CS231A Final Report
Embodiment for 3D Representation Learning

Guanzhi Wang
Stanford University
guanzhi@stanford.edu

Abstract

The field of computer vision has witnessed tremendous success in the past few years. Since the introduction of AlexNet in 2012, the paradigm of supervised deep learning has significantly improved classical tasks like object classification, semantic segmentation and object detection. However, a neural network learner processes millions of disconnected, manually annotated images or videos to discover the statistical association between patterns and labels, while being completely oblivious to the physical body and movement that generate those data. In this work, we study how embodiment helps improve 3D representation learning of an agent. We train a policy network for PointGoal navigation tasks in iGibson environments and design a spectrum of 3D visual tasks to evaluate the learned skills of the agent. Extensive experiment results show embodiment helps the agent acquire sophisticated abilities by solving navigation tasks using deep reinforcement learning.

1. Introduction

Visual recognition has achieved tremendous success, powered by a collection of large-scale curated datasets [3, 21, 2]. Despite the high scores on the leaderboards, modern computer vision systems are still a far cry from the human vision system. Deep neural networks require millions of disconnected and manually annotated images, while being oblivious to the physical world that generates those data. A human toddler, however, is far more efficient in mastering novel object concepts, navigating complex spatial layouts, and learning sophisticated hand-eye coordination skills. A real-world robot, like a human toddler, actively obtains visual observations from the physical environment. It can acquire sophisticated abilities by tackling navigation [40, 11, 10] and recognition [16, 38] tasks via deep reinforcement learning (RL). How such embodiment affects the learned abilities of the agent is an open-ended problem.

In this work, we take a step towards closing the gap between AI and human intelligence by studying how embodiment affects 3D representation learning. We use iGibson [36, 30], a simulation environment for robotic tasks in large realistic scenes as a way of embodiment. More specifically, we build a distributed RL framework and train a policy network for PointGoal tasks [1], where a mobile robot (TurtleBot) is randomly spawned in an environment and it needs to navigate to a specified goal location using RGB visual signals. After we train the policy network, we finetune and evaluate the convolutional encoder of the policy network on different downstream tasks, including relative camera pose estimation, depth estimation, surface normal estimation, optical flow prediction and scene flow prediction. All downstream tasks are 3D visual tasks, which are specifically designed to evaluate the 3D representation of the scene learned by the agent. An overview of our leaning framework is shown in Figure 1.

2. Related Work

Representation Learning for Embodiment. Prior work uses self-supervised learning techniques to improve the performance of embodied learning. Image representation tasks leverage proxy tasks such as instance discrimination [33] and inpainting [25]. Video representation tasks utilize contrastive predictions [15, 31] and temporal consistency [35]. Reinforcement learning algorithms often suffer from high variance and sample inefficiency and therefore prior work uses self-supervised learning techniques to address these challenges. "curiosity" [24], an intrinsic reward which uses the prediction-error between the learned model and environment behavior, has been widely adopted by a large family of approaches for training the agents. Recent work also uses future prediction and implicit models to learn state representations [12, 9]. A more recent work, environment predictive coding [27] learns environment-level representations for embodied agents. Different from prior work, our work mainly focuses on studying how embodiment affects representation learning. Our main goal is not

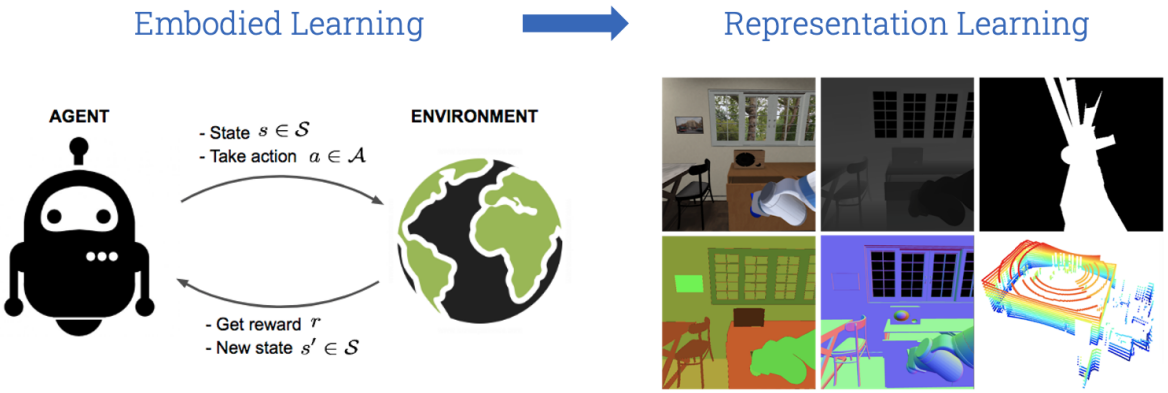

Figure 1: An overview of our learning framework. The first stage is embodied learning where we train an embodied agent in an environment to solve a robotic task. The second stage is representation learning where we finetune the policy network learned by the agent with a spectrum of downstream tasks. In this way, we will know how embodiment affects representation learning.

to improve the performance of embodied learning, but instead, we want to understand what skills are learned by the embodied agent and how to evaluate the learned 3D representations.

