Appendix: Learning Navigational Visual Representations with Semantic Map Supervision

Yicong Hong[12] Yang Zhou[1] Ruiyi Zhang[1]
Franck Dernoncourt[1] Trung Bui[1] Stephen Gould[2] Hao Tan[1]
[1]Adobe Research [2]The Australian National University
mr.yiconghong@gmail.com
stephen.gould@anu.edu.au, {yazhou,ruizhang,dernonco,bui,hatan}@adobe.com
Project URL: https://github.com/YicongHong/Ego2Map-NaViT

A. Data Sampling

We sample images and create top-down semantic maps for our Ego^2-Map pre-training from the large-scale Habitat-Matterport3D environments (HM3D) [3, 4]. HM3D provides 1,000 high-fidelity reconstructions of entire buildings containing indoor spaces at diverse geographical locations and physical sizes, with a total traversable area of 112.5k m^2. HM3D environments are divided into 800 training, 100 validation, and 100 testing scenes. In our work, we sample data from the training and validation scenes for pre-training and evaluation, respectively, and the sampling details will be provided in this section.

A.1. Viewpoint Sampling

We sample viewpoints from the environments using a virtual agent in the Habitat simulator [4]. We initialize the agent such that its physical dimensions match the standard configurations in R2R-CE [2] (0.10 m radius and 1.50 m height with a camera pointing horizontally at 1.25 m height). The setting of physical dimensions determines the virtual agent's navigable space, which is also the open space that we applied to sample the viewpoints.

We create a heuristic to control the sampling process; First, for each scene, we measure the size of the total navigable area $\mathcal{S}$ using the `sim.pathfinder.navigable_area()` function[1]. Then, we apply `sim.sample_navigable_point()` to randomly sample a viewpoint p in the open space. Each p needs to satisfy three criteria otherwise discarded, checking by three functions: (1) `sim.sample_navigable_point(`p`)`: its position should be reachable from all other open regions, (2) `sim.island_radius(`p`)` $<$1.50: the point should not be sampled in narrow spaces, and (3) `sim.geodesic_distance(`p_i`,` p_j`)` $<$0.40: the minimal distance between any two viewpoints should be greater than 0.40 meters. For each scene, we repeat the above procedure to sample $4\times\mathcal{S}$ or 500 viewpoints[2], whichever is smaller.

A.2. Image Sampling

For each valid viewpoint, we use the camera to capture four RGBD images with a resolution of 320$\times$320 and a horizontal field of view of 90°, where the range of depth sensor is $[0.5, 5.0]$ meters. The views are sampled at random orientations, and 2 pairs of views are created for measuring the angular offset θ, while using all views to predict the maximal explorable distance d. As mentioned in the *Main Paper* §3.3, the ground-truth angular offset is defined in range $[-\pi, \pi]$, denoting either clockwise (negative) or counter-clockwise (positive) rotation whichever is smaller in absolute value. The intuition behind such design is to encourage the model to learn the most efficient rotation between two orientations. As for the exploration distance prediction, we collect the ground-truth distances by asking the agent to move forward 5.0 meters in each view direction with a small step size of 0.10 meters, and use the `sim.previous_step_collided()` to detect collision.

A.3. Path Sampling

Path for connecting two distant views is sampled for creating the top-down semantic map for Ego^2-Map contrastive learning. In specific, we consider each viewpoint as a source p_s and randomly select a target viewpoint p_t within 7.0 meters geodesic distance radius. As described

[1]All functions correspond to the functions defined in the Habitat Simulator, please refer to their detailed definitions in the official codebase: https://github.com/facebookresearch/habitat-sim.

[2]A maximum is necessary because HM3D contains a few giant scans with repetitive spaces (*e.g.* hotel rooms), which only offer very little image diversity so that are inappropriate for training.

