ENVIRONMENT PREDICTIVE CODING FOR EMBODIED AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce *environment predictive coding*, a self-supervised approach to learn environment-level representations for embodied agents. In contrast to prior work on self-supervised learning for images, we aim to jointly encode a series of images gathered by an agent as it moves about in 3D environments. We learn these representations via a *zone prediction task*, where we intelligently mask out portions of an agent's trajectory and predict them from the unmasked portions, conditioned on the agent's camera poses. By learning such representations on a collection of videos, we demonstrate successful transfer to multiple downstream navigation-oriented tasks. Our experiments on the photorealistic 3D environments of Gibson and Matterport3D show that our method outperforms the state-of-the-art on challenging tasks with only a limited budget of experience.

1 INTRODUCTION

In visual navigation tasks, an intelligent embodied agent must move around a 3D environment using its stream of egocentric observations to sense objects and obstacles, typically without the benefit of a pre-computed map. Significant recent progress on this problem can be attributed to the availability of large-scale visually rich 3D datasets (Chang et al., 2017; Xia et al., 2018; Straub et al., 2019), developments in high-quality 3D simulators (Anderson et al., 2018b; Kolve et al., 2017; Savva et al., 2019a; Xia et al., 2020), and research on deep memory-based architectures that combine geometry and semantics for learning representations of the 3D world (Gupta et al., 2017; Henriques & Vedaldi, 2018; Chen et al., 2019; Fang et al., 2019; Chaplot et al., 2020b;c).

Deep reinforcement learning approaches to visual navigation often suffer from sample inefficiency, overfitting, and instability in training. Recent contributions work towards overcoming these limitations for various navigation and planning tasks. The key ingredients are learning good image-level representations (Das et al., 2018; Gordon et al., 2019; Lin et al., 2019; Sax et al., 2020), and using modular architectures that combine high-level reasoning, planning, and low-level navigation (Gupta et al., 2017; Chaplot et al., 2020b; Gan et al., 2019; Ramakrishnan et al., 2020a).

Prior work uses supervised image annotations (Mirowski et al., 2016; Das et al., 2018; Sax et al., 2020) and self-supervision (Gordon et al., 2019; Lin et al., 2019) to learn good image representations that are transferrable and improve sample efficiency for embodied tasks. While promising, such learned image representations only encode the scene in the nearby locality. However, embodied agents also need higher-level semantic and geometric representations of their history of observations, grounded in 3D space, in order to reason about the larger environment around them.

Therefore, a key question remains: *how should an agent moving through a visually rich 3D environment encode its series of egocentric observations?* Prior navigation methods build *environment-level* representations of observation sequences via memory models such as recurrent neural networks (Wijmans et al., 2020), maps (Henriques & Vedaldi, 2018; Chen et al., 2019; Chaplot et al., 2020b), episodic memory (Fang et al., 2019), and topological graphs (Savinov et al., 2018; Chaplot et al., 2020c). However, these approaches typically use hand-coded representations such as occupancy maps (Chen et al., 2019; Chaplot et al., 2020b; Ramakrishnan et al., 2020a; Karkus et al., 2019; Gan et al., 2019) and semantic labels (Narasimhan et al., 2020; Chaplot et al., 2020a), or specialize them by learning end-to-end for solving a specific task (Wijmans et al., 2020; Henriques & Vedaldi, 2018; Parisotto & Salakhutdinov, 2018; Cheng et al., 2018; Fang et al., 2019).

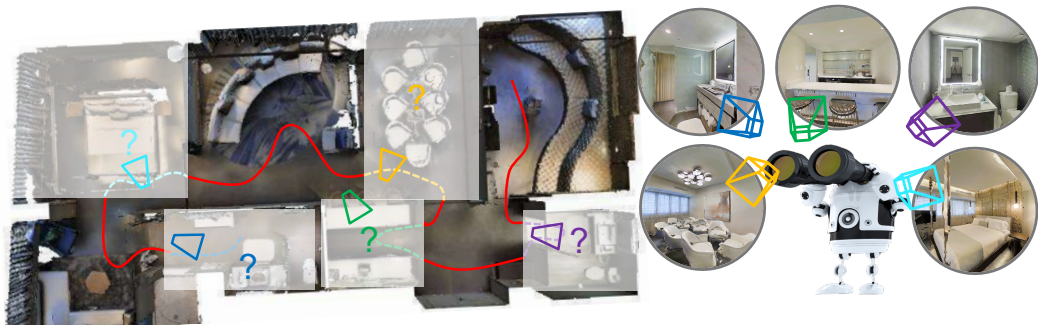

Figure 1: **Environment Predictive Coding:** During self-supervised learning, our model is given video walk-throughs of various 3D environments. We mask portions out of the trajectory (dotted lines) and learn to infer them from the unmasked parts (in red). We specifically mask out all overlapping views in a local neighborhood to limit the content shared with the unmasked views. The resulting EPC encoder builds environment-level representations of the seen content that are predictive of the unseen content (marked with a "?"), conditioned on the camera poses. The agent then uses this learned encoder in multiple navigational tasks in novel environments.

In this work, we introduce *environment predictive coding* (EPC), a self-supervised approach to learn flexible representations of 3D environments that are transferrable to a variety of navigation-oriented tasks. The key idea is to learn to encode a series of egocentric observations in a 3D environment so as to be predictive of visual content that the agent has not yet observed. For example, consider an agent that just entered the living room in an unfamiliar house and is searching for a refrigerator. It must be able to predict where the kitchen is and reason that it is likely to contain a refrigerator. The proposed EPC model aims to learn representations that capture these natural statistics of real-world environments in a self-supervised fashion, by watching videos recorded by other agents. See Fig. 1.

To this end, we devise a self-supervised *zone prediction* task in which the model learns environment embeddings by watching egocentric view sequences from other agents navigating in 3D environments in pre-collected videos. Specifically, we segment each video into zones of visually and geometrically connected views, while ensuring limited overlap across zones in the same video. Then, we randomly mask out zones, and predict the masked views conditioned on both the unmasked zones' views and the masked zones' camera poses. Intuitively, to perform this task successfully, the model needs to reason about the geometry and semantics of the environment to figure out what is missing. We devise a transformer-based model to infer the masked visual features. Our general strategy can be viewed as a context prediction task in sequential data (Devlin et al., 2018; Sun et al., 2019b; Han et al., 2019)—but, very differently, aimed at representing high-level semantic and geometric priors in 3D environments to aid embodied agents who act in them.