Multi-Task Learning. Prior work aims at developing systems that can provide multiple outputs for an input in one run [18, 4]. MTAN [22] proposes a multi-task learning architecture, which allows learning of task-specific feature-level attention. Their proposed method predicts semantic segmentation, depth and surface normal at the same time. Taskonomy [39] explicitly model the relations among tasks and extract a meta-structure. They designed a spectrum of downstream tasks for modeling the structure of space of visual tasks. In this work, we do not do multi-task learning, but we do bear the multi-task setup in mind because we want to finetune the encoder of our policy network for different downstream tasks. Inspired by [17] which unifies the tasks of instance segmentation (for thing classes) and semantic segmentation (for stuff classes), we use Feature Pyramid Network (FPN) as our backbone.

Deep Reinforcement Learning in Robotics. Deep RL has been applied to mobile robot navigation [41] and robot arm manipulation [19, 37]. There are both model-based and model-free RL approaches that are widely used. Model-based methods [19, 37] are sample efficient but do not generalize well due to their strong model assumptions. Model-free methods [42] often require a large amount of data but are more flexible. We specifically use Soft Actor-Critic [13], a model-free method to tackle PointGoal navigation tasks. We build a distributed and scalable RL framework on top of SURREAL [7] and get very good results of training embodied agents.

3. Approach

Given a policy network trained for the PointGoal navigation task in a cluttered environment, where a robot needs to navigate from an initial position to a target position and avoid objects and obstacles in the environment, we extract the visual encoder of the policy network and finetune it on a spectrum of downstream tasks to see how embodiment affects 3D representation learning of the agent by comparing it with an encoder that is trained from scratch on the downstream tasks.

We expect that the finetuned encoder is better than the encoder trained from scratch. In order for the agent to avoid collisions and successfully navigate to the goal location, it needs to understand many 3D concepts, e.g., which object is closer, will any obstacle appear if following the current trajectory, etc. Therefore, the navigation task will benefit the agent in understanding the 3D environment and thus make it obtain a good pretrained 3D representation, which is useful for downstream finetuning.

3.1. Distributed RL Framework

Since iGibson training required GPU for both heavy graphics simulation and neural network policy training, existing reinforcement RL libraries either did not scale well or could not be easily customized. Therefore, we design our own researcher-friendly, and highly scalable distributed RL framework (Figure 2), that can be used to train many agents in parallel. And it can scale to hundreds of CPUS and dozens of GPUs. This is a major upgrade in speed and customizability from SURREAL [7]. We adopt a data parallel paradigm. Every GPU runs multiple instances of iGibson simulators in parallel. Each GPU and each simulator instance can actually run on different 3D scanned rooms

