What Does Really Matter in Image Goal Navigation?

Gianluca Monaci, Philippe Weinzaepfel, Christian Wolf
NAVER LABS Europe, Meylan, France
`firstname.lastname@naverlabs.com`

Abstract

Image goal navigation requires two different skills: firstly, core navigation skills, including the detection of free space and obstacles, and taking decisions based on an internal representation; and secondly, computing directional information by comparing visual observations to the goal image. Current state-of-the-art methods either rely on dedicated image-matching, or pre-training of computer vision modules on relative pose estimation. In this paper, we study whether this task can be efficiently solved with end-to-end training of full agents with RL, as has been claimed by recent work. A positive answer would have impact beyond Embodied AI and allow training of relative pose estimation from reward for navigation alone. In this large experimental study we investigate the effect of architectural choices like late fusion, channel stacking, space-to-depth projections and cross-attention, and their role in the emergence of relative pose estimators from navigation training. We show that the success of recent methods is influenced up to a certain extent by simulator settings, leading to shortcuts in simulation. However, we also show that these capabilities can be transferred to more realistic setting, up to some extent. We also find evidence for correlations between navigation performance and probed (emerging) relative pose estimation performance, an important sub skill.

1. Introduction

The interplay between perception and action in modern AI-based robotics is a well-studied topic covering contributions from multiple fields, including computer vision, sequential decision making, and optionally some form of mapping and planning. Independent of the specific robotics task at hand, there is an ongoing debate on whether these components should be trained end-to-end, *e.g.* with RL [8, 76], typically for higher performance, or whether modular techniques could provide higher robustness, *e.g.* [15, 37].

In the context of navigation, the *ImageNav* task adds an additional complexity, as perception not only involves the currently observed image, but also the goal image, requiring
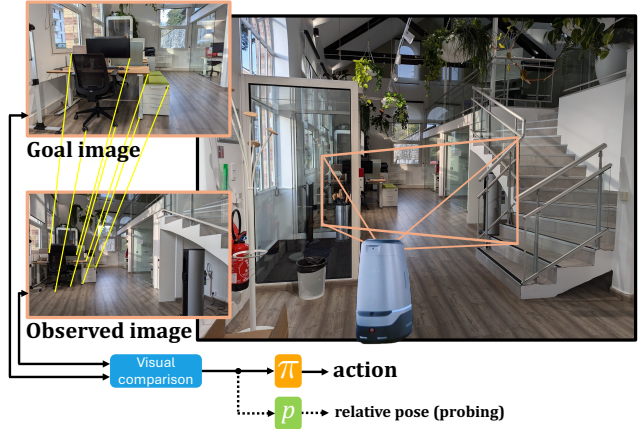

Figure 1. **Image goal navigation**, or "*ImageNav*", requires general navigation skills, but also in particular the extraction of directional information towards the goal. We analyze which architecture design choices influence these capabilities, and to what degree they — and the underlying sub task of relative pose estimation, which we probe with a dedicated head p — can be trained end-to-end from the navigation loss directly, without pose ground-truth.

the comparison of two images to extract *directional* information. Different strategies have been proposed to address this, and the recent literature seems to provide conflicting reports on their performances and up- and downsides. We aim to shed some light on this with an in-depth analysis.

Early works addressed the problem in a purely data-driven way by end-to-end training *Late Fusion* (eg. *Siamese*) image encoders [78], with mixed results. It was conjectured, that the weak and sparse learning signal from RL might not be sufficient to train high capacity encoders comparing images. Since the *ImageNav* task introduces a sub task of relative pose estimation (*"where is the goal with respect to my current position?"*), there is a strong case for modular methods separating perception and decision taking. For instance, Krantz *et al.* [37] employ local features and matching (*SuperGlue* [60]) combined with map and plan strategies for navigation. However, it has since been shown [8] that image correspondences can actually emerge directly in the cross-attention layers of a binocular transformer, provided it is appropriately pre-trained. This

strategy, used in the *DEBiT* agent [8] outperforms explicit matching approaches. Key to this solution is, firstly, pre-training, and secondly, the fusion of observation and goal such that comparisons between local parts (*e.g.* patches) of the two respective images can be calculated by the architecture, as opposed to comparing embedding vectors.

Recent work [68] however indicates that *ImageNav* can be addressed by very low-capacity convolutional encoders with channel-wise stacking and through RL training alone, without any pre-training. *FGPrompt* [68] achieves results which are on par with *DEBiT* with a fraction of the model capacity, which raises interesting questions:

Q1: Can *ImageNav* be learned by RL alone?

Q2: Which architecture can support that best?

Q3: Is the solution related to the estimation of directional information between observation and goal images?

The implications could potentially go beyond image goal navigation and have impact on the wider field of relative pose estimation and localization. In this paper we systematically disentangle the different mediators to shed a light on what choices matter in *ImageNav*, and we find that:

R1: Agents trained with **RL alone underperform** in realistic navigation settings.

R2: *Early Fusion* performs better, with **early patch-wise fusion being essential**, compared to *Late Fusion*.

R3: We find **correlations between navigation and (emergent) relative pose estimation performance**, which we probe from representations trained with RL.

Bonus: We discovered that success of recent frugal architectures using channel stacking and trained with RL alone is **mostly due to a simulator setting that allows agents to slide along walls**. Surprisingly, while this setting is known to hamper sim2real transfer since [32], we have obtained further insights: (i) unrealistic motion simulation has also a negative impact on the learned perception capabilities, and (ii) some learned capabilities can still be transferred to realistic settings **if the transfer includes weights of the perception module**.

2. Related Work

Navigation from visual observations – has been addressed in several different fields. Robotics for a long time focused on explicit modeling [13, 43, 45], which is mainly based on mapping and localization [12, 39, 69], explicit planning [36, 64] and low-level control [26, 59]. Accurate sensor and observation models are essential, followed by filtering, dynamical models and optimization techniques.

Modern AI-based solutions are typically trained on large-scale photorealistic simulators like Habitat [61] or AI2-Thor [35]. Modular agents [17, 53, 55] decompose the problem in sub modules, typically mapping, pose esti-

mation, planning and local decision taking. On the other end of this spectrum, end-to-end trained models directly map input to actions, with RL [30, 48, 70, 77] or Imitation Learning (IL) [23], offline-RL [67], or unsupervised RL [34, 42]. In recent work, end-to-end trained agents are combined with visual foundation models, eg. *DINOv2* [50] in *PoliFormer* [77], or binocular ones for image comparisons as in DEBiT [8]. Agent memory is implemented as recurrent representations [10, 31], occupancy maps [17], semantic maps [16], latent metric maps [6, 29, 51], topological maps [5, 18, 65], scene graphs [66], explicit episodic memory [20, 24, 25, 56], implicit representations [38, 46] or navigability [9]. In this work we investigate end-to-end trained agents with recurrent memory.

