Optical Illusion in Reinforcement Learning and Interactive Robot

Khang V. Huynh
Department of Computer Science
The University of Virginia - Charlottesville
United States of America
szq2sj@virginia.edu

Danya Balakumaran Meenakumari
Department of Electrical and Computer Engineering
The University of Virginia - Charlottesville
United States of America
yka2dk@virginia.edu

Abstract

Robots relying on vision-based systems often encounter challenges when interpreting visual inputs distorted by optical illusions, which can lead to inaccurate perception and suboptimal actions. This study explores how reinforcement learning (RL) can be leveraged to enable robots to effectively interpret and respond to such deceptive visual scenarios. By integrating Vision-Language Models (VLMs) with RL, we aim to evaluate the system's ability to adapt and maintain accuracy when exposed to optical illusions. The final result show some inclination toward to claim that illusion actually played a part in how an agent work. However, the level of significance of the claim are still questionable.

Key metrics : perception accuracy, decision-making efficiency, and adaptability

1 Introduction

Optical illusion is a visual phenomena where the brain's interpretation of an image differs from reality. It can vary in colors, shapes, patterns tricking one's perception from seeing it and misjudging the images in various aspects.Incorporating Optical illusion with reinforcement learning could understand these illusions and provide novel insights on how robots interpret and interact with the environment.By exploring this method, we can able to understand the deeper knowledge about the decision-making, adaptability and interaction of robots.For example, a cube placed in the checkerboard environment creating an illusion-ed space, where the robot can misjudge on locating the object to grasp it and leading to incorrect grip to the object or missing the object

2 Related Work

Though there are not much of documentations on robots with optically illusion-ed environment, but there are some relevant works of altering/manipulating visual inputs and record the robot's behaviour towards them.

Multi-sensory fusion like combining visual inputs with other sensory data allows robots to detect discrepancies that may arise from visual illusions. From Shridhar et al. (2022) [1] , Robots use both touch and vision to recognize objects, making them less likely to be deceived by purely visual cues.Multi-sensory fusion demands complex processing, which raises computational load and latency, affecting real-time response.

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

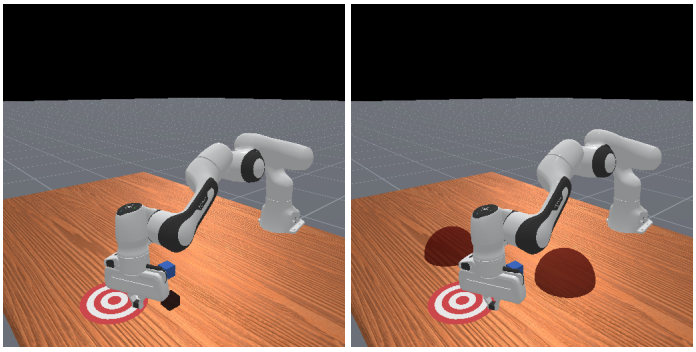

Figure 1: Illusion - From left to right: Transparent Box Illusion to Transparent Sphere as Target

The use of simulation to real-world transfer (Sim2Real) techniques as mentioned in Tobin et al.'s (2017)[2] , can make robots resilient to visual perturbations, such as shadows or reflections that can mimic illusions. In specific, Domain randomization adds variability to simulated environments so robots learn robust visual representations. However, factors such as lighting conditions, and material properties in real life are far more intricate, which can lead to misalignment between what the robot learns in simulation and reality.

From Huang et al. (2017) [3] , This paper can serve as a foundation for studying optical illusions in reinforcement learning. Just as adversarial attacks exploit vulnerabilities in neural network policies to alter their perception and decisions, optical illusions can challenge how vision-based models interpret visual stimuli. Both scenarios highlight the gap between machine perception and robust, human-like understanding.

Gauss's work on predicting celestial body motions from limited or distorted data parallels how reinforcement learning (RL) models handle visual inputs affected by optical illusions. Just as Gauss used methods like least squares to refine predictions, RL can be used to adjust robot actions in response to misleading visual data, improving decision-making and adaptability in distorted environments [4].

In the context of optical illusions in reinforcement learning, [5] Lagrange's method of simplifying complex systems is analogous to how RL models can be trained to adapt and make accurate decisions despite distorted visual inputs from illusions, improving robotic adaptability in deceptive environments .

In summary some relevant papers with , Liu et al. (2022) focus on learning generalizable manipulation skills, which could be crucial for adapting robots to handle visual distortions [6]. Zhan et al. (2022) [7] explore how pre-training and data augmentation improve visual control, potentially enhancing RL models' resilience in robotic manipulation .Lastly Calandra et al. (2018) [8]. demonstrate how combining vision and touch helps robots grasp objects, which could help mitigate the effects of visual manipulation.

