From Words to Actions: Language-Guided Hierarchical RL for Object Rearrangement

Joe Lin
Department of Computer Science
University of California, Los Angeles
Los Angeles, USA
joelintech@ucla.edu

Allen Luo
Department of Computer Science
University of California, Los Angeles
Los Angeles, USA
atluo04@ucla.edu

Nishant Ray
Department of Computer Science
University of California, Los Angeles
Los Angeles, USA
nishantray@ucla.edu

***Abstract*—We explore the use of natural language to guide hierarchical reinforcement learning (HRL) agents in long-horizon indoor rearrangement tasks. Using the Habitat Lab framework and the ReplicaCAD *rearrange_easy* benchmark, we train a high-level policy to select among pre-trained low-level skills based on both visual observations and language instructions. To fuse multimodal inputs, we evaluate two techniques: feature-wise linear modulation (FiLM) and cross-attention. Language instructions are generated using a large language model and embedded via a pretrained CLIP encoder. Our results show that incorporating language slightly decreases task performance; however, the overall success rate remains low due to compounding errors from unreliable low-level skills. This suggests that while language offers valuable context, its benefits are limited without reliable underlying skills. We discuss key challenges and propose directions for mitigating cascading failures.**

I. Introduction

Society has long considered the development and integration of automated agents within the home environment to assist with everyday tasks. Recent advances in robotics, computer vision, and natural language processing, have made it possible for a portion of this vision to come to fruition, yet we are still far from developing robust embodied agents that interact with humans in a realistic and natural manner.

A crucial component for such an agent is its ability to perform everyday indoor tasks, which is a significant challenge for embodied AI. While reinforcement learning (RL) has shown promise in enabling agents to learn behaviors through interaction, scaling to long-horizon tasks remains difficult due to sparse rewards and large action spaces [1]. Hierarchical reinforcement learning (HRL) offers a natural solution by dividing long-horizon tasks into composable low-level skills, which are orchestrated by a high-level policy [2]. Additionally, recent advances in multimodal learning suggest that language can serve as an effective interface for specifying high-level goals [1], [3]–[5]. Moreover, feature-wise linear modulation (FiLM) [6] and cross-attention mechanisms [7], [8] present effective strategies for fusing vision and natural language observations. So, in this work, we build upon the Habitat Lab [9]–[11] framework to incorporate natural language instructions into the high-level controller, enabling it to choose low-level skills conditioned on both language and visual observations.

We use the ReplicaCAD [10] *rearrange_easy* benchmark and augment it with natural language instructions generated from template-based scene descriptions. Our experiments compare the performance of language-conditioned HRL agents against visual-only baselines.

II. Problem Statement

We focus on addressing the indoor environment rearrangement task. Our goal is to develop a hierarchical reinforcement learning agent that can:

- Utilize natural language to guide low-level skill selection
- Effectively fuse language and visual input to improve task performance and generalization
- Train and evaluate on realistic 3D rearrangement tasks with diverse goals and object placements

III. Methods

A. Dataset

We leverage the ReplicaCAD dataset, a derivative of the Replica dataset [12], which provides an artist's recreation of the *FRL apartment* intended for use with the Habitat simulator. More specifically, we use the *rearrange_easy* subset for training and evaluation. The train split contains 50000 episodes of rearrangement problems across 63 scenes. The validation split contains 1000 episodes from a separate set of 21 diverse scenes.

We augment the dataset with natural language instructions by first extracting the target object and goal locations from the episode data. Because the locations are provided as numberings, we build a mapping from provided indices to receptacle locations in the scene, and then construct a template sentence for each episode. This is then passed into a LLM to reword into natural language. Finally, each generated sentence is transformed into visual-aware features through a CLIP model. Our pipeline utilizes Google's Gemini 2.0 Flash [13] and OpenAI's ViT-B/32 CLIP model [14] as our LLM and CLIP model, respectively.