Image goal navigation – or "*ImageNav*", adds a matching aspect to navigation, as the agent needs to compare the goal image to the observed image. Explicit methods have addressed this with local feature matching [37], or by retrieving features from a topological map [5]. End-to-end trained agents compare images by extracting features with ResNets [1, 78] or ViTs [76], potentially stacking multiple observations over time [65], with ViTs followed by cross-attention [8], or by stacking images channel-wise [68]. Modular approaches have also been proposed [22, 74].

Depending on whether goal images are randomly chosen or correspond to images of semantically meaningful objects, the task can either be supported by semantic features or requires purely geometric image understanding. In this work we focus on the latter, and study how success depends on architectural key design choices. We review these choices in detail in Sec. 3.1 and their implementations in the state of the art in Sec. 3.2.

Relative pose estimation (RPE) – was tackled for decades with pixel-level image matching techniques [28, 49, 62]. Learning-based approaches have also been proposed [33], and to overcome the lack of generalizability of early methods, self-supervised techniques have been used [41]. More recently, *DUSt3R* [71] regresses pointmaps of each image expressed in the coordinate system of the first image while *MASt3R* [40] additionally learns an extra descriptor inspired by standard image matching. Both leverage *CroCo* pre-training [72, 73]. Similarly, *MicKey* [4] regresses pointmaps of each image in its respective coordinate system and learns descriptors to obtain correspondences, supervised by relative pose [7, 11]. These recent methods have led to impressive results, even under scenarios with little image overlap as in the MapFree-Relocalization benchmark [3]. In this work, we study RPE as a sub-task of ImageNav and probe it explicitly from agent representations.

3. Can we learn goal-perception with actions?

We study the *ImageNav* task in photo-realistic 3D environments, where an agent is given a goal image $\mathbf{g} \in \mathbb{R}^{3 \times H \times W}$
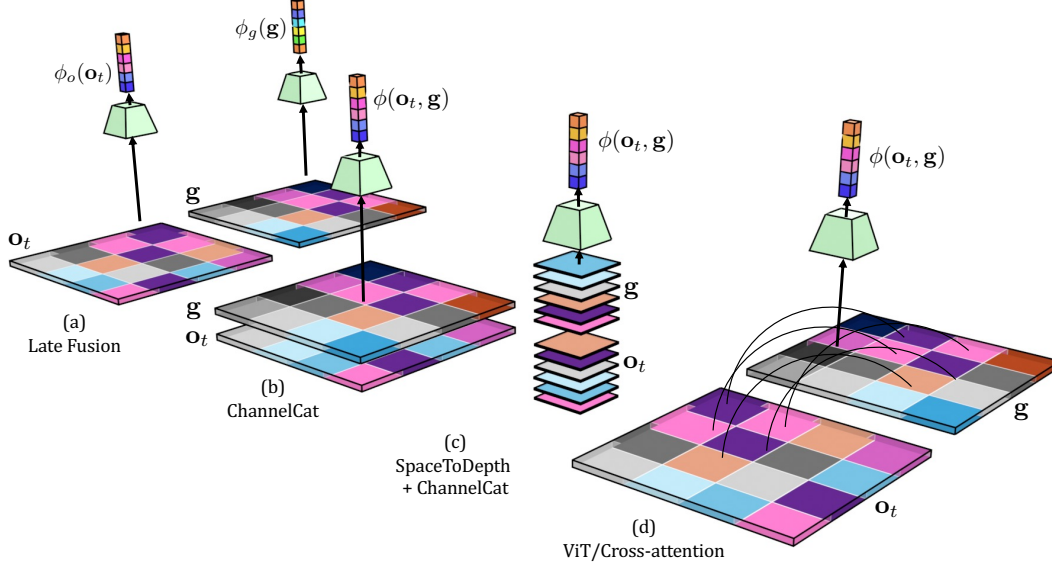
Figure 2. **Different architecture choices for binocular encoders** learning to compare the observed image $\mathbf{o}_t$ with the goal image $\mathbf{g}$: (a) *Late Fusion* encodes them separately and comparison is done "*late*" between embedding vectors $\phi_o(\mathbf{o}_t)$ and $\phi_g(\mathbf{g})$, making correspondence computations difficult. (b) *ChannelCat* stacks images over the channel dimension, followed by convolutional encoders $\phi([\mathbf{o}_t, \mathbf{g}]_{\text{dim}=1})$. It makes correspondence computations possible in principle if the CNN receptive field is big enough. (c) *SpaceToDepth* reshapes the patch dimension into the channel dimension. Combined with *ChannelCat* [68], it could allow correspondence to emerge in each layer directly through conv filters. (d) *Binocular ViTs* [8] model correspondence directly as cross-attention between patch tokens.

and is required to navigate from a starting location to the position shown in the goal. At each timestep t the agent observes an RGB image $\mathbf{o}_t \in \mathbb{R}^{3 \times H \times W}$, all images are of size 112×112. The action space is the discrete set $\mathcal{A} = \{$MOVE FORWARD 0.25m, TURN LEFT 10°, TURN RIGHT 10°, and STOP$\}$. An episode is considered successful if the agent is within 1m of the goal position *and* it calls the STOP action within its 1000 steps budget. We use the Habitat simulator [61] and the Gibson dataset [75].

All our experiments are done with different variants of the same identical base agent, which maintains a recurrent episodic memory $\mathbf{h}_t$, integrates the observation $\mathbf{o}_t$ and the goal image $\mathbf{g}$, and predicts actions $\mathbf{a}_t$:

$$
\begin{aligned}
\tilde{\mathbf{g}}_t &= \phi(\mathbf{o}_t, \mathbf{g}) & \text{// binocular perception} \\
\mathbf{h}_t &= h(\mathbf{h}_{t-1}, \tilde{\mathbf{g}}_t, \zeta(\mathbf{a}_{t-1})) & \text{// state update} \\
\mathbf{a}_t &\sim \pi(\mathbf{h}_t), & \text{// policy}
\end{aligned}
\tag{1}
$$

where h is the function updating the hidden state $\mathbf{h}_t$ of a GRU [21] and gating equations have been omitted for brevity; ϕ and ζ are trainable encoders, and π is a linear policy. All agent variants are trained from scratch with PPO [63] with reward function defined as in [8, 19]: $r_t = \text{K} \cdot \mathbf{1}_{\text{success}} - \Delta_t^{\text{Geo}} - \lambda$, where $\lambda = 0.01$ is a slack cost to encourage efficiency, $K = 10$, and Δ_t^{Geo} is the increase in geodesic distance to the goal.