Through extensive experiments on Gibson and Matterport3D, we show that our method achieves good improvements on multiple navigation-oriented tasks compared to state-of-the-art models and baselines that learn image-level embeddings.

2 Related Work

Self-supervised visual representation learning: Prior work leverages self-supervision to learn image and video representations from large collections of unlabelled data. Image representations attempt proxy tasks such as inpainting (Pathak et al., 2016) and instance discrimination (Oord et al., 2018; Chen et al., 2020; He et al., 2020), while video representation learning leverages signals such as temporal consistency (Wei et al., 2018; Fernando et al., 2017; Kim et al., 2019) and contrastive predictions (Han et al., 2019; Sun et al., 2019a). The VideoBERT project (Sun et al., 2019a;b) jointly learns video and text representations from unannotated videos via filling in masked out information. Dense Predictive Coding (Han et al., 2019; 2020) learns video representations that capture the slow-moving semantics in videos. Whereas these methods focus on capturing human activity for video recognition, we aim to learn geometric and semantic cues in 3D spaces for embodied agents. Accordingly, unlike the existing video models (Sun et al., 2019a;b; Han et al., 2019), our approach is grounded in the 3D relationships between views.

Representation learning via auxiliary tasks for RL: Reinforcement learning approaches often suffer from high sample complexity, sparse rewards, and unstable training. Prior work tackles these

challenges by using auxiliary tasks for learning image representations (Mirowski et al., 2016; Gordon et al., 2019; Lin et al., 2019; Shen et al., 2019; Ye et al., 2020). In contrast, we encode image sequences from embodied agents to obtain environment-level representations. Recent work also learns state representations via future prediction and implicit models (Ha & Schmidhuber, 2018; Eslami et al., 2018; Gregor et al., 2019; Hafner et al., 2019; Guo et al., 2020). In particular, neural rendering approaches achieve impressive reconstructions for arbitrary viewpoints (Eslami et al., 2018; Kumar et al., 2018). However, unlike our idea, they focus on pixelwise reconstruction, and their success has been limited to synthetically generated environments like DeepMind Lab (Beattie et al., 2016). In contrast to any of the above, we use egocentric videos to learn predictive feature encodings of photorealistic 3D environments to capture their naturally occurring regularities.

Scene completion: Past work in scene completion performs pixelwise reconstruction of 360 panoramas (Jayaraman & Grauman, 2018; Ramakrishnan et al., 2019), image inpainting (Pathak et al., 2016), voxelwise reconstructions of 3D structures and semantics (Song et al., 2017), and image-level extrapolation of depth and semantics (Song et al., 2018; Yang et al., 2019b). Recent work on visual navigation extrapolates maps of room-types (Wu et al., 2019; Narasimhan et al., 2020) and occupancy (Ramakrishnan et al., 2020a). While our approach is also motivated by anticipating unseen elements, we learn to extrapolate in a high-dimensional feature space (rather than pixels, voxels, or semantic categories) and in a self-supervised manner without relying on human annotations. Further, the proposed model learns from egocentric video sequences captured by other agents, without assuming access to detailed scans of the full 3D environment as in past work.

Learning image representations for navigation: Prior work exploits ImageNet pretraining (Gupta et al., 2017; Anderson et al., 2018a; Chen et al., 2019), mined object relations (Yang et al., 2019a), video (Chang et al., 2020), and annotated datasets from various image tasks (Sax et al., 2020; Chaplot et al., 2020c) to aid navigation. While these methods also consider representation learning in the context of navigation tasks, they are limited to learning image-level functions for classification and proximity prediction. In contrast, we learn predictive representations for sequences of observations conditioned on the camera poses.

3 APPROACH

We propose *environment predictive coding* (EPC) to learn self-supervised environment-level representations (Sec. 3.1). To demonstrate the utility of these representations, we integrate them into a transformer-based navigation architecture and refine them for individual tasks (Sec. 3.2). As we will show in Sec. 4, our approach leads to both better performance and better sample efficiency compared to existing approaches.

3.1 ENVIRONMENT PREDICTIVE CODING

Our hypothesis is that it is valuable for an embodied agent to learn a predictive coding of the environment. The agent must not just encode the individual views it observes, but also learn to leverage the encoded information to anticipate the unseen parts of the environment. Our key idea is that the environment embedding must be predictive of unobserved content, conditioned on the agent's camera pose. This equips an agent with the natural priors of 3D environments to quickly perform new tasks, like finding the refrigerator or covering more area.

We propose the proxy task of zone prediction to achieve this goal (see Fig. 2). For this task, we leverage egocentric video walkthroughs of other agents in various unseen environments (Fig. 2, top). For each video, we assume access to RGB-D, egomotion data, and camera intrinsics. Specifically, our current implementation uses egocentric camera trajectories from photorealistic scanned indoor environments (Gibson (Xia et al., 2018)) to sample the training videos; we leave leveraging in-the-wild consumer video as a challenge for future work.

We do *not* assume that the agents who generated those training videos were acting to address a particular navigation task. In particular, their behavior need not be tied to the downstream navigation-oriented tasks for which we test our learned representation. For example, a training video may show agents moving about to maximize their area coverage, whereas the encoder we learn is applicable to an array of navigation tasks (as we will demonstrate in Sec. 4). Furthermore, we assume that the environments seen in the videos are *not* accessible for interactive training. In practice, this means