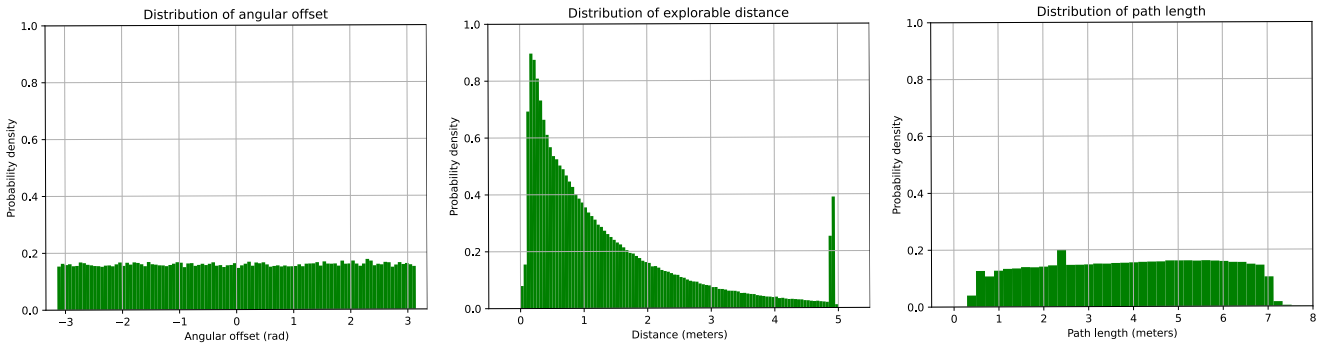

Figure 1: Distribution of the sampled data. Left: angular offset for views pairs. Middle: explorable distance. Right: path length between source and target viewpoints (computed by the `ShortestPathFollower()` function).

in *Main Paper* §3.2, each viewpoint in Ego^2-Map is represented by a single egocentric view, which we directly use one of the 4 sampled images for this purpose. Then, we apply the `ShortestPathFollower`(p_s, θ_s, p_t) to compute the shortest path (and actions) for traveling from p_s to p_t[3], where θ_s is the agent's orientation at the source viewpoint. Our agent uses a small turning angle of 5° and a small forward step of 0.10 meters, so that more fine-grained and accurate paths can be created. The shortest path is found if the agent's final position is within 0.50 meters of geodesic distance to the target and the total number of actions is less than 140 steps, otherwise, a new target viewpoint will be paired for examination. Note that, we control each view to be either a unique source or target view in all trajectories to avoid repetitive use of the same image in Ego^2-Map. Finally, `ShortestPathFollower()` returns the agent's position, orientation and action at each time step, which are applied for generating the semantic map.

A.4. Generating Semantic Maps

We apply the Semantic MapNet (SMNet) [1] to generate top-down semantic maps from the sampled paths. Briefly, SMNet takes a sequence of the agent's egocentric RGBD observations and poses as input; for each step, the network encodes the RGBD frame and projects it to a floor plan. Then, a spatial memory tensor accumulates the projected egocentric features from all steps. Finally, a map decoder produces the allocentric semantic map from the aggregated memory tensor. We refer the readers to the paper and code[4] of the SMNet for more details.

In this work, we slightly shift the map so that the agent's starting position p_s is at the center of the map. We also scale the map such that it covers a $[-6, 6]$ meters range (slightly less than the sampling radius of p_t). To avoid rare cases where little semantic information is captured at the target position, we augment the paths by adding a 360° rotation at p_t. Moreover, we highlight the agent's transition by drawing the path as a gradient color line on the map. Note that the pre-trained SMNet applies a different resolution and field of view for the egocentric images than the views in our image sampling process, we follow SMNet's default configurations to sample the RGBD images on the path.

A.5. Creating Dataset

By following the aforementioned procedure, 252,537 viewpoints are sampled from the 800 training environments, from which 500,000 $(I_{\theta_0}, I_{\theta_1})$ views pairs as well as 500,000 (I_s, I_t, M) triplets are created for learning $\mathcal{L}_\theta$ and $\mathcal{L}_c$, respectively. Figure 1 shows the distribution of the sampled angular offsets (left), explorable distances (middle) and the distance between source and target viewpoints (right). Note that the maximal geodesic distance for sampling (I_s, I_t) is 7.0 meters, but the graph displays the actual path length returned by the `ShortestPathFollower()` function, hence a few paths are longer than 7.0 meters. Overall, we can see that the sampled angular offset and the path length are roughly uniform in the sampling range, whereas the majority of the explorable distances are in the $[0.2, 2.0]$ meters range due to the structure of indoor spaces.

We employ the WebDataset library[5] for efficient storing and loading data. Specifically, we create 1,000 shards each contains 500 data points for training, where each data point includes a $(I_{\theta_0}, I_{\theta_1})$ views pair and a (I_s, I_t, M) triplet. Following the same procedure, we also create 20 shards for validating the pre-training objectives (see Appendix §B).