3.1. Architecture design choices

At the heart of the problem is the sub task of learning to compare images $\mathbf{o}_t$ and $\mathbf{g}$ to infer directional information

related to relative pose estimation. In the classical *Image-Nav* task, goal images can be taken from any position and do not necessarily show a semantically meaningful object. Given the possible absence of semantic cues, which would have allowed to compare images on a global level, we conjecture that successful comparison requires the computation of some form of correspondence between local parts of the respective images. While pose estimation methods based on local feature matching perform this explicitly [57, 60], recent work have shown that this kind of correspondence can emerge implicitly from large-scale training without explicit supervision of matches, *e.g.* in *CroCo* [72] or *DUSt3R* [71].

In the context of robotics and navigation, the question arises whether it is possible to go a step further, and forgo pose-related losses completely, training image comparisons end-to-end together with the navigation agent from RL alone. This has been the promise of works like *FGPrompt* [68] and *OVRL-v2* [76], and we will analyze this in-depth.

Assuming that correspondence computations are necessary to extract directional information, we conjecture that these computations need to be supported by underlying network architectures. The basic building blocks for image comparison in the literature, also illustrated in Fig. 2, are:

Late Fusion networks use separate networks ϕ_o and ϕ_g to encode observation and goal. The representation fed to the agent is given as $\phi(\mathbf{o}_t, \mathbf{g}) = [\phi_o(\mathbf{o}_t), \phi_g(\mathbf{g})]$, where $[.]$ denotes concatenation, see Fig. 2(a). The comparison is thus performed "*late*" between embeddings $\phi_o(\mathbf{o}_t)$ and $\phi_g(\mathbf{g})$, which makes it generally harder to be done on a

local image level, unless output representations retain sufficient spatial structure going through encoder layers.

ChannelCat uses a single network to encode both observation and goal, which are channel stacked into one input image $\phi([\mathbf{o}_t, \mathbf{g}]_{\text{dim}=1})$, where 0 is the batch dimension and 1 the channel dimension — see Fig. 2(b).

If encoders are implemented as CNNs, a common choice in robotics for computational complexity reasons, comparisons between local parts of images can be done only if the receptive field of the network is big enough to encompass the full image. Given a sufficiently deep encoder correspondence can therefore be computed, but not necessarily efficiently, as individual matching results need to be accumulated in the output of different filter kernels.

If encoders are ViTs, each feature patch corresponds to a pair of patches having the same coordinates in both images. Correspondence can be more easily computed if different self-attention heads are learned with query and key projections focusing on one of the two channel groups of the two respective images.[1]

SpaceToDepth reshapes image patches into channel values [58]. *FGPrompt* [68] uses *SpaceToDepth* in combination with *ChannelCat* of observation and goal, achieving strong navigation results. We investigate whether a ResNet in this configuration can compute correspondences across a large spatial dimension through a single convolutional layer, somewhat reminiscent of cross-attention, only with very few parameters — see Fig. 2(c).

Cross-attention is a natural way to compute correspondences between local parts of images [8], as each patch in one image can be naturally linked to one or more patches in the other image through the cross-attention distribution, see Fig. 2(d).

3.2. Implementations in prior work

Early methods used *Late Fusion* approaches based on CNNs [1, 78], but later works switched to ViTs. Cross-attention for *ImageNav* was introduced in *DEBiT* [8] with code available,[2] and combined with pre-training, first for cross-view completion as in [72], then for relative pose and visibility estimation. The binocular ViT was combined with an additional CNN with only the observation as input, trained with RL, providing current state-of-the-art results. Interestingly, the authors have shown that pre-training is essential, as the high-capacity binocular transformer cannot be trained from scratch with RL only. The learning signal from the RL loss alone seems to be too weak to drive the en-

coder to discover correspondence computations. The inability of RL to train big encoders was somewhat confirmed in the *Late Fusion* method *OVRL-v2* [76], as "*switching from the 50.9M parameters ViT-Small to the 179.2M parameters ViT-Base produces another negative result: SR drops -2.4% while SPL minimally increases by +0.7%*".

FGPrompt [68] proposed channel stacking for *ImageNav*, with code available,[3] and it is based on previous code of *ZSEL* [1]. Both codebases use a non-standard ResNet9 architecture with several custom blocks, the most significant one being *SpaceToDepth* described earlier. Interestingly, *SpaceToDepth* was first introduced in *TResNet* [58] as a general purpose network block for computer vision optimized for memory efficiency, targeting image classification and object detection. Correspondence problems were not explicitly targeted. The ResNet9 encoder in this paper is exactly the same as those used in *ZSEL* and *FGPrompt*.

Given the good results *FGPrompt* obtained without any pre-training, 92% SR, vs. *DEBiT*'s 94% SR relying on pre-training, we raise the question whether it is possible to use the weak learning signal provided by RL to learn binocular perception modules capable of providing directional information. A positive answer would have significant impact also on training vision modules targeting RPE directly. In the experimental section we will dive into this question and benchmark architectures and design choices.

4. Experimental Setup

Unfortunately there are different evaluation protocols in the literature on Embodied AI, and in particular on the *ImageNav* task. We therefore train all agents ourselves in identical conditions and with the same experimental protocol. The agents are trained on the 72 scenes of the Gibson dataset [75] and use the standard Habitat episode definitions (a subset of the literature uses the definitions from *Episodic Transformers* [47] and another subset those from *NSRNL* [27]). We set the maximum episode length to the default value of 1000 steps and we train for 500M steps, which allows all agent variants to converge easily.

Implementation details – We test perception networks ϕ implemented as *ResNet9*, *ViT-Small*[4] and *DEBiT-B* architectures, available in their respective public repositories. We test *CannelCat*, *Late Fusion* and *SpaceToDepth* variants of ResNet and ViT, while for *DEBiT* we either use the pretrained binocular encoder or train the whole agent from scratch. Network versions are chosen based on size-performance tradeoff — see Supplementary for details. The function ζ embeds previous actions into a 32D feature, h is a GRU with 2 layers and hidden dimension 128, followed by a linear Actor-Critic policy π.

---

[1]As an illustrative example, consider an input image X corresponding to channel stacking of two images X_A and X_B. Then, if for a given attention head the Query projection only uses the channels of image X_A and the Key projection only use the channels of image X_B, then this self-attention on X mimics cross-attention between X_A and X_B.

[2]https://github.com/naver/debit

[3]https://github.com/XinyuSun/FGPrompt

[4]https://github.com/huggingface/pytorch-image-models

Validation – To avoid overfitting evaluation over the choice of checkpoints, while at the same time staying comparable, we use three different splits. We train on the standard Gibson train split (72 scenes) and report on the standard Gibson validation split (14 scenes), as most literature, but we use an independent custom holdout set of unused Gibson scenes to choose the checkpoint.

