State effort is supported in part by the DARPA Machine Common Sense program. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the US Government or any sponsor.

References

[1] Ziad Al-Halah, Santhosh Kumar Ramakrishnan, and Kristen Grauman. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[2] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 2, 6

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[4] Vaibhav Bansal, Gian Luca Foresti, and Niki Martinel. Where did i see it? object instance re-identification with attention. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[5] Vaibhav Bansal, Stuart James, and Alessio Del Bue. re-obj: Jointly learning the foreground and background for object instance re-identification. In *International Conference on Image Analysis and Processing (ICIAP)*, 2019. 3

[6] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020. 2, 3, 6

[7] Yash Bhalgat, Joao F Henriques, and Andrew Zisserman. A light touch approach to teaching transformers multi-view geometry. *arXiv preprint arXiv:2211.15107*, 2022. 3

[8] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations (ICLR)*, 2020. 2, 5

[9] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 5, 6

[10] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[11] Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. Soundspaces: Audio-visual navigation in 3d environments. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[12] Yunho Choi and Songhwai Oh. Image-goal navigation via keypoint-based reinforcement learning. In *International Conference on Ubiquitous Robots (UR)*, 2021. 3

[13] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR)*, 2005. 3

[14] Matt Deitke, Dhruv Batra, Yonatan Bisk, Tommaso Campari, Angel X Chang, Devendra Singh Chaplot, Changan Chen, Claudia Pérez D'Arpino, Kiana Ehsani, Ali Farhadi, et al. Retrospectives on the embodied ai workshop. *arXiv preprint arXiv:2210.06849*, 2022. 2

[15] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Computer Vision and Pattern Recognition workshops (CVPRW)*, 2018. 3, 4

[16] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 2006. 3

[17] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *arXiv preprint arXiv:2203.10421*, 2022. 2, 3

[18] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *arXiv preprint arXiv:2212.00922*, 2022. 2, 3, 4, 6

[19] Meera Hahn, Devendra Singh Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No rl, no simulation: Learning to navigate without navigating. In *Neural Information Processing Systems (NeurIPS*, 2021. 2, 3, 5

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *International Conference on Computer Vision (ICCV)*, 2017. 3

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[22] Charles C Kemp, Aaron Edsinger, Henry M Clever, and Blaine Matulevich. The design of stretch: A compact, lightweight mobile manipulator for indoor human environments. In *International Conference on Robotics and Automation (ICRA)*, 2022. 8

[23] Jacob Krantz and Stefan Lee. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *European Conference on Computer Vision (ECCV)*, 2022. 5

[24] Jacob Krantz, Stefan Lee, Jitendra Malik, Dhruv Batra, and Devendra Singh Chaplot. Instance-specific image goal navigation: Training embodied agents to find object instances. *arXiv preprint arXiv:2211.15876*, 2022. 2, 3, 5, 6

[25] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[26] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 2020. 3

[27] David G Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, 1999. 3

[28] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *arXiv preprint arXiv:2206.12403*, 2022. 2, 3

[29] Lina Mezghan, Sainbayar Sukhbaatar, Thibaut Lavril, Oleksandr Maksymets, Dhruv Batra, Piotr Bojanowski, and Karteek Alahari. Memory-augmented reinforcement learning for image-goal navigation. In *International Conference on Intelligent Robots and Systems (IROS)*, 2022. 2, 3

[30] 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 (ICML)*, 2021. 5, 6

[31] Santhosh K. Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[32] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021. 2, 5

[33] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 4

[34] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *International Conference on Learning Representations (ICLR)*, 2018. 3

[35] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *International Conference on Computer Vision (ICCV)*, 2019. 5

[36] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6

[37] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences (PNAS)*, 1996. 5

[38] Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Ving: Learning open-world navigation with visual goals. In *International Conference on Robotics and Automation (ICRA)*, 2021. 2

[39] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid Exploration for Open-World Navigation with Latent Goal Models. In *Conference on Robot Learning (CoRL)*, 2021. 2, 3

[40] Dhruv Shah and Sergey Levine. ViKiNG: Vision-Based Kilometer-Scale Navigation with Geographic Hints. In *Robotics: Science and Systems (RSS)*, 2022. 2

[41] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision (ICCV)*, 2003. 3

[42] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Neural Information Processing Systems (NeurIPS)*, 2021. 5

[43] Fuwen Tan, Jiangbo Yuan, and Vicente Ordonez. Instance-level image retrieval using reranking transformers. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[44] Justin Wasserman, Karmesh Yadav, Girish Chowdhary, Abhinav Gupta, and Unnat Jain. Last-mile embodied visual navigation. In *Conference on Robot Learning (CoRL)*, 2022. 3

[45] Erik Wijmans, Irfan Essa, and Dhruv Batra. Ver: Scaling on-policy rl leads to the emergence of navigation in embodied rearrangement. In *Neural Information Processing Systems (NeurIPS)*, 2022. 6

[46] Yi Wu, Yuxin Wu, Aviv Tamar, Stuart Russell, Georgia Gkioxari, and Yuandong Tian. Bayesian relational memory for semantic visual navigation. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[47] Karmesh Yadav, Jacob Krantz, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Jimmy Yang, Austin Wang, John Turner, Aaron Gokaslan, Oleksandr Maksymets, Angel X Chang, Manolis Savva, Devendra Singh Chaplot, Alexander Clegg, and Dhruv Batra. Habitat challenge 2023. https://aihabitat.org/challenge/2023/, 2023. 3

[48] Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. Ovrl-v2: A simple state-of-art baseline for imagenav and objectnav. *arXiv preprint arXiv:2303.07798*, 2023. 2, 6

[49] Karmesh Yadav, Ram Ramrakhya, Arjun Majumdar, Vincent-Pierre Berges, Sachit Kuhar, Dhruv Batra, Alexei Baevski, and Oleksandr Maksymets. Offline visual representation learning for embodied navigation. *arXiv preprint arXiv:2204.13226*, 2022. 2, 3

[50] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, et al. Habitat-matterport 3d semantics dataset. *arXiv preprint arXiv:2210.05633*, 2022. 5

[51] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 1997. 4

[52] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision (ECCV)*, 2022. 5

[53] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017. 2, 3