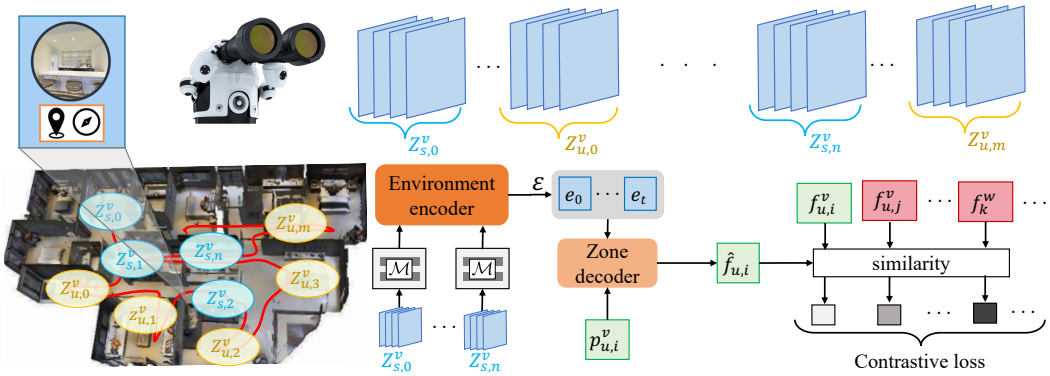

Figure 2: We propose the *zone prediction task* for self-supervised learning of environment embeddings from video walkthroughs generated by other agents. Each frame consists of the egocentric view and camera pose (top left). We group the frames in video v into seen zones in cyan $\{Z^v_{s,0}, \cdots, Z^v_{s,n}\}$ and unseen zones in yellow $\{Z^v_{u,0}, \cdots, Z^v_{u,m}\}$ (top row). The zones are generated automatically based on viewpoint overlap in 3D space (bottom left). Given a camera pose $p^v_{u,i}$ sampled from the unseen zone $Z^v_{u,i}$, we use a transformer-based encoder-decoder architecture that generates environment embeddings $\mathcal{E}$ from the seen zones, and predicts the feature encoding $\hat{f}_{u,i}$ of $Z^v_{u,i}$ conditioned on the pose $p^v_{u,i}$ (bottom center). The model is trained to distinguish the positive $f^v_{u,i}$ from negatives in the same video $\{f^v_{u,j}\}_{j \neq i}$ as well from other videos $\{f^w_k\}_{t \neq s}$ (bottom right).

that we can parallelly collect data from different robots deployed in a large number of environments, without having to actually train our navigation policy on those environments. These assumptions are much weaker than those made by prior work on imitation learning and behavioral cloning that rely on *task-specific* data generated from experts (Bojarski et al., 2016; Giusti et al., 2016).

Our method works as follows. First, we automatically segment videos into "zones" which contain frames with significant view overlaps. We then perform the self-supervised zone prediction task on the segmented videos. Finally, we incorporate the learned environment encoder into an array of downstream navigation-oriented tasks. We explain each step in detail next.

Zone generation At a glance, one might first consider masking arbitrary individual frames in the training videos. However, doing so is inadequate for representation learning, since unmasked frames having high viewpoint overlap with the masked frame can make its prediction trivial. Instead, our approach masks *zones* of frames at once. We define a zone to be a set of frames in the video which share a significant overlap in their viewpoints. We also require that the frames across multiple zones share little to no overlap.

To generate these zones, we first cluster frames in the videos based on the amount of pairwise-geometric overlap between views. We estimate the viewpoint overlap $\psi(o_i, o_j)$ between two frames o_i, o_j by measuring their intersection in 3D point clouds obtained by backprojecting depth inputs into 3D space. See Appendix for more details. For a video of length L, we generate a distance matrix $D \in \mathbb{R}^{L \times L}$ where $D_{i,j} = 1 - \psi(o_i, o_j)$. We then perform hierarchical agglomerative clustering (Lukasová, 1979) to cluster the video frames into zones based on D (see Fig. 2, bottom left). While these zones naturally tend to overlap near their edges, they typically capture disjoint sets of content in the video. Note that the zones segment *video trajectories*, not floorplan maps, since we do not assume access to the full 3D environment.

Zone prediction task Having segmented the video into zones, we next present our EPC zone prediction task to learn environment embeddings (see Fig. 2). We randomly divide the video v into seen zones $\{Z^v_{s,i}\}^n_{i=1}$ (cyan) and unseen zones $\{Z^v_{u,i}\}^m_{i=1}$ (yellow), where a zone Z is a tuple of images and the corresponding camera poses $Z_i = \{(o_j, p_j)\}^{|Z_i|}_1$. Given the seen zones, and the camera pose from an unseen zone $p^v_{u,i}$, we need to infer a feature encoding of the unseen zone $Z^v_{u,i}$. To perform this task, we first extract visual features x from each RGB-D frame o in the video using pretrained CNNs (see Sec. 3.2). These features are concatenated with the corresponding pose p and projected using an MLP $\mathcal{M}$ to obtain the image-level embedding. The target features for the unseen zone $Z^v_{u,i}$ are obtained as follows:

$$f^v_{u,i} = \frac{1}{|Z^v_{u,i}|} \sum_{[x,p] \in Z^v_{u,i}} \mathcal{M}([x, p]). \tag{1}$$

The rationale behind the feature averaging is that we want to predict the high-level visual content of the zone, while ignoring viewpoint specific variations within the zone.

We use a transformer-based encoder-decoder model to perform this task (Vaswani et al., 2017). Our model consists of an environment encoder and a zone decoder which infers the zone features (see Fig. 2, bottom). The environment encoder takes in the image-level embeddings $\mathcal{M}([x, p])$ from the input zones, and performs multi-headed self-attention to generate the environment embeddings $\mathcal{E}$. The zone decoder attends to $\mathcal{E}$ using the average camera pose from the unseen zone $p^v_{u,i}$ and predicts the zone features as follows:

$$\hat{f}_{u,i} = \text{ZoneDecoder}(\mathcal{E}, p^v_{u,i}). \tag{2}$$

We transform all poses in the input zones relative to $p^v_{u,i}$ before encoding, which provides the model an egocentric view of the world. The environment encoder, zone decoder, and the projection function $\mathcal{M}$ are jointly trained using noise-contrastive estimation (Gutmann & Hyvärinen, 2010). We use $\hat{f}_{u,i}$ as the anchor and $f^v_{u,i}$ from Eqn. 1 as the positive. We sample negatives from other unseen zones in the same video and from all zones in other videos. The loss for the ith unseen zone in video v is:

$$L^v_i = -\log \frac{\exp\big(\text{sim}(\hat{f}_{u,i}, f^v_{u,i})\big)}{\sum_{j=1}^{m} \exp\big(\text{sim}(\hat{f}_{u,i}, f^v_{u,j})\big) + \sum_{w \neq v, k} \exp\big(\text{sim}(\hat{f}_{u,i}, f^w_k)\big)}, \tag{3}$$

where $\text{sim}(q, k) = \frac{q \cdot k}{|q||k|} \frac{1}{\tau}$ and τ is a temperature hyperparameter. The idea is to predict zone representations that are closer to the ground truth, while being sufficiently different from the negative zones. Since the unseen zones have only limited overlap with the seen zones, the model needs to effectively reason about the geometric and semantic context in the seen zones to differentiate the positive from the negatives. We discourage the model from simply capturing video-specific textures and patterns by sampling negatives from within the same video.

3.2 Environment embeddings for embodied agents

Having introduced our approach to learn environment embeddings in a self-supervised fashion, we now briefly overview how these embeddings are used for agents performing navigation-oriented tasks. To this end, we integrate our pre-trained environment encoder into the Scene Memory Transformer (SMT) (Fang et al., 2019). Our choice of SMT is motivated by the recent successes of transformers in both NLP (Devlin et al., 2018) and vision (Sun et al., 2019b; Fang et al., 2019). However, our idea is potentially applicable to other forms of memory models as well.

