

Integrating Vision Foundation Models with Reinforcement Learning for Enhanced Object Interaction

Ahmad Farooq^{*1} and Kamran Iqbal²

Abstract— This paper presents a novel approach that integrates vision foundation models with reinforcement learning to enhance object interaction capabilities in simulated environments. By combining the Segment Anything Model (SAM) and YOLOv5 with a Proximal Policy Optimization (PPO) agent operating in the AI2-THOR simulation environment, we enable the agent to perceive and interact with objects more effectively. Our comprehensive experiments, conducted across four diverse indoor kitchen settings, demonstrate significant improvements in object interaction success rates and navigation efficiency compared to a baseline agent without advanced perception. The results show a 68% increase in average cumulative reward, a 52.5% improvement in object interaction success rate, and a 33% increase in navigation efficiency. These findings highlight the potential of integrating foundation models with reinforcement learning for complex robotic tasks, paving the way for more sophisticated and capable autonomous agents.

Index Terms: Reinforcement Learning, Object Interaction, Vision Foundation Models, Segment Anything Model, AI2-THOR Simulation

I. INTRODUCTION

The development of autonomous agents capable of interacting with complex environments is a fundamental goal in robotics and artificial intelligence. A key challenge in achieving this goal lies in equipping agents with advanced perception and decision-making abilities that enable them to understand and manipulate their surroundings effectively. Robust perception allows agents to recognize and localize objects, comprehend spatial relationships, and interpret dynamic scenes [1], [2]. Reinforcement Learning (RL) provides a framework for agents to learn optimal behaviors through trial-and-error interactions with the environment [3], [4].

Recent advancements in computer vision and RL have led to significant progress in tasks such as visual navigation [5], [6], object manipulation [7], [8], and human-robot interaction [9], [10]. However, integrating sophisticated perception models with RL agents remains a challenging problem due to factors such as the complexity of real-time processing in dynamic environments and computational overhead.

This work was not supported by any organization.

^{*}Corresponding Author: Ahmad Farooq

¹Ahmad Farooq is a Ph.D. Candidate in the Electrical and Computer Engineering Department at the University of Arkansas at Little Rock, AR, 72204, USA afarooq@ualr.edu

²Kamran Iqbal is a Professor in the Electrical and Computer Engineering Department at the University of Arkansas at Little Rock, AR, 72204, USA kxiqbal@ualr.edu

This is the author's preprint version of an article published in RCVF'25: Proceedings of the 2025 3rd International Conference on Robotics, Control and Vision Engineering. The final published version is available in the ACM Digital Library: <https://doi.org/10.1145/3747393.3747399>.

Vision foundation models, including YOLOv5 [11] and Segment Anything Model (SAM) [12], have demonstrated exceptional capabilities in object detection and segmentation tasks. These models are pre-trained on extensive datasets and can generalize across various domains, reducing the dependence on task-specific training data. By integrating these models with RL agents, we can enhance the agents' perceptual understanding, leading to improved performance in tasks that involve complex interactions with the environment.

In this work, we propose a novel approach that combines vision foundation models with reinforcement learning to enhance object interaction capabilities in simulated environments. Specifically, we integrate SAM and YOLOv5 into the perception pipeline of a Proximal Policy Optimization (PPO) agent operating within the AI2-THOR simulation environment [13]. The AI2-THOR environment offers richly interactive 3D scenes, providing a suitable platform for training agents in object interaction and navigation tasks.

Our approach addresses several key challenges:

- **Perception Integration:** We develop a method to effectively incorporate the outputs of SAM and YOLOv5 into the agent's observation space, allowing for enhanced scene understanding without incurring prohibitive computational costs.
- **Reward Function Design:** We formulate a reward function that balances exploration, object interaction, and goal achievement, guiding the agent to learn efficient strategies for interacting with objects.
- **Policy Learning:** We design a policy network architecture that leverages the enriched perceptual input to learn effective policies for object interaction tasks.

Our main contributions are as follows:

- 1) We present a novel framework that integrates vision foundation models with reinforcement learning agents to enhance object interaction capabilities.
- 2) We develop techniques to efficiently incorporate advanced perception outputs into the RL framework, addressing computational and integration challenges.
- 3) We conduct extensive experiments demonstrating significant performance improvements over a baseline agent, highlighting the effectiveness of our approach.
- 4) We provide insights into the implications of integrating foundation models with reinforcement learning, suggesting avenues for future research.