B. Low-Level Skills Training

Habitat Lab provides a framework for training low-level skills that can be composed by a high-level policy in a hierarchical reinforcement learning (HRL) setup [10]. These skills correspond to discrete, reusable behaviors such as picking or placing objects, and are trained independently using Proximal

Policy Optimization (PPO) [15]. Once trained, these low-level controllers can be used by a high-level policy conditioned on task-level goals. Of the 9 low-level skills that Habitat Lab provides, we use the following three as we deem them the most essential for the rearrangement task:

1) `pick`
2) `place`
3) `navigate`

Each skill policy takes in a set of input state and visual observations and learns to output low-level actions for robot control. For the `pick` skill, we provide `head_depth` sensor reading $Z \in \mathbb{R}^{256\times256\times1}$, robot grasping state `is_holding` $g \in 0, 1$, revolute `joint` states $J \in \mathbb{R}^7$, object's starting position $p_o \in \mathbb{R}^3$, and desired `relative_resting_position` $p_r \in \mathbb{R}^3$. `pick`'s action space includes the arms' joint-level motor control $a_J \in \mathbb{R}^7$, a suction gripper's control $a_g \in \mathbb{R}^1$, and robot base controls $a_b \in \mathbb{R}^2$. Hence, $A_{\text{pick}} = a_J \cup a_g \cup a_b \in \mathbb{R}^{10}$. Note that the base is restricted in planar movement controls. The `place` skill receives the same observations and its action space is identical to `pick`. For the `navigate` skill, we provide `goal_to_agent` position $p_{ga} \in \mathbb{R}^2$, `head_depth` sensor reading Z, and revolute `joint` states J. `navigate` uses the robot base's action space $A_{\text{nav}} \in \mathbb{R}^3$.

C. Hierarchical RL Training

We train a HRL agent based on Habitat Lab's benchmark consisting of a high-level policy that selects among a set of pre-trained low-level skills. The high-level policy receives both visual observations and language instructions, and outputs a sequence of skill calls with corresponding arguments. Additionally, the target object and its starting position, and the goal position are given to the model. Along with a baseline with no language input, we experimented with 2 different approaches to fusing language and visual input:

1) Visual feature conditioning through feature-wise linear modulation (FiLM)
2) Visual and language feature fusion through cross-attention

We will explain our methodology for each of these in the following sections:

Baseline: We run the benchmark provided by Habitat Lab. Visual observations are transformed into features through the provided ResNet-18 encoder [16], and then fed into a recurrent policy network trained with PPO. The agent is not conditioned on language input and must solve the task using only visual and sensor observations.

FiLM: We modify the ResNet visual encoder to include FiLM layers in each residual block, similar to the architecture described in [6]. These FiLM layers allow the network to condition visual feature extraction on the language input by applying learned, instruction-dependent affine transformations. This enables the agent to extract task-relevant visual cues based on the provided language command.

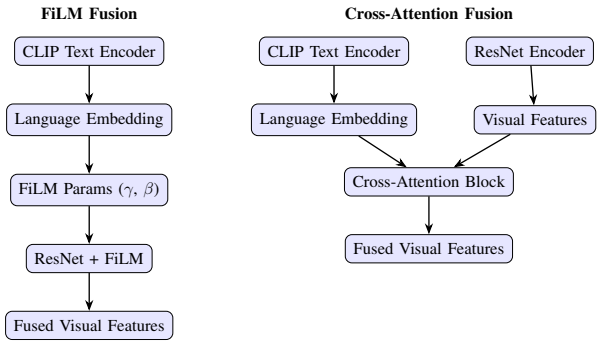

Fig. 1. Fused vision language feature pipeline using FiLM (left) and Cross-Attention (right).

$$\begin{aligned} \gamma, \beta &\leftarrow \text{FiLM}(E_l) \\ F_v &\leftarrow \gamma F_v + \beta \end{aligned} \tag{1}$$

FiLM layer implementation was done using PyTorch [17] linear layers. FiLM takes in precomputed CLIP embeddings E_l during the forward pass and outputs scale and shift parameters, γ and β. Then, the visual features F_v are linearly transformed with (1) before being passed to the following ReLU layer.