We briefly overview the SMT architecture (see Fig. 3, center). It consists of a scene memory that stores visual features $\{x_i\}_{i=0}^t$ and agent poses $\{p_i\}_{i=0}^t$ generated from the observations seen during an episode. The environment encoder uses self-attention on the history of observations to generate a richer set of environment embeddings $\{e_i\}_{i=1}^t$. At a given time-step $t+1$, the policy decoder attends to the environment embeddings using the inputs o_{t+1}, which consist of the visual feature x and agent pose p at time $t+1$. The outputs of the policy decoder are used to sample an action a_{t+1} and estimate the value v_{t+1}. We detail each component in the Appendix.

To incorporate our EPC environment embeddings, we modify two key components from the original SMT model. First, and most importantly, we initialize the environment encoder with our pre-trained EPC (see Fig. 3, left). Second, we replace the end-to-end trained image encoders with MidLevel features that are known to be useful across a variety of embodied tasks (Sax et al., 2020) (see Fig. 3, right). We consider two visual modalities as inputs: RGB and depth. For RGB, we extract features from the pre-trained models in the max-coverage set proposed by Sax et al. (2020). These include surface normals, keypoints, semantic segmentation, and 2.5D segmentation. For depth, we extract features from pre-trained models that predict surface normals and keypoints from depth (Zamir et al., 2020). For training the model on a navigation task, we keep the visual features frozen, and only finetune the environment encoder, policy decoder, policy π, and value function V.

4 Experiments

We validate our pre-trained EPC environment embeddings for zone prediction (Sec. 4.1) and multiple downstream tasks that require an embodied agent to move intelligently through an unmapped environment (Sec. 4.2).

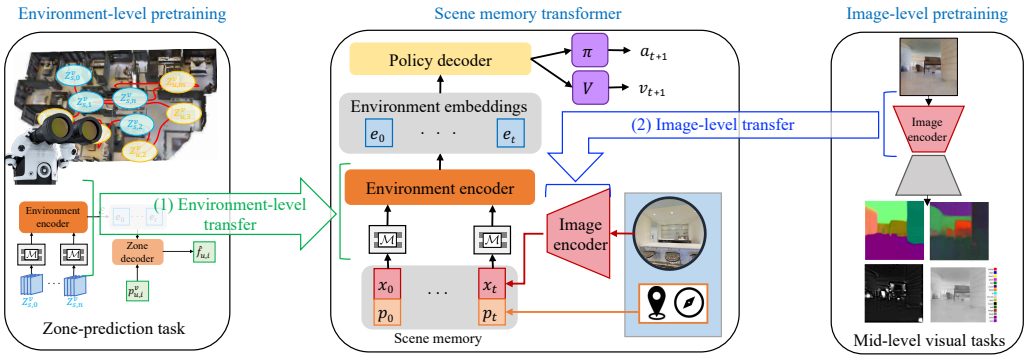

Figure 3: **Integrating environment-level pre-training for navigation: Left:** The first level of transfer occurs for the environment-level representations. We transfer the proposed EPC environment encoder and projection function $\mathcal{M}$ that are pre-trained for zone prediction. **Right:** The second level of transfer occurs for the image-level representations. We transfer a pre-trained MidLevel image encoder (Sax et al., 2020) to generate visual features for each input in the scene memory. **Center:** To train the SMT on a task, we keep the visual features frozen, and finetune the environment encoder and projection function $\mathcal{M}$ with the rest of the SMT model.

EXPERIMENTAL SETUP AND TASKS We perform experiments on the Habitat simulator (Savva et al., 2019b) with Matterport3D (MP3D) (Chang et al., 2017) and Gibson (Xia et al., 2018), two challenging and photorealistic 3D datasets with ~ 90 and 500 scanned real-world indoor environments, respectively. Our observation space consists of 171×128 RGB-D observations and odometry sensor readings that provide the relative agent pose $p = (x, y, \theta)$ w.r.t the agent pose at $t = 0$. Our action space consists of: MOVE-FORWARD by 25cm, TURN-LEFT by $30°$, and TURN-RIGHT by $30°$. For all methods, we assume noise-free actuation and odometry for simplicity.

We use MP3D for interactive RL training, and reserve Gibson for evaluation. We use the default train/val/test split for MP3D (Savva et al., 2019b) for 1000-step episodes. For Gibson, which has smaller environments, we evaluate on the 14 validation environments for 500-step episodes, and gather walkthrough trajectories for self-supervised learning from the 332 training environments. We generate the walkthroughs with an area-coverage agent trained on MP3D; the agent explores starting from multiple locations and gathers the RGB-D and odometer readings. This results in ~ 5000 videos, which we divide into an 80-20 train/val split. Following prior work (Ramakrishnan et al., 2020a; Chaplot et al., 2020b), we divide results on Gibson into small and large environments.

We evaluate our approach on three standard tasks from the literature:

1. **Area coverage** (Chen et al., 2019; Chaplot et al., 2020b; Ramakrishnan et al., 2020b): The agent is rewarded for maximizing the area covered (in m^2) within a fixed time budget.

2. **Flee** (Gordon et al., 2019): The agent is rewarded for maximizing the flee distance (in m), i.e., the geodesic distance between its starting location and the terminal location, for fixed-length episodes.

3. **Object coverage** (Fang et al., 2019; Ramakrishnan et al., 2020b): The agent is rewarded for maximizing the number of categories of objects covered during exploration (see Appendix). Since Gibson lacks extensive object annotations, we evaluate this task only on MP3D.

Together, these tasks capture different forms of geometric and semantic inference in 3D environments (e.g., area/object coverage encourage finding large open spaces/new objects, respectively).

BASELINES We compare to the following baselines:

Scratch baselines: We randomly initialize the visual encoders and policy and train them end-to-end for each task. Images are encoded using ResNet-18. Agent pose and past actions are encoded using FC layers. These are concatenated to obtain the features at each time step. We use three temporal aggregation schemes. *Reactive (scratch)* has no memory. *RNN (scratch)* uses a 2-layer LSTM as the temporal memory. *SMT (scratch)* uses a Scene Memory Transformer for aggregating observations (Fang et al., 2019).

SMT (MidLevel): extracts image features from pre-trained encoders that solve various mid-level perceptual tasks (Sax et al., 2020). This is an ablation of our model from Sec. 3.2 that uses the same image features, but randomly initializes the environment encoder. This SoTA image-level encoder is a critical baseline to show the impact of our proposed EPC environment-level encoder.