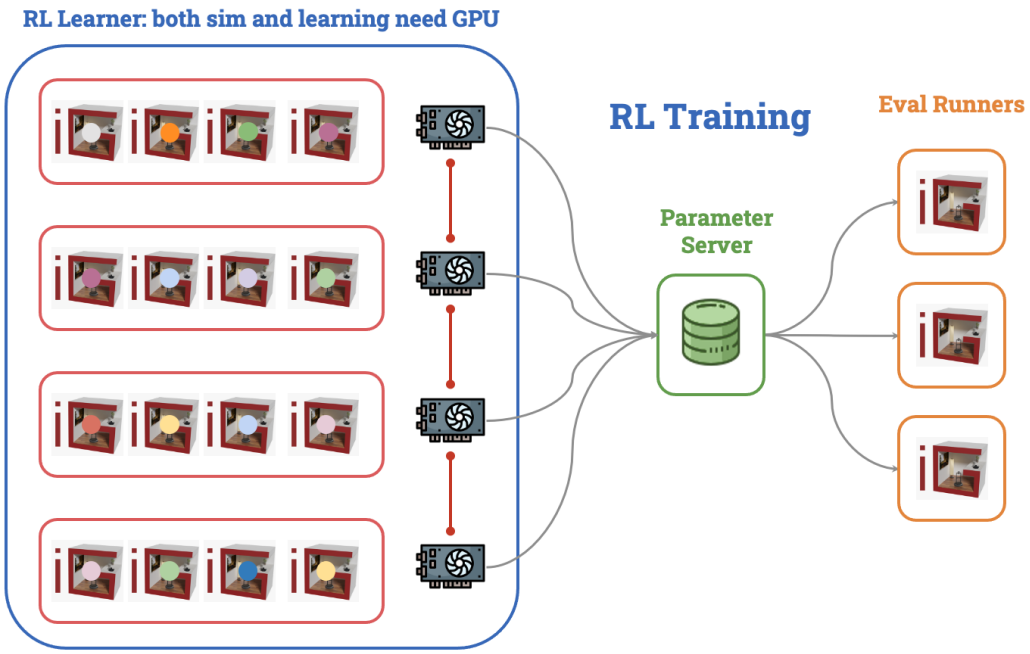

Figure 2: Our distributed RL framework. We adopt a data parallel paradigm. Every GPU runs multiple instances of iGibson simulators in parallel. Each GPU and each simulator instance can actually run on different 3D scanned rooms with different furnitures.

with different furnitures. Each agent collects experience in parallel and the learning takes place in a synchronized data parallel fashion. We have a parameter server that can periodically pushes the latest parameters to a bunch of evaluation agents, which evaluate the performance in real time.

3.2. Policy Network

The foundation of RL is Markov Decision Process [32], defined as the tuple $(\mathcal{S}, \mathcal{A}, P, \gamma)$. The components are states $s \in \mathcal{S} = \mathbb{R}^n$, actions $a \in \mathcal{A}$, and state transition probability function, $P = P(s_{t+1}, r_t | s_t, a_t)$. An RL agent seeks to maximize the expected reward, $R = \sum_{t=0}^{\infty} \gamma^t r_t$ with discount factor $\gamma \in [0, 1)$. In this work, we mainly consider continuous control tasks from raw pixels. The agent receives a high-dimensional image observation $o_t = O(s_t) \in \mathbb{R}^k$, an indirect representation of the state, and outputs a continuous action $\mathcal{A} \in \mathbb{R}^m$.

In this work, we use Soft Actor-Critic (SAC) as our policy network. SAC [13, 14] is a state-of-the-art off-policy RL algorithm for continuous control. SAC learns a policy $\pi(a|o)$ and a critic $Q(o, a)$ that maximize a weighted objective of expected reward and policy entropy, $\mathbb{E}_{s_t, a_t \sim \pi} [\sum_t r_t + \alpha \mathcal{H}(\pi(\cdot|o_t))]$. SAC stores experiences into a replay buffer $\mathcal{D}$. The critic parameters are updated by minimizing the Bellman error using transitions sampled from $\mathcal{D}$:

$$\mathcal{L}_Q = \mathbb{E}_{\tau \sim \mathcal{D}} \left[\left(Q(o_t, a_t) - (r_t + \gamma V(o_{t+1})) \right)^2 \right]. \quad (1)$$

The target value of next state is estimated by sampling an action under the current policy:

$$V(o_{t+1}) = \mathbb{E}_{a' \sim \pi} \left[\tilde{Q}(o_{t+1}, a') - \alpha \log \pi(a'|o_{t+1}) \right], \quad (2)$$

where $\tilde{Q}$ denotes an exponential moving average of the critic parameters. The policy is learned by minimizing the divergence from the exponential of the soft-Q function:

$$\mathcal{L}_\pi = -\mathbb{E}_{a \sim \pi} \left[Q(o_t, a) - \alpha \log \pi(a|o_t) \right], \quad (3)$$

where α is a learnable temperature parameter that controls the stochasticity of the optimal policy.

3.3. Visual Encoder

In the PointGoal navigation task, observations of the agent include RGB visual signals, the robot's linear and angular velocities, the goal location in the robot's reference frame, and the next 10 waypoint locations following the shortest path between the robot's current location and the goal location. We concatenate all signals except for the RGB signals and denote them by sensor signals. We use a frame stack of 4 for both RGB and sensor signals.