Cross-Attention: We add a cross-attention block to fuse visual features from the ResNet encoder with language embeddings. Following a common multi-modal fusion strategy, the language features are used as the query Q_l, while the visual features serve as the key K_v and value K_v. This allows the agent to attend selectively to spatial regions in the visual input that are most relevant to the language input.

$$F = \text{softmax}\left(\frac{Q_l K_v^T}{\sqrt{d_k}}\right) V_v \tag{2}$$

D. Reward Function Design

Furthermore, to facilitate better learning, we add a dense reward signal. Originally, Habitat Lab uses a `pddl_subgoal_reward` as its reward signal, which is a sparse reward based on completion of certain subgoals (i.e. navigating to an object, picking up the object, etc.). However, during training, we observe that the agent struggles to learn effectively from sparse rewards. To address this, we augment the reward with additional terms that are more dense than the original, building a `CompositeReward`. This is the sum multiple rewards, including the original `pddl_subgoal_reward` and a `move_obj_reward`. Thus, our reward function becomes:

$$r_t = \lambda_p r_t^{(p)} + \lambda_m r_t^{(m)} \tag{3}$$

where r_t is the total reward at timestep t, $r_t^{(p)}$ is the `pddl_subgoal_reward` at timestep t, $r_t^{(m)}$ is the `move_obj_reward` at timestep t, and λ_p, λ_m are tunable coefficients (we used $\lambda_p = \lambda_m = 1$).

The `move_obj_reward` includes three primary components. (1) a distance reward is given to steer the agent towards