Metrics – Navigation performance is evaluated by success rate (SR), *i.e.*, fraction of episodes terminated within a distance of <1m to the goal by the agent calling the STOP action, and SPL [2], *i.e.*, SR weighted by the optimality of the path, $SPL = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{\ell_i^*}{\max(\ell_i, \ell_i^*)}$, where S_i be a binary success indicator in episode i, ℓ_i is the agent path length and ℓ_i^* the shortest path length.

Comparing the comparable – An important choice in the Habitat simulator is the binary Sliding switch, which, when set to True, allows the agent in simulation to slide along obstacles when colliding, against the more realistic behavior of stopping. This parameter is known to have a big impact on sim2real transfer [32]. In our own analysis, we saw that the increased difficulty of Sliding=False goes beyond sim2real transfer and also impacts task difficulty in simulation: when the agent hits a wall, it can in principle continue the episode, but is frequently restricted in its motion, often stuck, and typically needs to turn to recover.

While this setting is switched to True by default, there is consensus in the field that it should be set to False, as the goal is to decrease the sim2real gap and to evaluate agents in situations which are as close as possible to real physical robots and environments. Looking closer into the available code of navigation repositories, it seems that indeed most use Sliding=False (*DEBiT* [8], *ZSEL* [1], *ZSON* [44], *PIRLNav* [54]), with a couple of exceptions using Sliding=True, notably *FGPrompt* [68] (details in Supplementary). We therefore performed experiments with both settings, and were surprised to see that there is a big influence of this parameter. Given its impact, we color-coded the entries in all tables into three different groups: (i) Sliding=True : the visual encoder ϕ has been trained with this setting on; (ii) Sliding=False : it has been trained with this setting off; (iii) Pre-train : the visual encoder ϕ has been pre-trained on an auxiliary task, typically relative pose and visibility estimation (RPVE).

4.1. Benchmarking architectures

Tab. 1 summarizes results for different architectures and settings. With Sliding=True , *ChannelCat* (d)-(f) obtains excellent performance, close to *DEBiT-B* (h) which has a larger, more complex architecture, and is pre-trained on RPVE. Without pre-training, *DEBiT* is not exploitable, as also reported in [8]. *Late Fusion* architectures (a)-(c) underperform, and *SpaceToDepth* has no significant impact on either *ChannelCat* or *Late Fusion* models : being

Model	s2d†	Backbone	Slide=True SR	Slide=True SPL	Slide=False SR	Slide=False SPL
(a) Late Fusion	✗	ResNet9	13.8	8.0	12.8	7.1
(b) Late Fusion [1]	✓	ResNet9	12.5	7.6	13.2	8.9
(c) Late Fusion	✗	ViT-Small	12.5	6.7	6.9	4.5
(d) ChannelCat	✗	ResNet9	83.2	43.9	44.6	23.4
(e) ChannelCat [68]	✓	ResNet9	83.6	42.1	31.7	18.7
(f) ChannelCat	✗	ViT-Small	71.1	34.3	35.3	16.2
(g) Cross-attn	✗	DEBiT-B	0.0	0.0	0.0	0.0
(h) Cross-attn [8]	✗	DEBiT-B‡	90.5	60.3	81.7	52.0

Table 1. **Agents with different visual encoders** trained and validated with Sliding=True or Sliding=False . **s2d**†=*SpaceToDepth*. ‡=pre-trained for RPVE .

able to compare local patches across images in a single layer through convolutions does not translate into numerical gains. With Sliding=False the trends change dramatically. While the impact on the (previously already underperforming) *Late Fusion* architecture is similar, *ChannelCat*, (d)-(f), now breaks down and performance is halved, or less. In contrast, *DEBiT* is able to cope well with the more realistic Sliding=False setting, arguably because of its strong pre-trained visual encoder, confirming the importance of visual pre-training.

To summarize, and coming back to *Q1* and *Q2* raised in Sec. 1, we can conclude the following:

R1: In realistic settings (Sliding=False), agents trained with RL alone perform significantly worse than *DEBiT* with pre-trained visual encoder.

R2: *Early Fusion* agents perform better than *Late Fusion* counterparts, with early patch-wise fusion being the best option, but requiring pre-training.

Additionally, we found that recent success of agents based on small CNNs trained end-to-end with RL only, was mostly due to setting Sliding=True in Habitat.

4.2. Transferring capabilities obtained w. "Sliding"

The huge impact of the Sliding simulator setting came somewhat as a surprise. While its role in sim2real transfer of point goal navigation was established in [32], the common conjecture was that it impacted "low-level navigation", *i.e.*, finding free space, avoiding obstacles, capabilities which are independent of the extraction of goal directional information. Our experiments, however, suggest that the choice of binocular visual encoder ϕ is impacted by this hyper-parameter, which should be related to motion only.

This then raises the question, whether (i) the two agents trained in their respective settings have similar capabilities but perform differently purely due to the difference in task difficulty, or (ii) whether, for some reason, training with sliding actually leads to different and perhaps better performing agent capabilities. We tested these hypotheses by performing experiments loading the weights

Checkpoint	Perception	Action			(%)	
	ϕ	h	ζ	π	**SR**	**SPL**
(a) Load all "false"	$f*$	$f*$	$f*$	$f*$	31.7	18.7
(b) Load all "true"	$t*$	$t*$	$t*$	$t*$	54.6	27.5
(c) Load all "true"	$t\rightarrow$	$t\rightarrow$	$t\rightarrow$	$t\rightarrow$	**65.7**	**34.1**
(d) Load action "true"	↻	$t*$	$t*$	$t*$	0.0	0.0
(e) Load action "true"	↻	$t\rightarrow$	$t\rightarrow$	$t\rightarrow$	6.1	4.8
(f) Load perception "true"	$t*$	↻	↻	↻	26.4	14.3
(g) Load perception "true"	$t\rightarrow$	↻	↻	↻	38.5	20.3

Table 2. **OOD behavior and cross-domain transfer (*Channel-Cat+SpaceToDepth agent*):** Can navigation knowledge learned with sliding be transferred to the non-sliding setting through fine-tuning? f: load from agent trained with `Sliding=False`, t: load from agent trained with `Sliding=True`, $*$: frozen , $\rightarrow$: finetune , ↻: re-train from scratch .