The rest of the paper is organized as follows. In Section II, we review related work on reinforcement learning in robotics

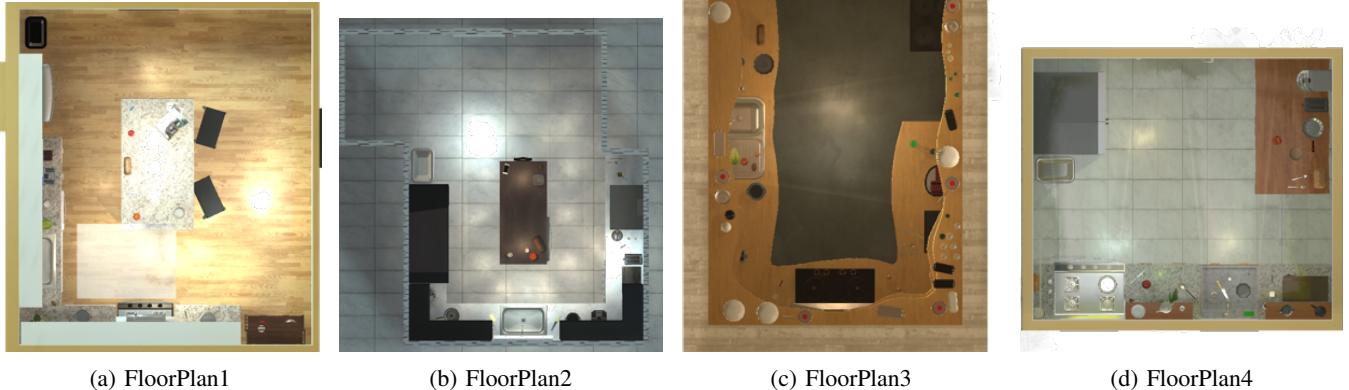


Fig. 1: Top-down views of the four kitchen environments in AI2-THOR used in our experiments. Each environment presents distinct challenges due to differences in layout and object placement.

and the integration of perception models. Section III details our proposed approach, including the integration of SAM and YOLOv5 with the RL agent and the design of the reward function. Section IV describes our experimental setup and evaluation metrics. We present and analyze the results in Section V, and discuss the implications of our findings in Section VI. We conclude the paper in Section VII, and outline directions for future research in Section VIII.

II. RELATED WORK

In this section, we review the literature on reinforcement learning in robotics, the integration of advanced perception models with RL agents, vision foundation models, and the use of simulation environments for training and evaluation.

A. Reinforcement Learning in Robotics

Reinforcement learning has been extensively applied to robotic control tasks, enabling agents to learn complex behaviors through interactions with the environment [3], [14]. Model-free RL algorithms, such as Proximal Policy Optimization (PPO) [4] and Deep Deterministic Policy Gradients (DDPG) [15], have shown success in continuous control problems. Applications include robotic manipulation [7], [16], locomotion [17], [18], and navigation [5], [6].

Challenges in applying RL to real-world robotics include sample inefficiency, safety during exploration, and the sim-to-real gap [19], [20]. Addressing these issues often involves leveraging simulation environments and incorporating prior knowledge or advanced perception.

B. Integration of Perception Models with RL

Integrating perception models with RL agents enhances their ability to interpret and interact with the environment. Early works utilized convolutional neural networks (CNNs) to process raw images for end-to-end learning [21]. Subsequent research incorporated object detection and semantic segmentation to provide richer observations [22], [23].

Recent studies have explored combining RL with advanced vision models to improve scene understanding. For instance, Zhu et al. [5] developed a target-driven visual navigation framework that enables agents to find specific targets in indoor environments using deep RL without requiring feature engineering or 3D reconstruction. Wang et al. [24] proposed a Reinforced Cross-Modal Matching approach that uses both extrinsic environmental rewards and intrinsic cycle-reconstruction rewards to better align navigation trajectories with natural language instructions.

However, the integration of large-scale foundation models like SAM and YOLOv5 into RL agents is still an emerging area, with potential to significantly boost performance in complex tasks.

C. Vision Foundation Models

Vision foundation models are pre-trained on large datasets and can generalize across various tasks. YOLOv5 [11] is renowned for real-time object detection with high accuracy, making it suitable for applications requiring quick responses. The Segment Anything Model (SAM) [12] provides prompt-based segmentation without additional training, enabling zero-shot generalization to new objects and scenes.

The deployment of these models in robotics has been facilitated by advances in computational hardware and optimization techniques [1], [25]. Integrating such models into RL agents offers the promise of enhanced perception without the need for extensive task-specific data collection and training.