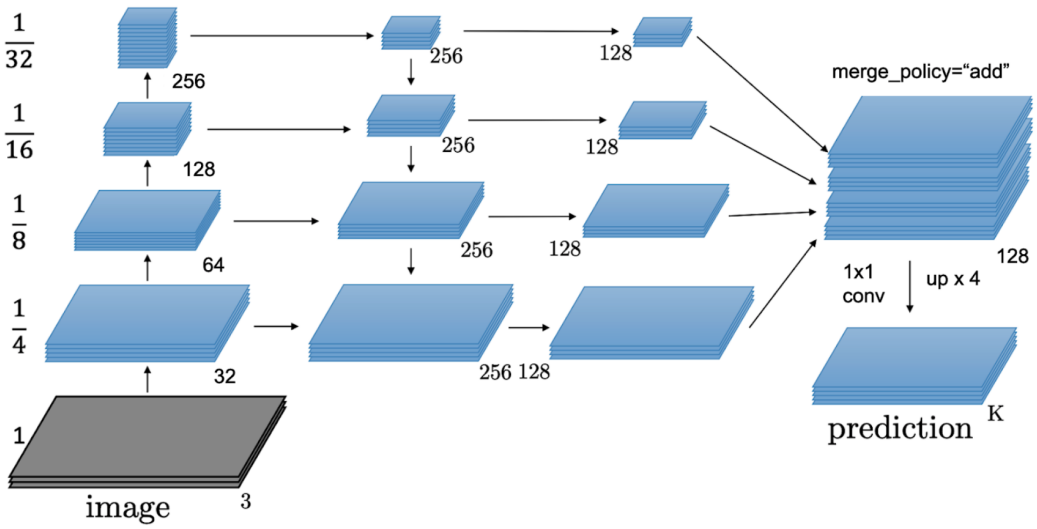

Figure 3: We adopt FPN as our backbone. The bottom-up pathway is the feedforward computation of the pretrained visual encoder. The top-down pathway hallucinates higher resolution features/outputs.

The encoder of the policy network consists of a 9-layer ResNet with Weight Standardization [26] and GroupNorm, followed by a 3-layer Conv1D block to encode RGB signals; a 2-layer MLP to encode sensor signals; a 2-layer fusion MLP to encode the concatenation of RGB embedding and sensor embedding.

3.4. Embodied Learning

The first stage is embodied learning, or pretraining, where we train the policy network for the PointGoal navigation task using SAC with the visual encoder introduced in Section 3.3. The task is successful if the robot gets closer than 0.36 m to the goal location, which is the size of the robot. The reward is shaped by the geodesic distance to the goal and has a collision penalty [28]. More specifically, the robot gets a positive reward if it gets closer to the target and a negative reward if it moves further away from the goal location. A one-time large and positive success reward is given if the task is successful. After the first stage, we only keep the visual encoder and remove all other components of the policy network.

3.5. Downstream Evaluation

The second stage is downstream evaluation, or finetuning the pretrained visual encoder on different downstream tasks.

For relative camera pose estimation, the input is a pair of RGB images. We use a siamese network by cloning the visual encoder and then concatenate the embeddings of the pair images from the siamese network and use an MLP layer to infer the relative camera pose. The loss function is the cross-entropy loss between the ground truth and the predicted relative camera pose.

For depth estimation and surface normal estimation, the input is an RGB image and the output is either a depth map or surface normal. This is an image generation task and we adopt the Feature Pyramid Network (FPN) [20] as our backbone. The bottom-up pathway is the feedforward computation of the pretrained visual encoder. The top-down pathway hallucinates higher resolution features and outputs either a depth map or surface normal. The network architecture is shown in Figure 3. We use L1 loss for depth estimation and cosine similarity loss for surface normal estimation.

For optical flow and scene flow estimation, we still use the FPN backbone. However, since the input is a video clip, we add a few Conv3D layers with a kernel size of $3\times1\times1$ to aggregate the image frames. We use L1 loss for both optical flow and scene flow estimation.

4. Experiment

4.1. Downstream Tasks

Relative Camera Pose Estimation. Following Taskonomy [39], for two different views with the same optical centers, we want to estimate the 6-DOF relative camera pose, including xyz translation and yaw, pitch, roll. Different from [39], we formulate this problem as a classification task, in which we need to collect a dataset composed of paired images with different camera poses. For the first image in pairs, we render it from a scene with a random pose; for the second image, we render it with $\{+0.3\text{m}, -0.3\text{m}\}$ of relative xyz translation and $\{+15°, -15°\}$ of relative yaw,

pitch, roll. There are $2^6 = 64$ categories in this task. Top-1 accuracy is used as the main metric for this task.

Depth Estimation. Estimating depth is very important for an agent to understand geometric relations within a scene. Such relations are essential to many recognition tasks and have rich applications in robotics, scene understanding and 3-D reconstruction [29]. We utilize the depth map rendered by iGibson and perform depth estimation. The raw depth values are clipped between 0m and 0.5m and normalized by 0.5m. We use absolute relative difference [6] as the main metric, which is scale-invariant.

Surface Normal Estimation. Surface normal is another modality that describes physical geometry and facilitate 3-D scene understanding. It is an indicator of where an object could be placed and helps understand the relation between objects and the environment. The main metric for this task is the mean angle distance on a per-pixel-basis [8].