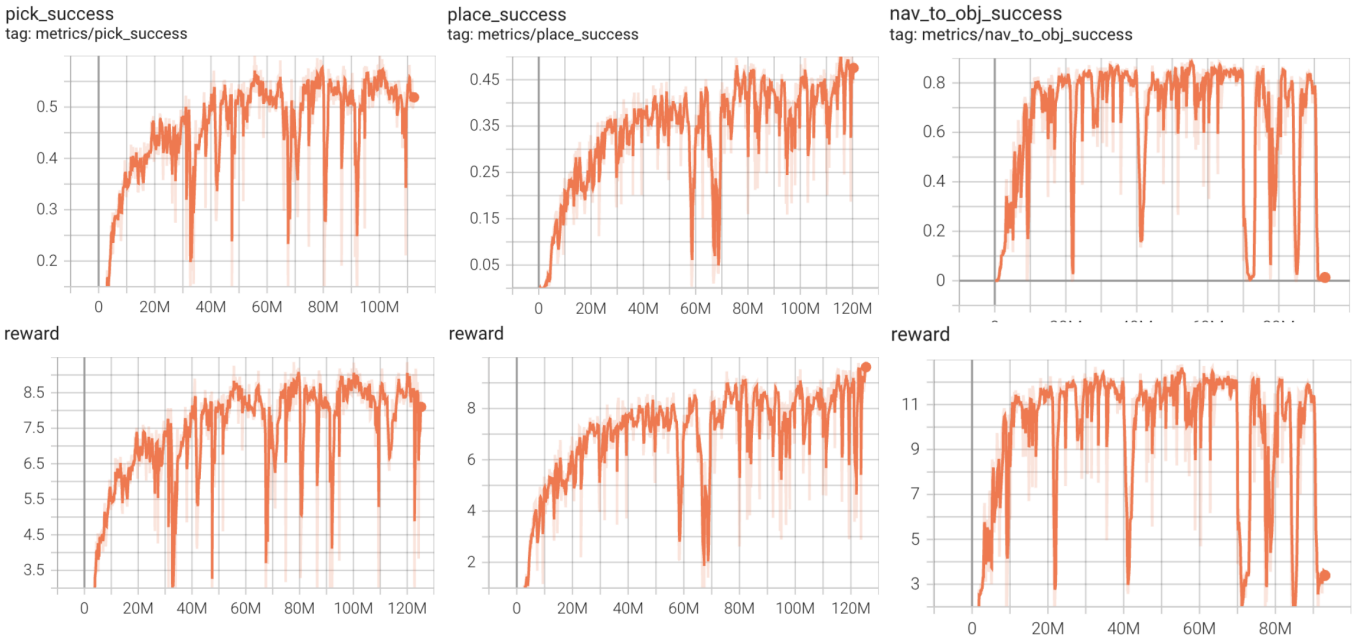

Fig. 2. Success rate and reward training curves for `pick`, `place`, and `nav` (respectively from left to right) skills.

the object and then the goal, (2) a one-time pick reward is given when the target object is successfully grasped, and (3) a reward given for accurately placing the target object.

IV. Results

A. Low-level Skills

Training the low-level skill policies proved to be quite difficult. These policies were extremely data hungry and required multiple days to train on a single NVIDIA RTX A5000. In the end, we achieved $\approx 60\%$ on `pick`, $\approx 45\%$ on `place`, and $\approx 80\%$ on `navigate`. Success rate and reward training curves are shown in Figure 2.

We observe that, as expected, the `action_loss` and `value_loss` were unrepresentative of training progress and we relied on monitoring the `reward` for training insights. The agent seems to excel rapidly when learning the `navigate` skill, whereas for `pick` and `place`, the agent struggles significantly more, oftentimes plateauing in success rate. We attribute this to the inherent difference in difficulty of these tasks. `navigate` is regarded as a high-level locomotion task, which requires less degrees of freedom to solve. `pick` and `place` on the other hand suffer from the higher order complexities of grasping or releasing an object in a way that does not cause undesired object-scene interference.

B. High-level Neural Policy

Because of compounding errors from poor low-level policies, training the high-level policy presented further difficulties. We managed to achieve $\approx 5.0\%$ success rate on the rearrangement task with our approach. Comparatively, the baseline achieves ≈ 5.9. Figure 3 shows training success rates for the baseline (left) and the cross-attention methods (right). We found no notable difference between the baseline and FiLM methods.

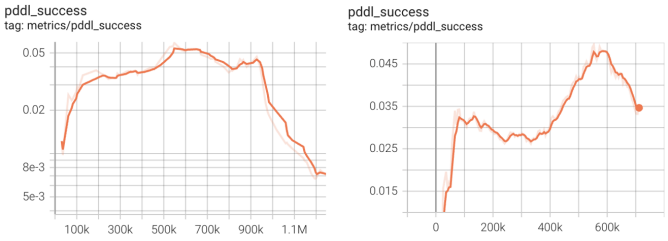

Fig. 3. Success rate curves for baseline (left) and our method (right) for the rearrangement task.

During training, we noticed that reward curves indicate some learning progress at first as the agent seems to hone in on the `navigate` skill. However, progress halts at 5% success rate and plummets. In this stage, we observe the rollout durations and object collisions to reduce drastically, hence we conclude the agent is learning to prefer inaction to avoid negative rewards.

V. Discussion

We find that adding language input slightly decreases task performance. However, given the hierarchical nature of the policy where errors can compound through skill chaining, the observed results are difficult to interpret. Combined with the overall low success rate, this makes it unclear whether the performance differences reflect issues with combining language and vision or arise from incidental correlations and artifacts in the training data.

Other works have also tackled this problem. The Habitat Rearrangement Challenge in 2022 [18] provides a baseline with 30% success rate. However, it is important to note that this implementation uses a fixed high level policy which is not learned, therefore reducing complexity and potential errors significantly. [19] achieves a success rate of 64% by implementing mobile manipulation skills, therefore reducing compounding errors seen with the original stationary skills.

These findings suggest that to achieve robust language-conditioned rearrangement performance, it is crucial to not only improve language grounding but also to address the structural instability caused by skill chaining. Mitigating compounding errors, whether by improving skill reliability, refining high-level decision-making, or incorporating corrective feedback mechanisms, may be a necessary prerequisite for realizing the full potential of multimodal hierarchical policies.