B. Pre-Training Statistics and Results

We present the loss curves and the validation results of the pre-training variants corresponding to Table 1 in the *Main Paper*. As shown in Appendix Figure 2 and Table 1,

[3] More precisely, from the source view I_s to p_t since the function does not specify a target orientation.

[4] SMNet: https://github.com/vincentcartillier/Semantic-MapNet.

[5] WebDataset: https://github.com/webdataset/webdataset.

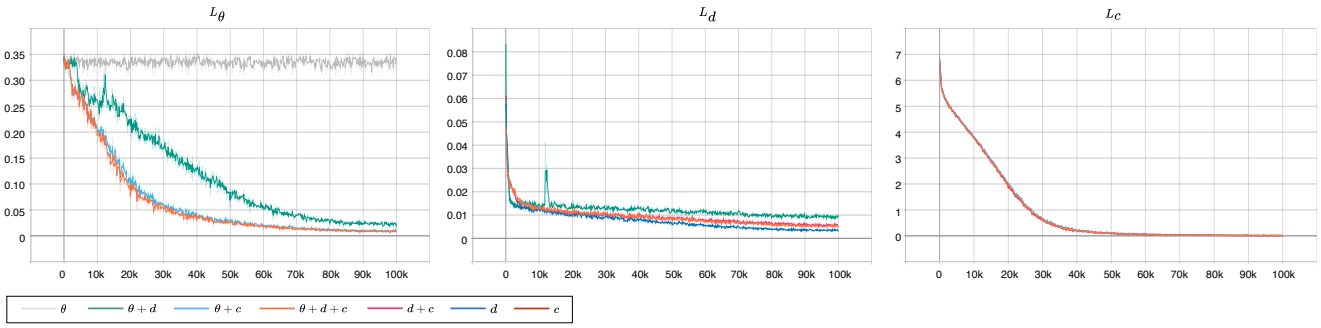

Figure 2: Loss curves of the pre-training objectives.

Model #	Pre-Training Objectives			Error / Accuracy			
	Angular	Explorable	Contrastive	$\Delta\theta$ (rad)↓	Δd (m)↓	I→M(%)↑	M→I(%)↑
1	✓			1.564	–	–	–
2		✓		–	0.198	–	–
3			✓	–	–	92.01	92.12
4		✓	✓	–	0.239	91.79	91.70
5	✓		✓	0.810	–	91.53	91.63
6	✓	✓		0.804	0.262	–	–
7	✓	✓	✓	0.819	0.256	92.02	92.03

Table 1: Validation of the pre-training tasks. $\Delta\theta$ and Δd denote the averaged discrepancy between the predicted angular offset / explorable distance and their ground-truths, I→M and M→I are the alignment accuracy from views-pair to maps and from map to views-pairs (evaluated with batch size 128 on 10,000 novel (I_s, I_t, M) triplets). *Models* correspond to Table 1 in the *Main Paper*.

the loss $\mathcal{L}_\theta$ of the angular offset prediction does not converge to the same level as the others when minimized alone, showing large error in the prediction and leading to invalid features for the downstream navigation tasks. However, it is interesting to see that $\mathcal{L}_\theta$ can be learned in the presence of $\mathcal{L}_d$ or $\mathcal{L}_c$. One possible reason is that the model can extract valuable spatial information by predicting distance or aligning views and maps to facilitate the learning of angular relationships between views. In Model#5, Model#6, and Model#7, the predicted angular error is around 0.810 rad (46°), again suggesting the difficulty of learning angular offset from views. Besides, we can see that the learning of $\mathcal{L}_d$ only shows a minor difference across the model variants, and the prediction error is very low (up to 26.2 centimeters), which indicates that the learned features contain accurate information of whether a direction is explorable. As for the Ego^2-Map contrastive learning, the loss curves in training smoothly converge to zero. We evaluate the alignment accuracy with a batch size of 128 on 10,000 novel (I_s, I_t, M) triplets from the validation split; results show that both the views-pair to maps (I→M) and map to views-pairs (M→I) matching are highly accurate, suggesting the transfer of map information to egocentric representations.

References

[1] Vincent Cartillier, Zhile Ren, Neha Jain, Stefan Lee, Irfan Essa, and Dhruv Batra. Semantic mapnet: Building allocentric semantic maps and representations from egocentric views. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 964–972, 2021. 2

[2] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, 2020. 1

[3] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 1

[4] 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/CVF International Conference on Computer Vision*, pages 9339–9347, 2019. 1