Optical Flow Prediction. Optical flow captures the motion of objects and surfaces in a scene. Understanding optical flow helps an agent predicts the dynamics of a physical world. Given a video clip of 4 RGB frames as input, we would like to predict the optical flow between the 4th frame and 5th frame. The unit of optical flow rendered by iGibson is normalized device coordinate (NDC), which is in $[-1, 1] \times [-1, 1]$. The raw optical flow values are clipped between -1 and 1. As per [5], the metric for this task is end-point error (EPE).

Scene Flow Prediction. Scene flow is the 3D motion field of points in a scene [34], while optical flow is its projection into an image plane. Scene flow provides motion cues for many tasks including object segmentation, action recognition, camera pose estimation, etc [23]. Given a video clip of 4 RGB frames as input, we would like to predict the scene flow between the 4th frame and 5th frame. The unit of scene flow is meter, The raw scene flow values are clipped between -1m and 1m. We use 3D EPE as the main metric, following [23].

4.2. Downstream Datasets

We use iGibson to generate 3 datasets for downstream evaluation. The details are shown in Table 1. Note that we generate depth maps and surface normals at the same time. Similarly, optical flow and scene flow are generated simultaneously.

4.3. Embodied Learning Results

We render RGB signals with a resolution of 224×224. We train our policy network with 1M steps across 16 different scenes. The learning rate for SAC is 1e-4. Success weighted by (normalized inverse) Path Length (SPL) is a common metric used to measure the agent's navigation performance, which is defined as

$$\frac{1}{N} \sum_{i=1}^{n} S_i \frac{l_i}{\max(p_i, l_i)}. \tag{4}$$

N is the number of test episodes. l_i is the shortest path distance from the agent's initial position to the goal in episode i. p_i is the length of the path actually taken by the agent in this episode. S_i is a binary indicator of success in episode i. We evaluate 20 episodes for each scene and our SPL averaged over all 16 scenes is 0.597. A breakdown of all 16 scenes is shown in Table 2. For qualitative results of PointGoal navigation, please see supplementary material.

Table 1: Statistics for different types of datasets.

Task(s)	Training size	Validation size
relative camera pose	40960	20480
depth estimation surface normal estimation	40000	20000
optical flow prediction scene flow prediction	12000	4000

Table 2: Quantitative results on PointGoal navigation.

scene01 0.558	scene02 0.458	scene03 0.510	scene04 0.662
scene05 0.755	scene06 0.647	scene07 0.750	scene08 0.837
scene09 0.428	scene10 0.442	scene11 0.740	scene12 0.812
scene13 0.671	scene14 0.055	scene15 0.481	scene16 0.746

4.4. 3D Representation Learning Results

After the embodied learning stage, we come to 3D representation learning, where we use a spectrum of downstream tasks to evaluate the learned skills of the agent. Table 3 shows the results of downstream evaluation. For each task, we compared our approach (embodiment for representation learning) with training from scratch. We highlight the better approach using the green color. Our approach is better than training-from-scratch on 4 tasks. This is because understanding depth and surface normal and predicting optical flow and scene flow are quite important for the agent to avoid objects and obstacles. The camera pose, however, is not that important during navigation. We also show qualitative results of downstream evaluation from Figure 4 to Figure 8. The model learns quite good 3D representations of the scene.

Table 3: Quantitative results on downstream evaluation.

Task	Evaluation Metric	Training from Scratch	Embodiment for Representation Learning
relative camera pose estimation	top-1 accuracy	**10.9%**	9.8%
depth estimation	relative difference	0.344	**0.322**
surface normal estimation	mean angle distance	35.9	**35.2**
optical flow prediction	end-point error	0.211	**0.138**
scene flow prediction	3D end-point error	0.247	**0.217**

Figure 4: Relative camera pose estimation task. 010100 means relative x translation is -0.3m, relative y translation is 0.3m, relative z translation is -0.3m, yaw is $+15°$, pitch is $-15°$ and roll is $-15°$. Each row corresponds to a specific relative camera pose and contains 4 pairs of RGB images.