Checkpoint	Perception	Action			(%)	
	ϕ	h	ζ	π	**SR**	**SPL**
(a) Load all "false"	$p*$	$f*$	$f*$	$f*$	**81.7**	**52.0**
(b) Load all "true"	$p*$	$t*$	$t*$	$t*$	59.5	35.4
(c) Load all "true"	$p*$	$t\rightarrow$	$t\rightarrow$	$t\rightarrow$	79.6	46.9

Table 3. **OOD behavior (*DEBiT agent*):** f: load from agent trained with `Sliding=False`, t: load from agent trained with `Sliding=True`, p: pre-trained with RPVE, $*$: frozen , $\rightarrow$: finetune .

of an agent trained with `Sliding=True` and finetuning it with `Sliding=False`. For these experiments we used the *ChannelCat + SpaceToDepth* agent, for which the difference in SR dropped from 83.6% to 31.7% when sliding was switched off (cf. Tab. 1).

Tab. 2 shows the results of these experiments. Taking the agent trained with `Sliding=True` and validating it on `False`, row (b), makes SR jump to 54.6%, compared to the baseline of 31.7%, trained on `False` only. This is a highly surprising result. Although the performance is a far cry of the 83.6% obtained by training and validating with `True`, it indicates that a certain amount of transfer is possible — some capabilities are learned with sliding enabled. Fine-tuning this agent for 100M steps on `False` provides further gains and yields SR=65.7%, row (c).

To pinpoint the effect, we loaded different parts of the agent trained with `Sliding=True`: only the perception part (weights of the visual encoder ϕ), only the action part (GRU h, policy π, previous action encoder ζ) or both. Transferring the action part of the agent, *i.e.* GRU h, policy π, encoder ζ, rows (d) and (e), does not lead to any discernible performance: learning the upstream visual encoder ϕ from scratch through pre-loaded downstream weights of the GRU and policy seems a very hard task. However, transferring the perception weights ϕ and training the action part from scratch, rows (f) and (g), leads to exploitable results, although significantly lower than loading the full agent weights. With a SR of 38.5% for the finetuning version, row (g), these are still higher than the baseline of 31.7% trained from scratch on `False`. In Tab. 3 we perform similar experiments with *DEBiT* which, however, uses a pre-trained visual encoder ϕ that is kept frozen. We load the rest of the agent trained with `Sliding=True` and evaluate it with `False` (b): in this case, no transfer happens and performance drops significantly. Finetuning the agent trained with `True` on `False`, (c), improves the `Sliding=True` policy, but not to the extent of reaching the performance of the agent trained on-distribution, *i.e.* `Sliding=False`, ar-

guably because RPVE pre-training of perception is already providing a strong visual encoder.

We the following lessons from these experiments. The difference between `Sliding=True` or `False` cannot only be described by the task difficulty alone. The easier task (`True`) allows to learn additional capabilities, which transfer to the harder task, and which are partially related to perception (since the performance in Tab. 2(g) > 2(a)), and also related to action / sequential decision taking, since Tab. 2(b) ≫ 2(a) and Tab. 2(b) ≫ 2(g). However, we conjecture that the transfer of the knowledge stored in the action component is *also* highly linked to improving the perception skills, as it does not happen when ϕ is pre-trained and frozen (Tab. 3(a) ≫ 3(b), and Tab. 3(a) > 3(c)). One possible explanation is that only training with `False` leads to undertraining of, both, action and perception: the policy keeps getting stuck (which we empirically confirmed) and does not learn to cope with the last meters of each episode; this, in turn, leads to undertraining the comparison between the (hardly ever seen) goals and observations.

4.3. Probing RPVE capabilities

Given that a certain limited capacity for visual reasoning has been learned by the visual encoder ϕ when the full agent has been trained for navigation with `Sliding=True`, we pursue this question further and directly investigate how well the different visual encoders can deal with extracting directional information. Or, motivated differently, given the potential impact on relative pose estimation of training these visual encoders from navigation losses only without pose supervision, we evaluate how well they work for this task. We took the frozen visual encoders ϕ of the agents evaluated in Tab. 1 and trained probing heads p on top of them predicting relative pose and visibility (RPVE):

$$(\mathbf{t}, \mathbf{R}, v) = p\Big(\phi(\mathbf{o}, \mathbf{g})\Big). \tag{2}$$

In line with [8], relative pose between the observed image and the goal image is composed of two components, translation $\mathbf{t} \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$. Visibility quantifies the amount of overlap between the two images, necessary since in navigation settings the goal image might not even be observed at certain moments. It is defined as the proportion of 16×16 patches of the goal image $\mathbf{g}$ which are

Model (Table nr. + row)	s2d†	Backbone	S‡	%corr.poses 1m,10°	2m,20°	%corr.vis. <0.05
Late Fusion 1a	✗	ResNet9	✓	7.6	26.0	13.8
Late Fusion 1b	✓	ResNet9	✓	9.0	29.6	16.1
ChannelCat 1d	✗	ResNet9	✓	11.4	29.3	13.7
ChannelCat 1e	✓	ResNet9	✓	18.4	41.6	20.8
Late Fusion 1a	✗	ResNet9	✗	7.8	26.8	13.2
Late Fusion 1b	✓	ResNet9	✗	8.7	28.5	16.1
ChannelCat 1d	✗	ResNet9	✗	9.8	26.9	13.8
ChannelCat 1e	✓	ResNet9	✗	12.5	31.9	19.2
ChannelCat 2c	✓	ResNet9	→	18.2	41.4	21.1
ChannelCat 2d	✓	ResNet9	→	5.8	22.9	6.7
ChannelCat 2e	✓	ResNet9	→	7.2	26.1	11.9
ChannelCat 2g	✓	ResNet9	→	18.6	41.6	21.0
Cross-attn 1h	✗	DEBiT-B	N/A	92.1	96.8	88.8
Late fusion	✗	DEBiT-B	N/A	14.8	38.6	19.6
Late fusion	✗	DINOv2	N/A	12.9	34.0	22.7

Table 4. **Probing relative pose and visibility estimation:** representations $\phi(\mathbf{o}, \mathbf{g})$ trained with an RL (navigation) loss are frozen, and then we train a probing head. *The last block of methods in italic* is not comparable, as the encoders were pre-trained: *DEBiT* [8] was pre-trained on RPVE losses, DINOv2 [50] with SSL. We report RPVE performance on a hold-out set. **s2d†**=*SpaceToDepth*, **S‡** = Sliding=True. The third block shows agents which have been finetuned (→) from True to False, see Tab. 2.

(a) Nav evaluated w. Sliding=True   (b) Nav evaluated w. False

Figure 3. **Nav *vs*. Rel-pose:** navigation perf. (SR,%) plotted against pose est. probing accuracy (% for err $<2m, 20°$) for 4 types of visual encoders ϕ: trained w. sliding , trained w/o sliding , pre-trained w. RPVE , trained w. sliding, finetuned w/o ; (LF = Late Fusion, CC=ChannelCat). The dashed line ------- relates the finetuned models to the same model trained w/o sliding .

visible in the observed image **o**.