Method	4-2 split		2-4 split	
	w/ inputs	w/o inputs	w/ inputs	w/o inputs
Nearest neighbors	0.033	0.591	0.011	0.295
Random masking	0.332	0.760	0.209	0.449
EPC (bilinear)	0.452	0.806	0.321	0.518
EPC (ℓ_2-norm)	**0.528**	**0.825**	**0.368**	**0.556**

Table 1: **Zone prediction:** Top-1 zone prediction accuracy on the validation walkthrough videos.

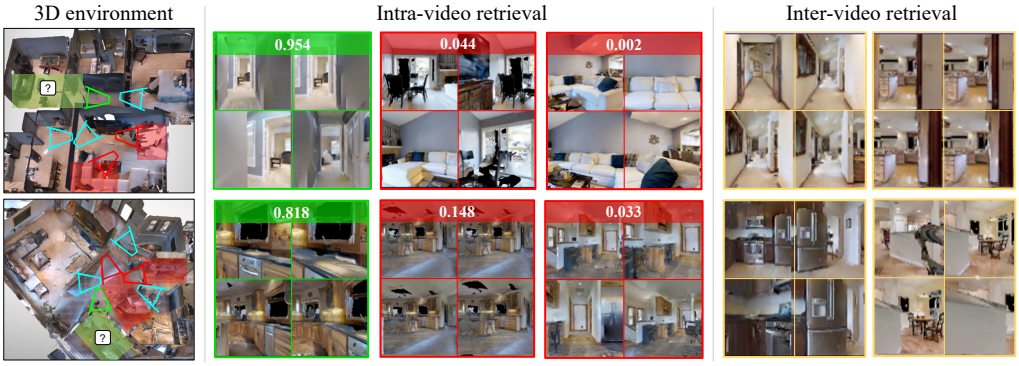

Figure 4: Each row shows one zone prediction example. **Left:** Top-down view of the 3D environment from which the video was sampled. The cyan viewing frusta correspond to the average pose for three input zones. Given the images and camera poses from each input zone, and a target camera pose (green frustum), the model predicts the corresponding zone feature (the masked green zone). **Center:** Given the inferred feature, we rank three masked (unobserved) zones from within the same video, where green is the positive zone and the red are the negatives. For each zone, we show four randomly sampled images along with the retrieval confidence. Our method retrieves the positive with high confidence. The model correctly predicts the existence of the narrow corridor (top row) and a kitchen counter (bottom row) given the target poses. **Right:** Top two retrieved zones from *other* videos that are closest to the inferred feature. The features predicted by the model are general enough to retrieve related concepts from other videos (narrow corridors and kitchens).

SMT (Video): Inspired by Dense Predictive Coding (Han et al., 2019), this baseline uses MidLevel features and pre-trains the environment encoder as a video-level model using the same training videos as our model. For pre-training, we randomly sample 25 consecutive frames as inputs and predict the average features corresponding to the next 15 frames. We query based on the time (not pose) and train the model using the NCE loss in Eqn. 3.

OccupancyMemory: This is the SoTA Active Neural SLAM model (Chaplot et al., 2020b) that maximizes area coverage, but using a state-of-the-art pointnav agent (Wijmans et al., 2020) instead of a planner for low-level navigation. It represents the environment as a top-down occupancy map.

All models are trained in PyTorch (Paszke et al., 2019) with DD-PPO (Wijmans et al., 2020) for 15M frames with 64 parallel processes and the Adam optimizer. See Appendix.

4.1 Zone prediction performance

First we evaluate the EPC embedding quality in terms of zone prediction on the validation videos. We divide each video into m seen and n unseen zones and infer the features for each unseen zone, given its average camera pose. We rank the features from the n unseen zones based on their similarity with the inferred feature, and measure the top-1 retrieval accuracy. We evaluate with $(m = 4, n = 2)$ and $(m = 2, n = 4)$ splits. The larger the value of m, the easier the task, since more information is available as input. We also test two simple baselines. **Nearest neighbors** uses the query pose to retrieve the 50 closest frames in the input zones, and outputs their averaged features. **Random masking** uses a different proxy task to learn the environment representations, randomly masking out 10 consecutive frames in the video and predicting their averaged feature from the rest. **EPC (bilinear)** uses bilinear product similarity (Oord et al., 2018) instead of the ℓ_2-norm below Eqn. 3. We report retrieval from only the unseen zones (w/o inputs) as well as the more challenging case where input zones are also candidates (w/inputs).

Method	Area coverage (m^2)			Flee (m)			Object coverage (#obj)	
	Gibson-S	Gibson-L	MP3D	Gibson-S	Gibson-L	MP3D	MP3D-cat.	MP3D-inst.
Reactive (scratch)	17.4 ± 0.2	22.8 ± 0.6	68.0 ± 1.3	1.9 ± 0.1	2.5 ± 0.3	5.1 ± 0.3	6.2 ± 0.0	19.0 ± 0.2
RNN (scratch)	20.6 ± 0.5	28.6 ± 0.3	79.1 ± 2.1	2.3 ± 0.2	2.8 ± 0.4	5.9 ± 0.1	6.1 ± 0.0	18.6 ± 0.2
SMT (scratch)	23.0 ± 0.7	32.3 ± 0.8	104.8 ± 2.3	3.3 ± 0.2	4.4 ± 0.4	6.9 ± 0.6	7.0 ± 0.1	23.2 ± 0.9
SMT (MidLevel)	29.1 ± 0.1	47.2 ± 1.7	155.7 ± 2.0	4.2 ± 0.0	6.0 ± 0.4	10.6 ± 0.3	7.6 ± 0.1	26.8 ± 0.6
SMT (Video)	28.7 ± 0.5	50.6 ± 2.6	129.7 ± 2.8	4.1 ± 0.0	5.0 ± 0.6	10.9 ± 0.5	7.3 ± 0.1	25.4 ± 1.0
OccupancyMemory	29.4 ± 0.1	$\mathbf{67.4} \pm 0.9$	155.6 ± 1.4	2.8 ± 0.0	7.0 ± 0.4	$\mathbf{14.1} \pm 0.6$	7.8 ± 0.1	27.8 ± 0.4
Ours (EPC)	$\mathbf{29.9} \pm 0.3$	56.4 ± 2.1	$\mathbf{165.6} \pm 2.8$	$\mathbf{4.5} \pm 0.1$	$\mathbf{7.1} \pm 0.4$	12.8 ± 0.6	$\mathbf{8.6} \pm 0.1$	$\mathbf{34.5} \pm 0.8$

Table 2: **Downstream task performance** at the end of the episode. Gibson-S/L means small/large. MP3D-cat./inst. means categories/instances. All methods are evaluated on three random seeds. See Appendix for performance vs. time step plots.