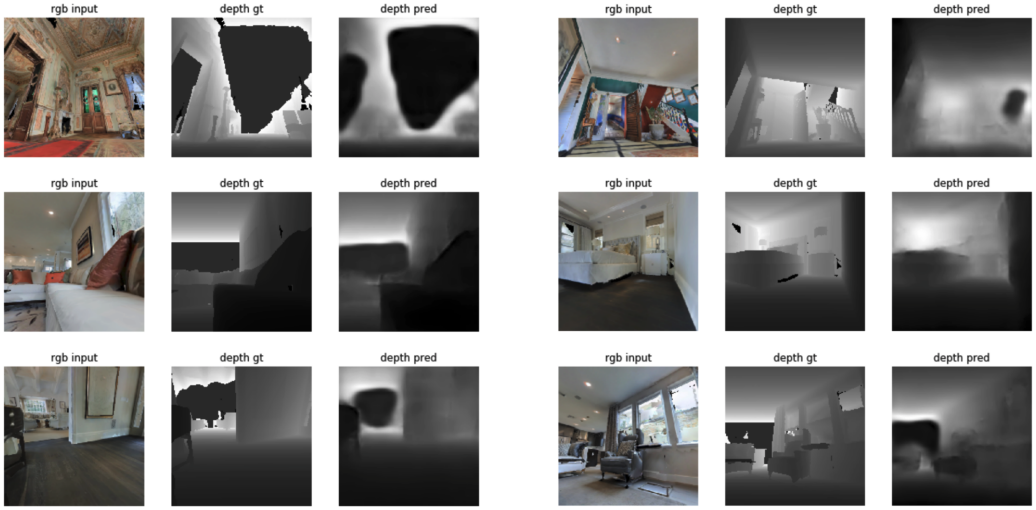

Figure 5: Depth estimation task. For each triplet of images, the left one is RGB input, the middle one is the ground truth of depth output, and the right one is the predicted depth output.

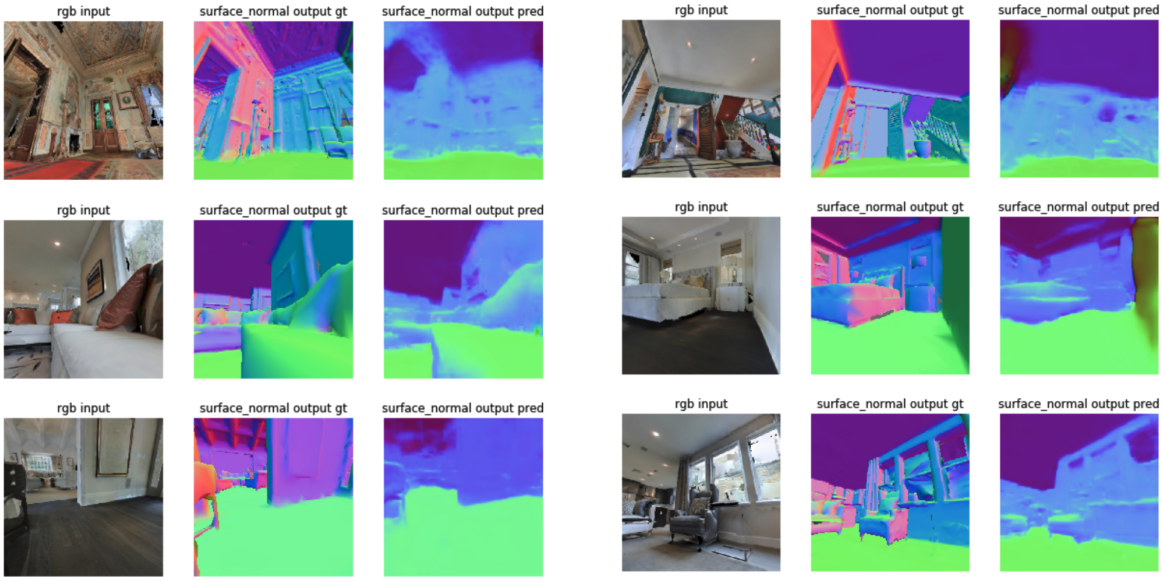

Figure 6: Surface normal estimation task. For each triplet of images, the left one is RGB input, the middle one is the ground truth of surface normal, and the right one is the predicted surface normal.