The probing heads p are adapted to the different architectures, while at the same time providing a comparable capacity of $\sim 3M$ parameters for each head variant. They collect features at the last spatial representation (feature map for ResNet, patch embeddings for ViT, concatenation of these for *Late Fusion* representations) and first linearly project token-wise to a lower dimension, before flattening to obtain a global representation which is passed to an MLP with 1024 hidden units. The low dimension of the projection changes according to architectures (as the spatial resolution, *i.e.*, the number of tokens change) to make the parameter counts similar: 64 for ViTs, 192 (resp. 16) for ResNet9 with (resp. without) *SpaceToDepth*.

As in [8], we generated a probing dataset by combining the 3D scene datasets Gibson [75], MP3D [14], and HM3D [52], following their standard train/val scene splits. We sample pairs of points randomly in the scene, calculate the shortest path between them, and extract 10 intermediate poses on the path. These 10 poses are then each combined with the end point to form pairs of observation **o** and goal **g**. Ground-truth for relative pose is directly available from the simulator, and visibility can be estimated from the two point clouds generated from the respective images. We generated around 68M image pairs total.

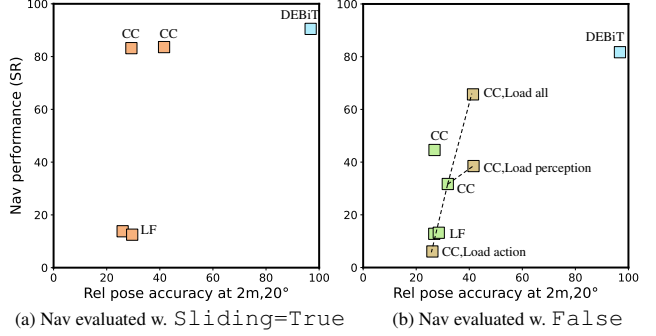We train the probing heads p with a loss combining all three components, translation, rotation and visibility, where pose supervision is switched off when visibility is low,

$$\mathcal{L}_p = \sum_i \Big[|v_i - v_i^*| + \mathbf{1}_{v_i^* > \tau} \big\{ |\mathbf{t}_i - \mathbf{t}_i^*| + |\mathbf{R}_i - \mathbf{R}_i^*| \big\} \Big], \quad (3)$$

where i indexes image pairs over the probing dataset, $\mathbf{t}_i^*, \mathbf{R}_i^*, v_i^*$ denote ground truth values, $\mathbf{1}$ is the binary indicator function, $|.|$ denotes the L_1 loss and τ is a threshold.

Metrics – relative pose is evaluated over the pairs with visibility over τ in the percentage of correct poses for given thresholds on distance and angle, *e.g.* 1 meter and 10°. Visibility is evaluated over all pairs by its accuracy at ± 0.05, *i.e.*, the percentage of prediction within a 0.05 margin of the ground-truth value.

Results – Tab. 4 gives results comparing RPVE performance on a validation set for all tested visual encoders, and Fig. 3 provides a scatter plot relating it to navigation performance (taken from Tab. 1, 2). While the *Channel-Cat* architecture obtains high navigation performance in the Sliding=True setting, 3(a), its pose estimation performance remains limited. *Late Fusion* models generally provide low performance both for navigation and pose estimation. In the Sliding=False setting, 3(a), the navigation performance significantly drops for the *ChannelCat* models, as already discussed in Sec. 4.1, and also somewhat for pose estimation. Transferring the models trained with True on the False setting provides gains not only in navigation performance, as already discussed, but also on pose estimation — only when at least the weights of the visual encoder ϕ are transferred, but in particular when the whole agent is transferred and finetuned. This further corroborates our conjecture, that the perception model is undertrained when sliding is disabled. Finally, the best performance is obtained by the *DEBiT* model based on ViTs and cross-attention, both in navigation and pose probing. However, the pose estimation performance is *not comparable for obvious reasons* — the model has been pre-trained and frozen for this task. Its performance is given for information.
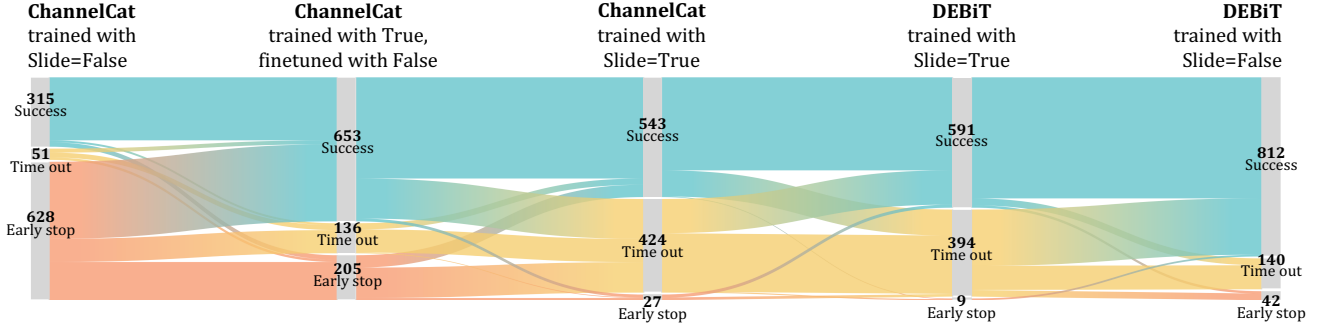
Figure 4. **Analysis of navigation behavior:** Sankey plots show the distribution of success/failure codes over 994 test episodes for different models, and their "flow" between certain pairs of models. For instance, the strength of the connection between "Time out" (left) and "success" (right) indicates how many episodes toggled from one to the other when switching from the left to the right model.

In Tab. 4 we also provide pose probing results for two additional models, where we explore whether the pose estimation performance of *Late Fusion* can be increased by using two different pre-trained and frozen encoders, namely DINOv2 and the ViT encoder part of *DEBiT*, without the cross-attention layers combining them[5]. They both underperform, further corroborating the importance of comparing visual representations early, allowing to compute correspondences between local image parts. These experiments support a positive answer to *Q3* in Sec. 1: there is a clear correlation between navigation and RPE performance.

4.4. Analysis of navigation behavior

In Fig. 4 we provide Sankey plots showing the differences in navigation behavior as a distribution of success/error types over 994 test episodes evaluated in environments with `Sliding=False`, and how this behavior varies across different models and different training settings. Here we distinguish between unsuccessful episodes due to *Time out*, when the agent does not complete an episode in the 1000 steps budget, and *Early stop*, when the agent terminates the episode but is further away than the 1m success threshold.