D. Simulation Environments

Simulation environments like AI2-THOR [13], Habitat [26], and Gibson [27] provide realistic and interactive platforms for training RL agents. These environments enable safe and efficient exploration, reducing the risks and costs associated with real-world experimentation.

AI2-THOR offers high-fidelity 3D scenes with physics-based interactions, making it suitable for tasks involving object

manipulation and navigation. Our work leverages AI2-THOR to integrate advanced perception models with RL agents, providing a platform to evaluate the effectiveness of our approach in complex, interactive settings.

III. METHODOLOGY

In this section, we describe our approach to integrating vision foundation models with a reinforcement learning agent to enhance object interaction capabilities. We outline the framework for the RL agent, the perception pipeline, the reward function design, and the policy network architecture.

Our goal is to enable an RL agent to effectively perceive and interact with objects in a complex environment by leveraging the advanced capabilities of SAM and YOLOv5. Figure 2 illustrates the overall system architecture.

A. Framework for RL Agent

We utilize the AI2-THOR simulation environment [13], focusing on four kitchen scenes: FloorPlan1 to FloorPlan4. Each scene presents unique spatial configurations and object placements, providing diverse challenges for the agent.

The agent's action space \mathcal{A} includes:

- **Navigation Actions:** *MoveAhead, RotateLeft, RotateRight, LookUp, LookDown.*
- **Interaction Actions:** *PickupObject, DropObject.*

The observation space O consists of RGB images captured from the agent's first-person perspective at each time step t .

B. Perception Pipeline

To enhance the agent's perception, we integrate SAM [12] and YOLOv5 [11] into the observation processing pipeline.

1) *YOLOv5 for Object Detection:* YOLOv5 processes the raw RGB image to detect objects, outputting bounding boxes $B = \{b_i\}_{i=1}^N$ and class labels $C = \{c_i\}_{i=1}^N$, where N is the number of detected objects. This provides the agent with information about the presence and location of objects.

2) *SAM for Object Segmentation:* Using the bounding boxes from YOLOv5 as prompts, SAM generates segmentation masks $M = \{m_i\}_{i=1}^N$, providing precise object contours and spatial relationships. This detailed segmentation aids the agent in understanding the environment's structure.

3) *Feature Encoding:* The combined outputs (B, C, M) are encoded using a convolutional neural network (CNN) to produce a feature representation $\phi(s_t)$, where s_t is the state at time t . This representation captures both the semantic and spatial information necessary for decision-making.

C. Reward Function Design

Designing an effective reward function is critical for guiding the agent's learning. We define the reward r_t at time t as:

$$r_t = \alpha \cdot \Delta d_t + \beta \cdot s_t - \gamma \cdot c_t \quad (1)$$

where:

- $\Delta d_t = d_{t-1} - d_t$ is the change in distance to the target object, encouraging the agent to approach the target.
- s_t is a success indicator, $s_t = 1$ if the agent successfully interacts with the target object, and 0 otherwise.

- c_t is a penalty for collisions or invalid actions.
- α, β , and γ are weighting factors tuned empirically.

This reward structure incentivizes efficient navigation, successful interaction, and discourages undesirable behaviors.

D. Policy Network Architecture

The policy network consists of two main components:

- 1) **Perception Encoder:** A CNN that processes the feature representation $\phi(s_t)$ to extract high-level features.
- 2) **Policy and Value Heads:** Two fully connected layers that output the action probabilities $\pi(a_t|s_t)$ and state value estimates $V(s_t)$.

We employ the PPO algorithm [4] to optimize the policy, utilizing a clipped surrogate objective to ensure stable learning.

Integrating large perception models introduces computational overhead. To address this, we optimize the inference pipelines and utilize batch processing where possible. We also design an efficient feature encoding strategy that captures essential information without overcomplicating the state representation.

IV. EXPERIMENTS

We conduct a series of experiments to evaluate the effectiveness of our approach. This section details the experimental setup, baseline comparisons, implementation details, and evaluation metrics.

A. Experimental Setup

Our experiments are carried out in the AI2-THOR environment [13], focusing on the four kitchen scenes (FloorPlan1–FloorPlan4) depicted in Figure 1. Each scene provides a unique set of challenges due to variations in layout, object types, and spatial arrangements.

The agent is tasked with navigating to and interacting with a target object specified at the beginning of each episode. The target objects vary and include items like *Mug, Apple, Knife*, and so on. The agent must locate the object, navigate to it, and perform an appropriate interaction (e.g., *PickupObject*).