Optical illusions in reinforcement learning challenges agents to adapt to distorted perceptions, improving decision-making and robustness. They help test the agent's ability to generalize, adapt, and handle misleading sensory inputs and can be extended to real time applications such as for the robots trained to handle illusions aid in AR/VR-based training scenarios.

2.1 Problem Setup

The experimental environment is a simulated robotic task setup where a robotic arm is required to push a target box toward a predefined target zone. The baseline task involves one box (the target box) and a clear target zone, with no additional distractors. The agent uses a depth camera and an RGB sensor to perceive the environment, and its policy is optimized using reinforcement learning. To introduce perceptual challenges, two experimental setups are designed:

- Transparent Box (Figure 1) as a Bait Objective: In this setup, a semi-transparent box is added to the environment. This box is placed in close proximity to the target box and is designed to visually resemble it. The bait box is strategically positioned to lure the arm into interacting with it, leading to incorrect execution of the task if the agent fails to identify the real target box. The transparency of the box further adds to the perceptual ambiguity, making it harder for the agent to distinguish between the bait and the real objective.
- Dim Spheres as Illusionary Targets (Figure 1): In this setup, two dimly visible spheres are placed near the actual target zone. These spheres are positioned to mislead the depth camera, creating an illusion of a closer or farther target. This scenario tests the agent's ability to accurately interpret spatial relationships when depth cues are corrupted by the presence of additional objects that mimic the target's visual signature. The spheres are designed with low opacity and are spatially offset to simulate a real-world setting where similar objects could interfere with depth perception.

2.2 Mathematical Representation

The task is modeled as a Markov Decision Process (MDP) with distortions in the agent's observation space. The MDP is defined by the tuple

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, \mathcal{O}, \mathcal{D}),$$

where:

- $\mathcal{S}$: The state space, representing the true configurations of the environment.
- $\mathcal{A}$: The action space, representing the agent's physical actions.
- $\mathcal{P}(s'|s, a)$: The transition probability function, defining the likelihood of transitioning to state s' after taking action a in state s.
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: The reward function, providing scalar feedback based on the agent's actions and their outcomes.
- $\gamma \in [0, 1]$: The discount factor, weighing future rewards relative to immediate ones.
- $\mathcal{O}$: The observation space, representing what the agent perceives as $\tilde{o}_t = \mathcal{D}(s_t)$, where $\mathcal{D}$ is a distortion function.
- $\mathcal{D} : \mathcal{S} \to \tilde{\mathcal{O}}$: The distortion function, introducing ambiguity or errors in perception.

The agent's objective is to learn a policy $\pi(a|\tilde{o})$ that maximizes the cumulative discounted reward:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right].$$

2.3 Transparent Box as a Bait Objective

In the first setup, the agent perceives two objects: the true target box and a transparent bait box. The distorted observation is given by:

$$\tilde{o}_t = \{(x_{\text{real}}, y_{\text{real}}, y_{\text{real}} d_{\text{real}}), \alpha(x_{\text{bait}}, y_{\text{bait}}, z_{\text{bait}}, d_{\text{bait}})\},$$

where (x, y, z, d) represent the spatial coordinates and depth of the objects, and α ($0 < \alpha < 1$) is a transparency factor reducing the visibility of the bait box. The agent must correctly prioritize the real target, which becomes challenging due to the presence of the visually ambiguous bait.

2.4 Dim Spheres as Illusionary Targets

In the second setup, two dim spheres are introduced near the target zone. These spheres distort the depth camera's perception, with the distorted observation given by:

$$\tilde{o}_t = \{d_{\text{target}} + \epsilon_{\text{target}}, d_{\text{sphere}_1} + \epsilon_1, d_{\text{sphere}_2} + \epsilon_2\},$$

where d_{target} is the true depth of the target zone, d_{sphere_i} $(i = 1, 2)$ are the depths of the spheres, and $\epsilon_{\text{target}}, \epsilon_1, \epsilon_2 \sim \mathcal{N}(0, \sigma^2)$ represent noise terms simulating sensor inaccuracies. The agent calculates the perceived depth using a weighted combination:

$$\tilde{d}_{\text{effective}} = \frac{\sum_{i=1}^{2} w_i d_{\text{sphere}_i} + w_{\text{target}} d_{\text{target}}}{\sum_{i=1}^{2} w_i + w_{\text{target}}},$$

where $w_{\text{target}}, w_1, w_2$ are weights representing the agent's confidence in each depth cue.

—

These formulations challenge the agent to correctly interpret ambiguous observations, revealing limitations in its perception and decision-making capabilities. Future work can focus on improving policies to overcome these distortions.