Left, the *ChannelCat* agent trained on `Sliding=False`, Tab. 2(a), stops too early in most of the episodes, confirming that its components are not properly trained, probably due to the difficulty of the task. Next, finetuning on `False` the *ChannelCat* agent trained with `True`, Tab. 2(c), allows to obtain an agent with good navigation capabilities. It improves the *ChannelCat* agent trained with `True`, Tab. 2(b), in the middle in Fig. 4, whose main failure mode is time out. We observe that most of these time-outs occur far from the goal and are due to the agent getting stuck, likely because it learned to slide along obstacles, which is not possible with `False`. On the 4th column, *DEBiT* trained with `True`, Tab. 3(b), exhibits a similar behavior, while *DEBiT* trained with `False`, Tab. 3(a), on the right-most column,

can effectively solve the complex navigation task, arguably thanks to its pretrained, high capacity, perception module.

5. Discussion and conclusion

We have studied image goal navigation and raised questions on agent architecture and whether components dedicated to vision or decision taking should be trained jointly with RL or separately. We have shown that the success of recent architectures based on channel stacking can be linked to a simulator setting allowing sliding. While agents trained with sliding enabled are known to transfer badly to real environments, where this is not possible, our analysis has shown that capabilities learned with this sliding setting can actually be partially transferred to more realistic environment settings. While sliding is a property of agent dynamics and not inherently linked to perception, we have shown that it significantly impacts training of the visual encoders of end-to-end trained agents: *transferring capabilities to the realistic setting only seems to be successful if it also involves transferring the visual encoder weights*. We conjecture that training with the realistic setting leads to undertraining both the perception module, as the goal is seen more rarely, and the rest of the agent, which has a difficulty to cover later portions of the episodes.

We also compared and analyzed different widely-used architectures: *Late Fusion, ChannelCat, SpaceToDepth* projections and *Cross-attention*. Results suggest that *successful architectures require support for early fusion of representations*, allowing to compute correspondences between local parts of images: *Late Fusion* generally underperforms. As expected, we also found a *correlation between navigation performance and (emerging) relative pose estimation performance*, which we probed with explicit heads. Finally, we argue that, up to our knowledge, there does not seem to exist a simple solution to learn with RL alone image goal navigation end-to-end with simple low-capacity architectures and without pre-training, which we judge to remain essential for goal-oriented navigation.

---

[5]*DEBiT* first encodes each image with a ViT and then passes both embeddings to a decoder using cross-attention, cf. [8].

References

[1] Ziad Al-Halah, Santhosh Kumar Ramakrishnan, and Kristen Grauman. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 4, 5

[2] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Zamir. On Evaluation of Embodied Navigation Agents. arXiv:1807.06757, 2018. 5

[3] Eduardo Arnold, Jamie Wynn, Sara Vicente, Guillermo Garcia-Hernando, Aron Monszpart, Victor Prisacariu, Daniyar Turmukhambetov, and Eric Brachmann. Map-free visual relocalization: Metric pose relative to a single image. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[4] Axel Barroso-Laguna, Sowmya Munukutla, Victor Prisacariu, and Eric Brachmann. Matching 2D images in 3D: Metric relative pose from metric correspondences. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[5] Edward Beeching, Jilles Dibangoye, Olivier Simonin, and Christian Wolf. Learning to Plan with Uncertain Topological Maps. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[6] Edward Beeching, Jilles Dibangoye, Olivier Simonin, and Christian Wolf. EgoMap: Projective Mapping and Structured Egocentric Memory for Deep RL. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2020. 2

[7] Aritra Bhowmik, Stefan Gumhold, Carsten Rother, and Eric Brachmann. Reinforced feature points: Optimizing feature detection and description for a high-level task. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[8] Guillaume Bono, Leonid Antsfeld, Boris Chidlovskii, Philippe Weinzaepfel, and Christian Wolf. End-to-End (Instance)-Image Goal Navigation through Correspondence as an Emergent Phenomenon,. In *International Conference on Learning Representations (ICLR)*, 2024. 1, 2, 3, 4, 5, 6, 7, 8

[9] Guillaume Bono, Leonid Antsfeld, Assem Sadek, Gianluca Monaci, and Christian Wolf. Learning with a Mole: Transferable Latent Spatial Representations for Navigation without Reconstruction. In *International Conference on Learning Representations (ICLR)*, 2024. 2

[10] Guillaume Bono, Hervé Poirier, Leonid Antsfeld, , Gianluca Monaci, Boris Chidlovskii, and Christian Wolf. Learning to Navigate Efficiently and Precisely in Real Environments. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[11] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[12] Guillaume Bresson, Zayed Alsayed, Li Yu, and Sébastien Glaser. Simultaneous localization and mapping: A survey of current trends in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 2017. 2

[13] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. The interactive museum tour-guide robot. In *American Association for Artificial Intelli- gence (AAAI)*, 1998. 2

[14] Angel X. Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from rgb-d data in indoor environments. In *International Conference on 3D Vision (3DV)*, 2018. 7

[15] Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, Roozbeh Mottaghi, Jitendra Malik, and Devendra Singh Chaplot. Goat: Go to any thing. In *Robotics: Science and Systems (RSS)*, 2024. 1

[16] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object Goal Navigation using Goal-Oriented Semantic Exploration. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 2

[17] 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

[18] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural Topological SLAM for Visual Navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[19] Prithvijit Chattopadhyay, Judy Hoffman, Roozbeh Mottaghi, and Aniruddha Kembhavi. Robustnav: Towards benchmarking robustness in embodied navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[20] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-Language Navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[21] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014. 3

[22] Abhishek Das, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Neural modular control for embodied question answering. In *Conference on Robot Learning (CoRL)*, 2018. 2

[23] Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned Imitation Learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 2

[24] Heming Du, Xin Yu, and Liang Zheng. VTNet: Visual transformer network for object goal navigation. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[25] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene Memory Transformer for Embodied Agents in Long-Horizon Tasks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[26] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 1997. 2

[27] 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 *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 4

[28] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[29] João F. Henriques and Andrea Vedaldi. MapNet: An allocentric spatial memory for mapping environments. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[30] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[31] Steeven Janny, Hervé Poirier, Leonid Antsfeld, Guillaume Bono, Gianluca Monaci, Boris Chidlovskii, Francesco Giuliari, Alessio Del Bue, and Christian Wolf. Reasoning in visual navigation of end-to-end trained agents: a dynamical systems approach. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