Figure 5: **Sample efficiency** on Matterport3D val split. Our environment-level pre-training leads to 2-4× training sample efficiency when compared to SoTA image-level pre-training. See Appendix for Gibson plots.

Tab. 1 shows the results. The EPC (ℓ_2-norm) model obtains superior retrieval performance on both settings. It retrieves the positive zones with high confidence (see Fig. 4 and Appendix). EPC's gain over random masking shows the value of the proposed zone generation step. Therefore, we select this model for downstream task transfer.

4.2 Downstream task performance

Now we transfer these features to downstream navigation tasks. Tab. 2 shows the results. On both datasets, we observe the following ordering:

$$\text{Reactive (scratch)} \;<\; \text{RNN (scratch)} \;<\; \text{SMT (scratch)}. \tag{4}$$

This is in line with results reported by Fang et al. (2019) and verifies our implementation of SMT. Using MidLevel features for SMT leads to significant gains in performance versus training image encoders from scratch.

Our environment-level pre-training provides substantial improvements compared to SMT (MidLevel), particularly for larger environments. Furthermore, SMT (Video)—the video-level pre-training strategy—often deteriorates performance compared to using only image-level pre-training. This highlights EPC's value in representing the underlying 3D spaces of the walkthroughs instead of treating them simply as video frames. EPC competes closely and even slightly outperforms the state-of-the-art OccupancyMemory on these tasks, with a significant gain on the object coverage metrics. Thus, our model competes strongly with a *task-specific* representation model on the tasks that the latter was designed for, while outperforming it significantly on other tasks.

Finally, Fig. 5 shows that EPC offers better sample efficiency than image-only pre-training: our method reaches the best performance of SMT (MidLevel) 2-4× faster. This confirms our hypothesis: transferring environment-level representations learned via contextual reasoning can help embodied agents learn faster compared to the current approach of transferring image-level encoders alone.

5 Conclusions

We introduced Environment Predictive Coding, a self-supervised approach to learn environment-level representations for embodied agents. By training on video walkthroughs generated by other agents, our model learns to infer missing content through a zone-prediction task. When transferred to multiple downstream embodied agent tasks, the resulting embeddings lead to better performance and sample-efficiency compared to the current practice of transferring only image-level representations. In future work, we plan to extend our idea for goal-driven tasks like PointNav and ObjectNav.

REFERENCES

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3674–3683, 2018a.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018b.

Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. Deepmind lab. *CoRR*, abs/1612.03801, 2016. URL `http://arxiv.org/abs/1612.03801`.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Angel Chang, Angela Dai, Tom Funkhouser, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2017. MatterPort3D dataset license available at: `http://kaldir.vc.in.tum.de/matterport/MP_TOS.pdf`.

Matthew Chang, Arjun Gupta, and Saurabh Gupta. Semantic visual navigation by watching youtube videos. *arXiv preprint arXiv:2006.10034*, 2020.

Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *arXiv preprint arXiv:2007.00643*, 2020a.

Devendra Singh Chaplot, Saurabh Gupta, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural mapping. *8th International Conference on Learning Representations, ICLR 2020*, 2020b.

Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12875–12884, 2020c.

Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Ricson Cheng, Ziyan Wang, and Katerina Fragkiadaki. Geometry-aware recurrent neural networks for active visual recognition. In *Advances in Neural Information Processing Systems*, pp. 5081–5091, 2018.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2054–2063, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 538–547, 2019.

Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3636–3645, 2017.

Chuang Gan, Yiwei Zhang, Jiajun Wu, Boqing Gong, and Joshua B Tenenbaum. Look, listen, and act: Towards audio-visual embodied navigation. *arXiv preprint arXiv:1912.11684*, 2019.

Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Juan P Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 2016.

Daniel Gordon, Abhishek Kadian, Devi Parikh, Judy Hoffman, and Dhruv Batra. Splitnet: Sim2sim and task2task transfer for embodied visual navigation. 2019.

Karol Gregor, Danilo Jimenez Rezende, Frederic Besse, Yan Wu, Hamza Merzic, and Aaron van den Oord. Shaping Belief States with Generative Environment Models for RL. 2019.

Daniel Guo, Bernardo Avila Pires, Bilal Piot, Jean-bastien Grill, Florent Altché, Rémi Munos, and Mohammad Gheshlaghi Azar. Bootstrap latent-predictive representations for multitask reinforcement learning. *arXiv preprint arXiv:2004.14646*, 2020.

Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2616–2625, 2017.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, 2010.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, pp. 2450–2462, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019.

Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Joao F Henriques and Andrea Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8476–8484, 2018.

Dinesh Jayaraman and Kristen Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. In *Computer Vision and Pattern Recognition, 2018 IEEE Conference on*, 2018.

Peter Karkus, Xiao Ma, David Hsu, Leslie Pack Kaelbling, Wee Sun Lee, and Tomás Lozano-Pérez. Differentiable algorithm networks for composable robot learning. *arXiv preprint arXiv:1905.11602*, 2019.

Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 8545–8552, 2019.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017.

Ananya Kumar, SM Eslami, Danilo J Rezende, Marta Garnelo, Fabio Viola, Edward Lockhart, and Murray Shanahan. Consistent generative query networks. *arXiv preprint arXiv:1807.02033*, 2018.

Xingyu Lin, Harjatin Baweja, George Kantor, and David Held. Adaptive auxiliary task weighting for reinforcement learning. 2019.

Alena Lukasová. Hierarchical agglomerative clustering procedure. *Pattern Recognition*, 11(5-6): 365–381, 1979.

Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J. Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, Dharshan Kumaran, and Raia Hadsell. Learning to navigate in complex environments. *CoRR*, abs/1611.03673, 2016. URL http://arxiv.org/abs/1611.03673.

Medhini Narasimhan, Erik Wijmans, Xinlei Chen, Trevor Darrell, Dhruv Batra, Devi Parikh, and Amanpreet Singh. Seeing the un-scene: Learning amodal semantic maps for room navigation. *arXiv preprint arXiv:2007.09841*, 2020.

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

Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *International Conference on Learning Representations*, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. 2016.

Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. Emergence of exploratory look-around behaviors through active observation completion. *Science Robotics*, 4(30), 2019. doi: 10.1126/scirobotics.aaw6326. URL https://robotics.sciencemag.org/content/4/30/eaaw6326.