B. Baseline

We compare our perception-enhanced agent against a Standard RL Agent using raw RGB observations without advanced perception models, trained with the same RL algorithm and hyperparameters. The baseline allows us to assess the benefits of integrating advanced perception models over traditional observation modalities.

C. Training Procedure

The agent is trained by interacting with the environment, collecting trajectories $\tau = \{(s_t, a_t, r_t, s_{t+1})\}$, and updating the policy network using gradient ascent on the PPO objective. The perception models (SAM and YOLOv5) are kept fixed during training to leverage their pre-trained capabilities without incurring the computational cost of fine-tuning.

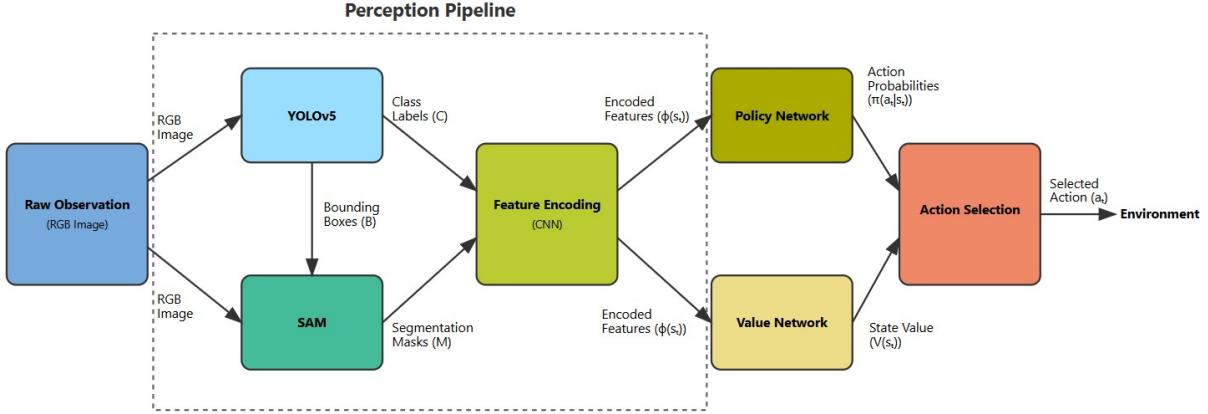


Fig. 2: System architecture integrating SAM and YOLOv5 with the RL agent. The perception pipeline processes the raw observation to extract rich features, which are then used by the policy network to make action decisions.

D. Implementation Details

All agents are implemented using PyTorch [28]. The perception-enhanced agent utilizes pre-trained SAM and YOLOv5 models, with inference optimized for real-time performance. We ensure consistent training conditions across all agents, including the same random seeds, to facilitate fair comparisons.

The hyperparameters are set as follows:

- **Learning Rate:** 3×10^{-4}
- **Discount Factor (γ):** 0.99
- **GAE Parameter (λ):** 0.95
- **PPO Clip Parameter (ϵ):** 0.2
- **Batch Size:** 64
- **Epochs per Update:** 4

Experiments are conducted on a workstation with an NVIDIA RTX 3080 GPU and 64 GB of RAM. Resource utilization is monitored to assess the computational overhead introduced by the perception models.

Agents are trained for 1×10^6 time steps, with performance evaluated at regular intervals. We use early stopping if the performance plateaus. Training curves are recorded for analysis of learning dynamics.

During training and evaluation, we collect detailed logs of the agents' actions, observations, rewards, and internal states. This data supports quantitative analysis and aids in diagnosing any performance issues.

To ensure the robustness of our results, we repeat each experiment five times with different random seeds and report the mean and standard deviation of the metrics.

E. Evaluation Metrics

We evaluate the agents using the following metrics:

- **Success Rate (%):** The percentage of episodes in which the agent successfully interacts with the target object.
- **Average Cumulative Reward:** The mean total reward accumulated per episode.

- **Navigation Efficiency (%):** The ratio of the optimal path length to the actual path length taken by the agent, averaged over successful episodes, multiplied by 100.
- **Interaction Efficiency:** The number of interaction attempts made before a successful interaction, measuring the agent's ability to correctly perform tasks without unnecessary actions.

V. RESULTS

In this section, we present the results of our experiments, comparing the performance of the perception-enhanced agent with the baseline agent. We provide both quantitative metrics and qualitative analysis.

A. Quantitative Analysis

Table I summarizes the performance metrics for all agents.

TABLE I: Performance comparison of agents across all four environments (floor plans). Results are averaged over five runs with standard deviations.