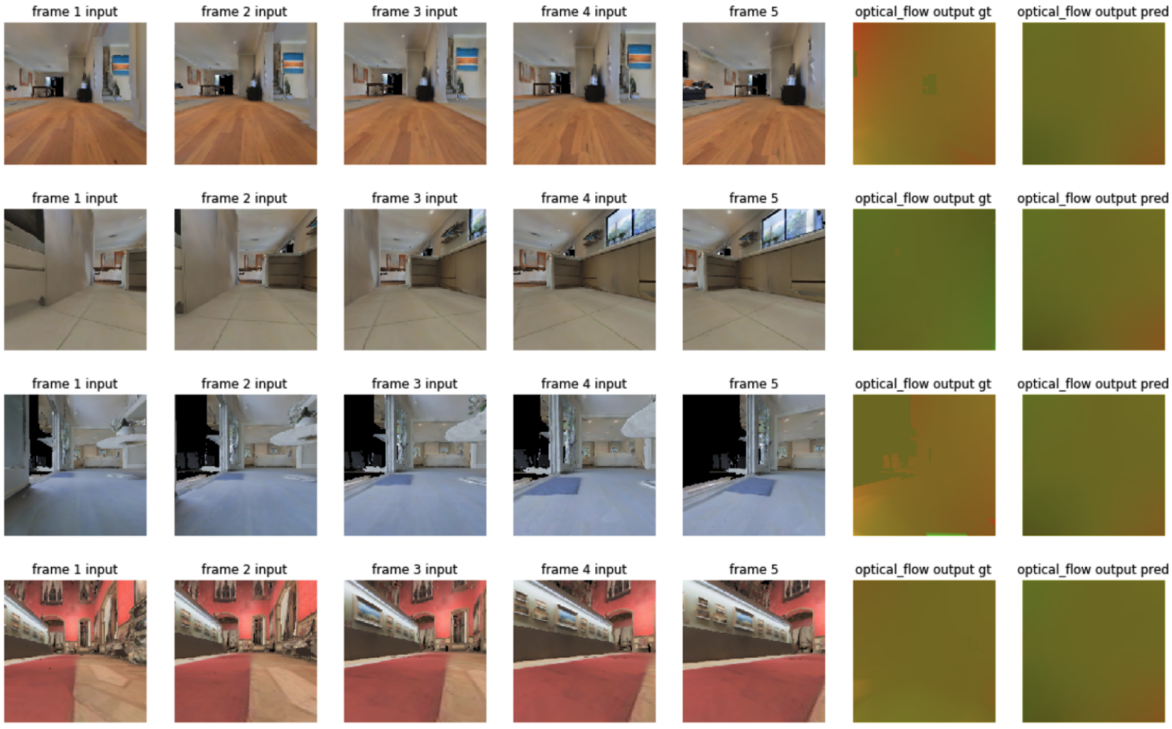

Figure 7: Optical flow prediction task. In each row, the first four images are the input video clip. The optical flow is the output and obtained from the 4th image (column) and the 5th image (column). The 6th image (column) is the ground truth of the optical flow, while the last column is the predicted optical flow.

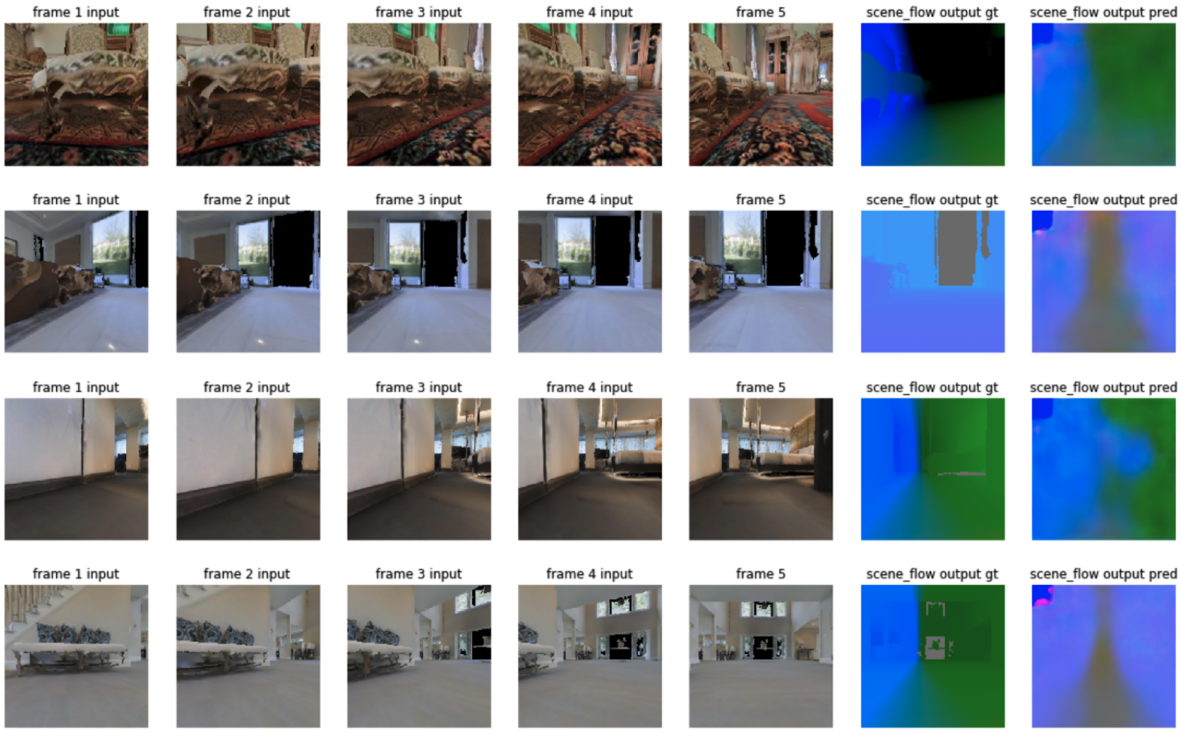

Figure 8: Scene flow prediction task. In each row, the first four images are the input video clip. The optical flow is the output and obtained from the 4th image (column) and the 5th image (column). The 6th image (column) is the ground truth of the scene flow, while the last column is the predicted scene flow.