Santhosh K. Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation, 2020a.

Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration, 2020b.

Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *International Conference on Learning Representations*, 2018.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019a.

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 *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9339–9347, 2019b.

Alexander Sax, Jeffrey O Zhang, Bradley Emi, Amir Zamir, Silvio Savarese, Leonidas Guibas, and Jitendra Malik. Learning to navigate using mid-level visual priors. In *Conference on Robot Learning*, pp. 791–812, 2020.

William B Shen, Danfei Xu, Yuke Zhu, Leonidas J Guibas, Li Fei-Fei, and Silvio Savarese. Situational fusion of visual representation for visual navigation. 2019.

Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Shuran Song, Andy Zeng, Angel X Chang, Manolis Savva, Silvio Savarese, and Thomas Funkhouser. Im2pano3d: Extrapolating 360 structure and semantics beyond the field of view. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3847–3856, 2018.

Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning video representations using contrastive bidirectional transformer. *arXiv preprint arXiv:1906.05743*, 2019a.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. VideoBert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7464–7473, 2019b.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8052–8060, 2018.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. 2020.

Yi Wu, Yuxin Wu, Aviv Tamar, Stuart Russell, Georgia Gkioxari, and Yuandong Tian. Bayesian relational memory for semantic visual navigation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018. Gibson dataset license agreement available at `https://storage.googleapis.com/gibson_material/Agreement%20GDS%2006-04-18.pdf`.

Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2): 713–720, 2020.

Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors, 2019a. URL `https://openreview.net/forum?id=HJeRkh05Km`.

Zhenpei Yang, Jeffrey Z. Pan, Linjie Luo, Xiaowei Zhou, Kristen Grauman, and Qixing Huang. Extreme relative pose estimation for rgb-d scans via scene completion. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019b.

Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. Auxiliary tasks speed up learning pointgoal navigation. *arXiv preprint arXiv:2007.04561*, 2020.

Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11197–11206, 2020.

Appendix

A ZONE GENERATION

As discussed in the main paper, we generate zones by first clustering frames in the video based on their geometric overlap. Here, we provide details on how this overlap is estimated. First, we project pixels in the image to 3D point-clouds using the camera intrinsics and the agent pose. Let D_i, p_i be the depth map and agent pose for frame i in the video. The agent's pose in frame i can be expressed as $p_i = (\boldsymbol{R}_i, \boldsymbol{t}_i)$, with $\boldsymbol{R}_i, \boldsymbol{t}_i$ representing the *agent's* camera rotation and translation in the world coordinates. Let $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ be the intrinsic camera matrix, which is assumed to be provided for each video. We then project each pixel x_{ij} in the depth map D_i to the 3D point cloud as follows:

$$w_{ij} = \begin{bmatrix} \boldsymbol{R}_i & \boldsymbol{t}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{K}^{-1} x_{ij}, \ \forall j \in \{1, ..., S_i\} \tag{5}$$

where S_i is the total number of pixels in D_i. By doing this operation for each pixel, we can obtain the point-cloud $\boldsymbol{W}_i$ corresponding to the depth map D_i. To compute the geometric overlap between two frames i and j, we estimate the overlap in their point-clouds $\boldsymbol{W}_i$ and $\boldsymbol{W}_j$. Specifically, for each point $w_i \in \boldsymbol{W}_i$, we retrieve the nearest neighbor from $w_j \in \boldsymbol{W}_j$ and check whether the pairwise distance in 3D space is within a threshold τ: $||w_i - w_j||_2 < \tau$. If this condition is satisfied, then a match exists for w_i. Then, we define the overlap fraction $\psi(D_i, D_j)$ the fraction of points in $\boldsymbol{W}_i$ which have a match in $\boldsymbol{W}_j$. This overlap fraction is computed pairwise between all frames in the video, and hierarchical agglomerative clustering is performed using this similarity measure.

B TASK DETAILS

For the object coverage task, to determine if an object is covered, we check if it is within $3\mathrm{m}$ of the agent, present in the agent's field of view, and if it is not occluded (Ramakrishnan et al., 2020b). We use a shaped reward function:

$$R_t = O_t - O_{t-1} + 0.02(C_t - C_{t-1}), \tag{6}$$

where O_t, C_t are the number of object categories and 2D grid-cells visited by time t (similar to Fang et al. (2019)).

C SCENE MEMORY TRANSFORMER

We provide more details about individual components of the Scene Memory Transformer Fang et al. (2019). As discussed in the main paper, the SMT model consists of a scene memory for storing the visual features $\{x_i\}_{i=0}^t$ and agent poses $\{p_i\}_{i=0}^t$ seen during an episode. The environment encoder uses self-attention on the scene memory to generate a richer set of environment embeddings $\{e_i\}_{i=1}^t$. The policy decoder attends to the environment embeddings using the inputs o_{t+1}, which consist of the visual feature x, and agent pose p at time $t+1$. The outputs of the policy decoder are used to sample an action a_{t+1} and estimate the value v_{t+1}. Next, we discuss the details of the individual components.

SCENE MEMORY It stores the visual features derived from the input images and the agent poses at each time-step. Motivated by the ideas from Sax et al. (2020), we use mid-level features derived from various pre-trained CNNs for each input modality. In this work, we consider two input modalities:

RGB, and depth. For RGB inputs, we extract features from the pre-trained models in the max-coverage set proposed in Sax et al. (2020). These include surface normals, keypoints, semantic segmentation, and 2.5D segmentation. For depth inputs, we extract features from pre-trained models that predict surface normals and keypoints from depth (Zamir et al., 2020). For simplicity, we assume that the ground-truth pose is available to the agent in the form of $(x_t, y_t, z_t, \theta_t)$ at each time-step, where θ_t is the agent heading. While this can be relaxed by following ideas from state-of-the-art approaches to Neural SLAM (Chaplot et al., 2020b; Ramakrishnan et al., 2020a), we reserve this for future work as it is orthogonal to our primary contributions.