Metric	Perception-Enhanced	Baseline
Success Rate (%)	73.5 ± 2.1	48.2 ± 4.5
Avg. Cumulative Reward	136.4 ± 5.6	81.1 ± 9.3
Navigation Efficiency (%)	82.1 ± 1.9	61.7 ± 3.1
Interaction Efficiency	1.2 ± 0.1	2.1 ± 0.3

1) *Success Rate:* The perception-enhanced agent achieves a significantly higher success rate compared to the baseline, with an improvement of 52.5%. This indicates the agent's enhanced ability to locate and interact with target objects effectively.

2) *Average Cumulative Reward:* Our agent attains a 68.2% increase in average cumulative reward over the baseline, reflecting more efficient policies and better adherence to the designed reward function.

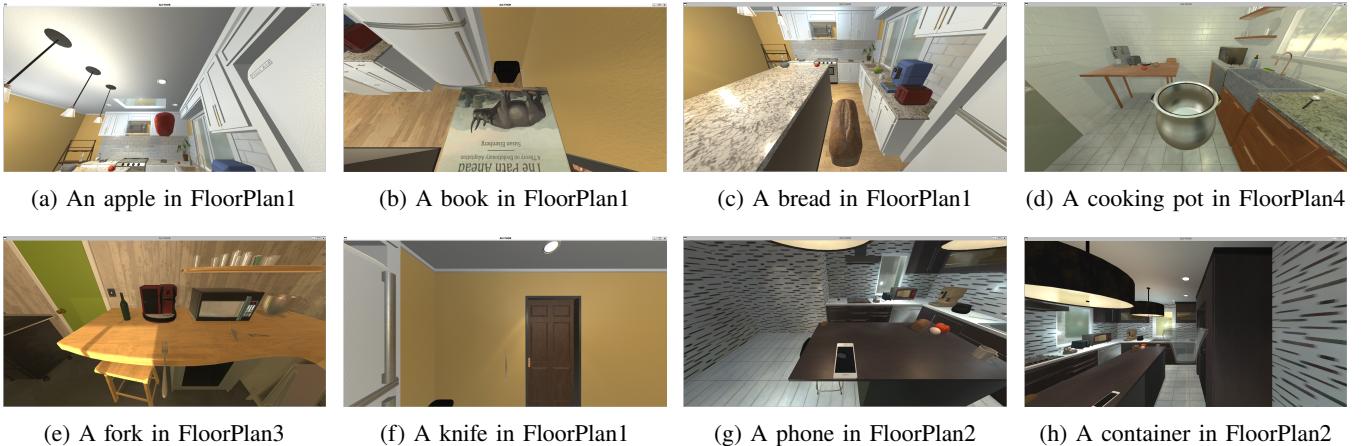


Fig. 3: Examples of various objects of different shape, size, color, mass, and opacity in different kitchen environments. These images showcase the diversity of objects the perception-enhanced agent encounters in its training, highlighting its ability to identify, navigate to, and interact with target objects efficiently.

3) Navigation Efficiency: An increase of 33.1% in navigation efficiency demonstrates that the perception-enhanced agent follows paths closer to the optimal route, reducing unnecessary movements.

4) Interaction Efficiency: The agent requires fewer interaction attempts to successfully complete tasks, indicating better decision-making regarding when and how to interact with objects.

B. Qualitative Analysis

As shown in Figure 3, the perception-enhanced agent effectively identifies and interacts with target objects with varied properties like shape, size, color, mass, opacity and so on, demonstrating versatility in handling different objectives in diverse environments. This capability highlights the robustness of the integrated perception models (SAM and YOLOv5), enabling the agent to navigate efficiently and interact precisely, regardless of the object’s physical attributes or spatial configuration. The agent’s adaptability across all four environments reinforces the effectiveness of combining advanced perception with reinforcement learning for complex interaction tasks.

VI. DISCUSSION

Our results demonstrate that integrating vision foundation models with reinforcement learning agents can significantly enhance performance in object interaction tasks. The perception-enhanced agent outperforms baseline agent across all metrics, indicating that advanced perception contributes to more effective decision-making.

A. Impact of Advanced Perception

The use of SAM and YOLOv5 provides the agent with detailed semantic and spatial information, enabling it to better understand the environment. This enriched perception allows the agent to:

- **Disambiguate Objects:** Correctly identify target objects among similar items.

- **Plan Efficient Paths:** Utilize spatial awareness to navigate around obstacles.
- **Optimize Interactions:** Position itself optimally for interaction tasks, reducing failed attempts.