[32] Abhishek Kadian, Joanne Truong, Aaron Gokaslan, Alexander Clegg, Erik Wijmans, Stefan Lee, Manolis Savva, Sonia Chernova, and Dhruv Batra. Sim2real predictivity: Does evaluation in simulation predict real-world performance? *IEEE Robotics and Automation Letters (RA-L)*, 2020. 2, 5

[33] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: a Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *International Conference on Computer Vision (ICCV)*, 2015. 2

[34] Junsu Kim, Seohong Park, and Sergey Levine. Unsupervised-to-online reinforcement learning. arXiv:2408.14785, 2024. 2

[35] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. AI2-Thor: An interactive 3D environment for visual AI. arXiv:1712.05474, 2017. 2

[36] Kurt Konolige. A gradient method for realtime robot control. In *International Conference on Intelligent Robots and Systems (IROS)*, 2000. 2

[37] Jacob Krantz, Theophile Gervet, Karmesh Yadav, Austin Wang, Chris Paxton, Roozbeh Mottaghi, Dhruv Batra, Jitendra Malik, Stefan Lee, and Devendra Singh Chaplot. Navigating to objects specified by images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2

[38] Obin Kwon, Jeongho Park, and Songhwai Oh. Renderable neural Radiance Map for Visual Navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[39] Mathieu Labbé and François Michaud. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 2019. 2

[40] Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3D with MASt3R. In *European Conference on Computer Vision (ECCV)*, 2024. 2

[41] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *International Conference on Robotics and Automation (ICRA)*, 2018. 2

[42] Grace Liu, Michael Tang, and Benjamin Eysenbach. A single goal is all you need: Skills and exploration emerge from contrastive RL without rewards, demonstrations, or subgoals. In *International Conference on Learning Representations (ICLR)*, 2025. 2

[43] Steve Macenski, Francisco Martín, Ruffin White, and Jonatan Ginés Clavero. The marathon 2: A navigation system. In *International Conference on Intelligent Robots and Systems (IROS)*, 2020. 2

[44] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: Zero-shot object-goal navigation using multimodal goal embeddings. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 5

[45] Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. In *International Conference on Robotics and Automation (ICRA)*, 2010. 2

[46] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Multi-Object Navigation with dynamically learned neural implicit representations. In *International Conference on Computer Vision (ICCV)*, 2023. 2

[47] Lina Mezghani, 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. 4

[48] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andy Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to navigate in complex environments. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[49] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robotics*, 2015. 2

[50] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research Journal (TMLR)*, 2024. 2, 7

[51] Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *In-*

ternational Conference on Learning Representations (ICLR), 2018. 2

[52] 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 *NeurIPS Datasets and Benchmarks Track*, 2021. 7

[53] Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[54] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. PIRLNav: Pretraining with imitation and rl finetuning for objectnav. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5

[55] Sonia Raychaudhuri, Tommaso Campari, Unnat Jain, Manolis Savva, and Angel X. Chang. MOPA: Modular object navigation with pointgoal agents. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WCAV)*, 2024. 2

[56] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *Transactions on Machine Learning Research Journal (TMLR)*, 2022. 2

[57] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2D2: Reliable and Repeatable Detector and Descriptor. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 3

[58] Tal Ridnik, Hussam Lawen, Asaf Noy, Emanuel Ben Baruch, Gilad Sharir, and Itamar Friedman. TResNet: High Performance GPU-Dedicated Architecture. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021. 4

[59] Christoph Rösmann, Frank Hoffmann, and Torsten Bertram. Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control. In *European Control Conference (ECC)*, 2015. 2

[60] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 3

[61] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[62] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[63] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347, 2017. 3

[64] James A. Sethian. A Fast Marching Level Set Method for Monotonically Advancing Fronts. *Proceedings of the National Academy of Sciences (PNAS)*, 93(4):1591–1595, 1996. 2

[65] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. In *Conference on Robot Learning (CoRL)*, 2023. 2

[66] Kunal Pratap Singh, Jordi Salvador, Luca Weihs, and Aniruddha Kembhavi. A general purpose supervisory signal for embodied agents. In *International Conference on Computer Vision (ICCV)*, 2022. 2

[67] Jost Tobias Springenberg, Abbas Abdolmaleki, Jingwei Zhang, Oliver Groth, Michael Bloesch, Thomas Lampe, Philemon Brakel, Sarah Bechtle, Steven Kapturowski, Roland Hafner, et al. Offline actor-critic reinforcement learning scales to large models. In *International Conference on Machine Learning (ICML)*, 2024. 2

[68] Xinyu Sun, Peihao Chen, Jugang Fan, Jian Chen, Thomas Li, and Mingkui Tan. FGPrompt: fine-grained goal prompting for image-goal navigation. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024. 2, 3, 4, 5

[69] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press Cambridge, 2005. 2

[70] Shagun Uppal, Ananye Agarwal, Haoyu Xiong, Kenneth Shaw, and Deepak Pathak. SPIN: Simultaneous perception, interaction and navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[71] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3d vision made easy. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 3

[72] Philippe Weinzaepfel, Vincent Leroy, Thomas Lucas, Romain Brégier, Yohann Cabon, Vaibhav Arora, Leonid Antsfeld, Boris Chidlovskii, Gabriela Csurka, and Jerome Revaud. CroCo: Self-Supervised Pretraining for 3D Vision Tasks by Cross-View Completion. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3, 4

[73] Philippe Weinzaepfel, Thomas Lucas, Vincent Leroy, Yohann Cabon, Vaibhav Arora, Romain Brégier, Gabriela Csurka, Leonid Antsfeld, Boris Chidlovskii, and Jerome Revaud. CroCo v2: Improved cross-view completion pretraining for stereo matching and optical flow. In *International Conference on Computer Vision (ICCV)*, 2023. 2

[74] Qiaoyun Wu, Jun Wang, Jing Liang, Xiaoxi Gong, and Dinesh Manocha. Image-goal navigation in complex environments via modular learning. *IEEE Robotics and Automation Letters (RA-L)*, 2022. 2

[75] Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: real-world perception for embodied agents. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3, 4, 7

[76] Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. OVRL-V2: A sim-

ple State-of-art Baseline for ImageNav and ObjectNav. arXiv:2303.07798, 2023. 1, 2, 3, 4

[77] Kuo-Hao Zeng, Zichen Zhang, Kiana Ehsani, Rose Hendrix, Jordi Salvador, Alvaro Herrasti, Ross Girshick, Aniruddha Kembhavi, and Luca Weihs. Poliformer: Scaling on-policy RL with transformers results in masterful navigators. In *Conference on Robot Learning (CoRL)*, 2024. 2

[78] 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)*, 2017. 1, 2, 4