ATTENTION MECHANISM Following the notations from Vaswani et al. (2017), we define the attention mechanism used in the environment encoder and policy decoder. Given two inputs $X \in \mathbb{R}^{n_1 \times d_x}$ and $Y \in \mathbb{R}^{n_2 \times d_y}$, the attention mechanism attends to Y using X as follows:

$$\text{Attn}(X, Y) = \text{softmax}\left(\frac{Q_X K_Y^T}{\sqrt{d_k}}\right) V_Y \tag{7}$$

where $Q_X \in \mathbb{R}^{n_1 \times d_k}, K_Y \in \mathbb{R}^{n_2 \times d_k}, V_Y \in \mathbb{R}^{n_2 \times d_v}$ are the queries, keys, and values computed from X and Y as follows: $Q_X = XW^q$, $K_Y = YW^k$, and $V_Y = YW^v$. W^q, W^k, W^v are learned weight matrices. The multi-headed version of Attn generates multiple sets of queries, keys, and values to obtain the attended context $C \in \mathbb{R}^{n_1 \times d_v}$.

$$\text{MHAttn}(X, Y) = \text{FC}([\text{Attn}^h(X, Y)]_{h=1}^H). \tag{8}$$

We use the transformer implementation from PyTorch (Paszke et al., 2019). Here, the multi-headed attention block builds on top of MHAttn by using residual connections, LayerNorm (LN) and fully connected (FC) layers to further encode the inputs.

$$\text{MHAttnBlock}(X, Y) = \text{LN}(\text{MLP}(H) + H) \tag{9}$$

where $H = \text{LN}(\text{MHAttn}(X, Y) + X)$, and MLP has 2 FC layers with ReLU activations. The environment encoder performs self-attention between the features stored in the scene memory to obtain the environment encoding E.

$$E = \text{EnvironmentEncoder}(M) = \text{MHAttnBlock}(M, M). \tag{10}$$

The policy decoder attends to the environment encodings E using the current observation x_t, p_t.

$$\text{PolicyDecoder}([x_t, p_t], E) = \text{MHAttnBlock}(\text{FC}([x_t, p_t]), E) \tag{11}$$

We transform the pose vectors $\{p_i\}_{i=1}^n$ from the scene memory relative to the current agent pose p_t as this allows the agent to maintain an egocentric view of past inputs Fang et al. (2019).

D HYPERPARAMETERS

We detail the list of hyperparameter choices for different tasks and models in Tab. 3. For the random masking baseline in Tab. 1, we tried masking out 10, 20 and 50 frames and picked 10 frames based on the zone prediction performance. For SMT (Video), we choose 25 frames as inputs and 15 frames as output based on Dense Predictive Coding Han et al. (2019).

RL Optimization	
Optimizer	Adam
Learning rate	0.00025 - 0.001
# parallel actors	64
PPO mini-batches	2
PPO epochs	2
PPO clip param	0.2
Value loss coefficient	0.5
Entropy coefficient	0.01
Advantage estimation	GAE
Normalized advantage?	Yes
Training episode length	1000
GRU history length	128
# training steps (in millions)	15
RNN hyperparameters	
Hidden size	128
RNN type	LSTM
Num recurrent layers	2
SMT hyperparameters	
Hidden size	128
Scene memory length	500
# attention heads	8
# encoder layers	1
# decoder layers	1
Occupancy memory hyperparameters	
Action space range	$48\text{m} \times 48\text{m}$
# global action sampling interval	25
Reward scaling factors for different tasks	
Task	Reward scale
Area coverage	0.3
Flee	1.0
Object coverage	1.0
Self-supervised learning optimization	
Optimizer	Adam
Learning rate	0.0001
Video batch size	20
Temperature (τ)	0.1

Table 3: Hyperparameters for training our RL and self-supervised learning models.

E DOWNSTREAM TASK PERFORMANCE VS. EPISODE TIME

We show the downstream task performance as a function of time in Fig. 6. We evaluate each model with 3 different random seeds and report the mean and the 95% confidence interval in the plots.

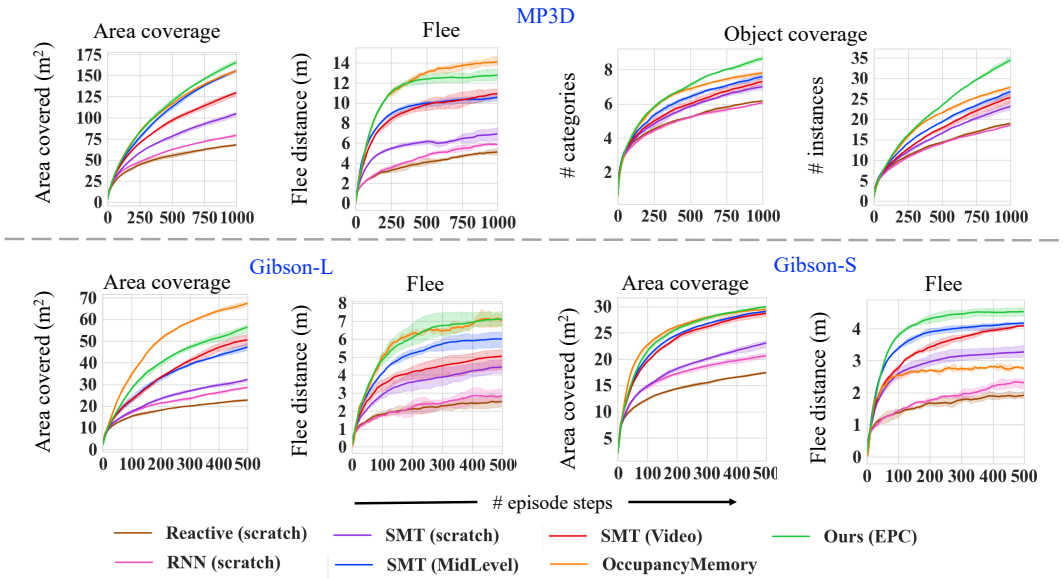

Figure 6: We highlight the downstream task performance as a function of episode time on both Matterport3D and Gibson.

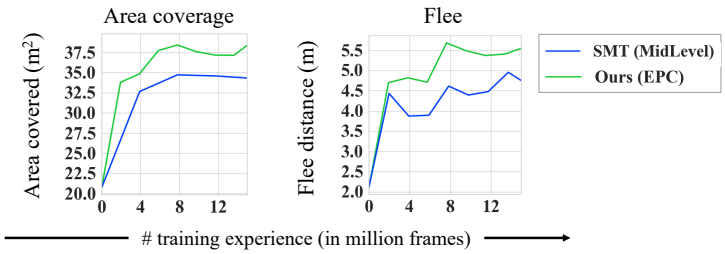

Figure 7: **Sample efficiency** on Gibson val split. Our environment-level pre-training leads to 2-4× training sample efficiency when compared to SoTA image-level pre-training.

F SAMPLE EFFICIENCY CURVES ON GIBSON

We plot the Gibson validation performance as a function of training experience in Fig. 7. EPC achieves better sample efficiency through environment-level pre-training when compared to the image-level pre-training baseline SMT (MidLevel).