5. Conclusion

In this work, we take a step towards closing the gap between AI and human intelligence by studying how embodiment affects 3D representation learning. We build a distributed RL framework and train a policy network for PointGoal navigation tasks. After that, we finetune the encoder on different downstream tasks to evaluate 3D representations of the scene learned by the agent. We show that embodiment helps the agent acquire sophisticated abilities by solving navigation tasks using deep RL. Another thing that is worth trying (suggested by Kuan Fang) is to use the estimated depth as the input to the policy network and see whether it improves the performance of embodied learning.

6. Acknowledgement

This course project is part of the research project that Guanzhi Wang has been working on at the Stanford Vision and Learning Lab. He is advised by Jim Fan, Prof. Fei-Fei Li and Prof. Silvio Savarese. He also got great advice from the course assistant, Kuan Fang.

7. Supplementary Material

Code repo: https://github.com/wangguanzhi/CS231A.
Navigation videos: Google Drive link.

References

[1] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[4] C. Doersch and A. Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

[5] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[7] L. Fan, Y. Zhu, J. Zhu, Z. Liu, O. Zeng, A. Gupta, J. Creus-Costa, S. Savarese, and L. Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*, pages 767–782. PMLR, 2018.

[8] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3d primitives for single image understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3392–3399, 2013.

[9] K. Gregor, D. J. Rezende, F. Besse, Y. Wu, H. Merzic, and A. v. d. Oord. Shaping belief states with generative environment models for rl. *arXiv preprint arXiv:1906.09237*, 2019.

[10] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017.

[11] S. Gupta, D. Fouhey, S. Levine, and J. Malik. Unifying map and landmark based representations for visual navigation. *arXiv preprint arXiv:1712.08125*, 2017.

[12] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. *arXiv preprint arXiv:1809.01999*, 2018.

[13] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

[14] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine. Soft actor-critic algorithms and applications, 2018.

[15] T. Han, W. Xie, and A. Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[16] D. Jayaraman and K. Grauman. End-to-end policy learning for active visual categorization. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1601–1614, 2018.

[17] A. Kirillov, R. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6399–6408, 2019.

[18] I. Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6129–6138, 2017.

[19] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[20] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[22] S. Liu, E. Johns, and A. J. Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1871–1880, 2019.

[23] X. Liu, C. R. Qi, and L. J. Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019.

[24] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

[25] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[26] S. Qiao, H. Wang, C. Liu, W. Shen, and A. Yuille. Weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

[27] S. K. Ramakrishnan, T. Nagarajan, Z. Al-Halah, and K. Grauman. Environment predictive coding for embodied agents. *arXiv preprint arXiv:2102.02337*, 2021.

[28] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. 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.

[29] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *Advances in neural information processing systems*, pages 1161–1168, 2006.

[30] B. Shen, F. Xia, C. Li, R. Martın-Martın, L. Fan, G. Wang, S. Buch, C. D'Arpino, S. Srivastava, L. P. Tchapmi, K. Vainio, L. Fei-Fei, and S. Savarese. igibson, a simulation environment for interactive tasks in large realistic scenes. *arXiv preprint*, 2020.

[31] C. Sun, F. Baradel, K. Murphy, and C. Schmid. Learning video representations using contrastive bidirectional transformer. *arXiv preprint arXiv:1906.05743*, 2019.

[32] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[33] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding, 2018.

[34] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999.

[35] D. Wei, J. J. Lim, A. Zisserman, and W. T. Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018.

[36] F. Xia, W. B. Shen, C. Li, P. Kasimbeg, M. E. Tchapmi, A. Toshev, R. Martín-Martín, and S. Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2):713–720, 2020.

[37] A. Yahya, A. Li, M. Kalakrishnan, Y. Chebotar, and S. Levine. Collective robot reinforcement learning with distributed asynchronous guided policy search. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 79–86. IEEE, 2017.

[38] J. Yang, Z. Ren, M. Xu, X. Chen, D. Crandall, D. Parikh, and D. Batra. Embodied visual recognition. *arXiv preprint arXiv:1904.04404*, 2019.

[39] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

[40] Y. Zhu, D. Gordon, E. Kolve, D. Fox, L. Fei-Fei, A. Gupta, R. Mottaghi, and A. Farhadi. Visual semantic planning using deep successor representations. In *Proceedings of the IEEE international conference on computer vision*, pages 483–492, 2017.

[41] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, pages 3357–3364. IEEE, 2017.

[42] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas, et al. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564*, 2018.