B. Policy Learning

The agent’s improved performance suggests that the policy network effectively leverages the enhanced perception. The feature representation $\phi(s_t)$ captures essential information without overwhelming the network, highlighting the importance of careful feature design.

C. Computational Considerations

Although the integration of large models increases computational demands, our optimizations mitigate the impact. The acceptable frame rate achieved suggests feasibility for real-time applications.

D. Limitations

Our experiments focused on specific environments and tasks. The agent’s ability to generalize to new settings or handle unexpected changes remains an area for exploration. Incorporating mechanisms for continual learning or domain adaptation could enhance robustness.

VII. CONCLUSION

We have presented a novel approach that integrates vision foundation models with reinforcement learning to enhance object interaction capabilities in simulated environments. By combining SAM and YOLOv5 with a PPO agent in the AI2-THOR environment, we demonstrate significant improvements over baseline agent in success rate, efficiency, and overall performance.

Our work highlights the potential of leveraging advanced perception models to enhance RL agents, enabling more effective interactions with complex environments. The positive results highlight the value of integrating state-of-the-art vision models into robotics applications.

We believe that our approach opens up new possibilities for developing sophisticated autonomous agents capable of operating in diverse and challenging settings.

VIII. FUTURE WORK

Building on our findings, future research can focus on enhancing the agent’s generalization to unseen environments via domain randomization or transfer learning, reducing computational overhead through model optimization techniques, and exploring real-world deployment by bridging the sim-to-real gap. Additionally, extending the agent’s capabilities to handle multi-task learning and incorporating natural language instructions could further enhance its versatility.

REFERENCES

- [1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.
- [2] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.
- [3] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.
- [4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [5] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2695–2704.
- [6] 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*, 2017, pp. 2616–2625.
- [7] D. Kalashnikov, A. Irpan, P. Pastor *et al.*, “Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation,” in *Conference on Robot Learning*, 2018, pp. 651–673.
- [8] M. Andrychowicz, B. Baker *et al.*, “Learning dexterous in-hand manipulation,” *International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.
- [9] D. Shao, Y. Zhao, Y. Wang, J. Wu, Y. Zhu, and J. J. Lim, “Concept2robot: Learning manipulation concepts from instructions and human demonstrations,” in *Proceedings of the International Conference on Robotics and Automation*, 2020, pp. 2768–2774.
- [10] S. Tellez, R. A. Knepper, A. Li, N. Roy, and D. Rus, “Robots that use language,” *Communications of the ACM*, vol. 63, no. 9, pp. 66–74, 2020.
- [11] G. Jocher *et al.*, “YOLOv5 by Ultralytics,” <https://github.com/ultralytics/yolov5>, 2020.
- [12] A. Kirillov, E. Mintun, N. Ravi *et al.*, “Segment anything,” *arXiv preprint arXiv:2304.02643*, 2023.
- [13] E. Kolve, R. Mottaghi *et al.*, “Ai2-thor: An interactive 3d environment for visual ai,” in *arXiv preprint arXiv:1712.05474*, 2017.
- [14] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.
- [15] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.
- [16] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.
- [17] J. Schulman *et al.*, “Trust region policy optimization,” in *International Conference on Machine Learning*, 2015, pp. 1889–1897.
- [18] N. Heess *et al.*, “Emergence of locomotion behaviours in rich environments,” *arXiv preprint arXiv:1707.02286*, 2017.
- [19] B. Zhao, Z. Ding, X. Chen *et al.*, “Sim-to-real transfer in deep reinforcement learning for robotics: A survey,” *International Symposium on Robotics Research*, 2020.
- [20] J. Zhang *et al.*, “Deep reinforcement learning for robotics: A survey,” *International Conference on Advanced Robotics and Mechatronics*, pp. 422–427, 2018.
- [21] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [22] A. Mousavian *et al.*, “Visual representations for semantic target driven navigation,” in *International Conference on Robotics and Automation*, 2019, pp. 8846–8852.
- [23] D. S. Chaplot *et al.*, “Object goal navigation using goal-oriented semantic exploration,” in *Advances in Neural Information Processing Systems*, 2020, pp. 4247–4258.
- [24] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, “Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6629–6638.
- [25] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.
- [26] M. Savva *et al.*, “Habitat: A platform for embodied ai research,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9339–9347.
- [27] F. Xia *et al.*, “Gibson env: Real-world perception for embodied agents,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9068–9079.
- [28] A. Paszke *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026–8037, 2019.