2.5 Robot Arm training and Agent Usage

The ManiSkill framework provides a suite of environments and tools designed to facilitate the training and evaluation of robotic manipulation skills. Within this framework, the Proximal Policy Optimization (PPO) and Soft Actor Critic (SAC) baselines serve as a foundational reinforcement learning algorithm, offering a standardized approach for training agents across various manipulation tasks. The code was obtained from research work presented as ManiSkill baselines [9]

2.5.1 Training Process of the PPO Agent

The PPO implementation in ManiSkill is adapted from the CleanRL repository, emphasizing clarity and simplicity in its single-file design. This design choice ensures that the code is accessible and easy to understand, making it suitable for both educational purposes and practical applications.

The training process begins with the initialization of multiple parallel environments, leveraging GPU acceleration to handle a large number of environments simultaneously. This parallelization is crucial for efficient data collection and accelerates the training process. In our case, for PushCube environment, we trained in 256 environment concurrently.

The agent's policy and value functions are parameterized using neural networks. The policy network outputs the mean and standard deviation of a Gaussian distribution, from which actions are sampled. The value network estimates the expected return from a given state, providing a baseline for advantage estimation. Both networks are trained using data collected from the parallel environments.

During training, the agent interacts with the environments, collecting trajectories of states, actions, rewards, and next states. These trajectories are used to compute the advantage function, which estimates the relative value of actions compared to the current policy. The policy is then updated to maximize the expected advantage while ensuring that the updates do not deviate significantly from the current policy, a mechanism enforced by the PPO clipping technique. This approach balances exploration and exploitation, allowing the agent to learn effective behaviors without overfitting to specific trajectories.

The training process also incorporates several hyperparameters, such as the learning rate, discount factor (gamma), and the number of epochs per update. These hyperparameters are tuned to ensure stable and efficient learning. For example, in the PushCube-v1 task, we used the fine-tuned hyperparameters that was provided by ManiSkill baselines. We trained the models with three set of parameters suitable for 100.000 timesteps to 1.000.000 timesteps with a jump of 100000 timestep. The ideas behind this is to test whether the quality of the agent will affect how much the agent will react to visual illusion

2.5.2 Training Process of the SAC agent

The Soft Actor-Critic (SAC) algorithm stands out as a prominent off-policy reinforcement learning method, particularly effective in continuous control tasks. This report delves into the SAC baseline

provided by ManiSkill, elucidating its training methodology and its role as a foundational benchmark within the platform.

Soft Actor-Critic (SAC) Overview

SAC is an off-policy actor-critic algorithm that integrates the maximum entropy framework to enhance exploration and robustness. By concurrently maximizing expected return and policy entropy, SAC encourages exploration of diverse behaviors, thereby preventing premature convergence to suboptimal policies. This characteristic renders SAC especially suitable for complex robotic manipulation tasks, where the exploration of various strategies is crucial for effective learning. ARXIV

Training Methodology in ManiSkill's SAC Baseline

The SAC baseline in ManiSkill is tailored to address the platform's diverse manipulation tasks. The training process encompasses several key components:

- Environment Interaction: The agent interacts with the specified environment (e.g., "PushCube-v1") to collect transition data, which includes the current state, executed action, resulting next state, and received reward.
- Replay Buffer: Collected transitions are stored in a replay buffer, enabling the agent to learn from a broad spectrum of past experiences. This mechanism enhances sample efficiency by allowing multiple updates from the same data.
- Policy and Value Updates: The agent updates its policy and value functions by sampling mini-batches from the replay buffer. The policy aims to maximize a trade-off between expected return and entropy, while the value function estimates the expected return of state-action pairs.
- Automatic Temperature Adjustment: SAC incorporates an automatic adjustment of the temperature parameter, which balances the importance of the entropy term relative to the reward. This adjustment ensures stability and adaptability across various tasks.
- Evaluation: Periodic evaluations are conducted to assess the agent's performance. Metrics such as cumulative reward and task success rate are recorded to monitor learning progress and facilitate comparisons with other baselines.

In this algorithm, we also obtained 10 different models with 10 different number of timestep from 100000 to 1000000 timesteps

3 Experiment Setup and Result

The experiments were performed using two reinforcement learning baselines: Proximal Policy Optimization (PPO) and Soft Actor-Critic (SAC). For each algorithm, ten models were trained with varying total timesteps, ranging from 100,000 to 1,000,000 in increments of 100,000. This setup allows us to analyze the effect of training duration on the agents' performance under visually challenging conditions.