A. Challenges

We faced many challenges during our experimentation. First off, we faced issues with bugs in the Habitat Lab repository, which hindered our progress in training skills, and we suspect are from version incompatibilities. The majority of these bugs were related to tensor shape issues, specifically with the recurrent neural network state encoders used in the policy (as Habitat Lab uses these encoders for both high and low level actions).

Additionally, as mentioned before, training took longer than expected, with some skills taking multiple days to train. Environment issues also stopped us from running on multiple devices.

VI. Conclusion

In this work, we explored the integration of natural language input into hierarchical reinforcement learning agents for the task of indoor environment rearrangement. By fusing language and vision through FiLM and cross-attention mechanisms, we aimed to improve high-level policy decisions in complex, multi-step tasks. While our experiments show modest performance decreases from incorporating language, the overall success rate remains low, highlighting the persistent challenge of compounding errors in hierarchical control.

Our results emphasize the importance of robust low-level skill execution. Furthermore, they suggest that language-conditioned policies alone are insufficient without architectural and algorithmic strategies that can reduce compounding errors.

Future work should explore tighter integration between high and low-level components and more effective training strategies. Additionally, richer and more realistic language annotations could help evaluate whether agents truly understand language, or merely exploit dataset-specific patterns.

A. Member Contributions

The contributions are as follows:

1) **Joe:** Training RL, implementing features, report
2) **Allen:** Dataset augmentation, help with features, report
3) **Nishant:** Website, help with debugging, report

References

[1] W. Shi, X. He, Y. Zhang, C. Gao, X. Li, J. Zhang, Q. Wang, and F. Feng, "Large language models are learnable planners for long-term recommendation," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR 2024. ACM, Jul. 2024, p. 1893–1903. [Online]. Available: http://dx.doi.org/10.1145/3626772.3657683

[2] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," 2018. [Online]. Available: https://arxiv.org/abs/1805.08296

[3] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng, "Do as i can, not as i say: Grounding language in robotic affordances," 2022. [Online]. Available: https://arxiv.org/abs/2204.01691

[4] D. Misra, J. Langford, and Y. Artzi, "Mapping instructions and visual observations to actions with reinforcement learning," 2017. [Online]. Available: https://arxiv.org/abs/1704.08795

[5] M. Dalal, T. Chiruvolu, D. Chaplot, and R. Salakhutdinov, "Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks," 2024. [Online]. Available: https://arxiv.org/abs/2405.01534

[6] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," 2017. [Online]. Available: https://arxiv.org/abs/1709.07871

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[8] H. Lin, X. Cheng, X. Wu, F. Yang, D. Shen, Z. Wang, Q. Song, and W. Yuan, "Cat: Cross attention in vision transformer," 2021. [Online]. Available: https://arxiv.org/abs/2106.05786

[9] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[10] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[11] X. Puig, E. Undersander, A. Szot, M. D. Cote, R. Partsey, J. Yang, R. Desai, A. W. Clegg, M. Hlavac, T. Min, T. Gervet, V. Vondrus, V.-P. Berges, J. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi, "Habitat 3.0: A co-habitat for humans, avatars and robots," 2023.

[12] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The Replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.

[13] "Google gemini 2.0 flash," https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash, accessed: 2025-06-08.

[14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021. [Online]. Available: https://arxiv.org/abs/2103.00020

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019. [Online]. Available: https://arxiv.org/abs/1912.01703

[18] A. Szot, K. Yadav, A. Clegg, V.-P. Berges, A. Gokaslan, A. Chang, M. Savva, Z. Kira, and D. Batra, "Habitat rearrangement challenge 2022," https://aihabitat.org/challenge/2022_rearrange, 2022.
[19] J. Gu, D. S. Chaplot, H. Su, and J. Malik, "Multi-skill mobile manipulation for object rearrangement," 2022. [Online]. Available: https://arxiv.org/abs/2209.02778