3.1 Experiment 1

In the first experiment, a semi-transparent box was introduced into the environment as a distractor, or "bait," to test the agent's ability to prioritize the correct target. The agent's task was to push a designated target box into a predefined goal zone. The bait box was placed closer to the agent's initial position than the target box, leveraging the agent's tendency to prioritize nearer objects. The bait's transparency reduced its visual prominence, adding ambiguity to the scene and forcing the agent to rely on features other than proximity to identify the correct target.

Each of the 10 models for both PPO and SAC was trained for a specific total timestep, ranging from 100,000 to 1,000,000. After training, the models were tested across 1000 trials in the modified environment with the bait box. One key performance metrics is task success rate. This metrics

allowed us to evaluate how training duration and algorithm choice influenced the agents' ability to correctly identify and interact with the target box.

3.1.1 Experiment 2

In the second experiment, two dimly visible spheres were added near the goal zone to distort the agent's depth perception. The spheres were placed at varying depths relative to the target zone, creating conflicting depth cues. The agent's task remained the same: to push the target box into the goal zone. The dim spheres were designed to simulate real-world challenges such as shadows or reflections, which can mislead an agent's perception.

Similar to the first experiment, 10 models of both PPO and SAC were trained for different total timesteps, ranging from 100,000 to 1,000,000. After training, the models were evaluated on 100 trials in the modified environment with the dim spheres. Key metrics for this experiment included task success rate, depth estimation error (mean squared error between the agent's predicted and true depth values for the target zone), and action error rate (percentage of actions influenced by the spheres' visual cues). These metrics quantified the impact of the visual distortion on the agents' decision-making and control accuracy.

3.1.2 How do We Determine which Factor Contribute the Unsuccessful case

To be able to see whether the task was "naturally cannot be completed" or "it was due to the effect of the illusion", we will measure the final destination of the grip in compared to the cube and "fake" cube. If the former is greater than the latter, we would determine that the arm has determined that the transparent cube as its designated object and indeed has been misled by the illusion.

For experiment 2, the procedure would be the same where we calculate the distance between the cube in comparison to the target and the two baits. If the distance to either the bait are smaller than the distance to the true target, than illusion has been the major factor contribute the unsuccesful case.

3.1.3 Result from Training Procedure

After finished obtaining a total of 20 models for both PPO and SAC, we randomized a 1000 initial observation and observe to calculate the success rate of the agent. This might resulted in a slightly lower success rate that what the training process actually report.

The two plots illustrate the task success rates of PPO (Proximal Policy Optimization) and SAC (Soft Actor-Critic) agents across a range of training timesteps, from 100,000 to 1,000,000. In the Figure 2, the success rate shows a steady increase as the total timesteps increase, starting at 24.5% for 100,000 timesteps and peaking at 85.4% for 1,000,000 timesteps. This indicates that PPO agents benefit significantly from longer training durations, learning more effective policies with increased exposure to the environment. The growth curve suggests that PPO's on-policy nature enables consistent improvements as training progresses, though the increase slows slightly at higher timesteps.

In contrast, Figure **??** exhibits a more fluctuating trend in success rates. While SAC agents also improve with more timesteps, the progression is less smooth compared to PPO. The success rate starts at 11.2% for 100,000 timesteps and climbs to 75.5% for 1,000,000 timesteps, with notable dips at intermediate timesteps (e.g., 300,000). This variability could be attributed to SAC's off-policy nature and reliance on replay buffers, which may introduce instability in some configurations. Nevertheless, SAC's entropy-maximization mechanism supports broader exploration, allowing it to eventually achieve competitive performance. Overall, the plots highlight the differing learning dynamics of PPO and SAC under varying training durations.

3.1.4 Result for PPO

Figure 3 provided two plots that gives out insight into the performance of PPO against cube and sphere illusions. The first plot, titled "Total Task Outcome Distribution Across Timesteps for PPO - Experiment 1", depicts PPO's ability to handle the cube illusion scenario, where a semi-transparent

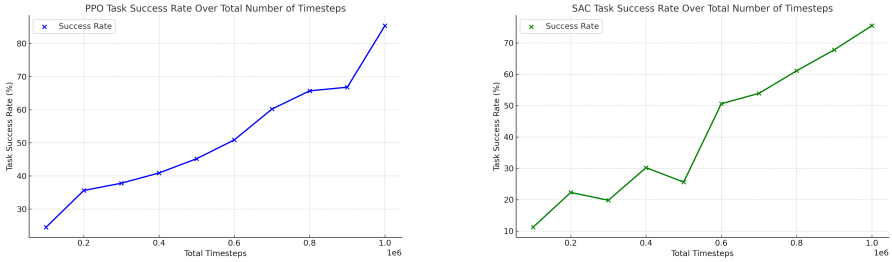

Figure 2: Task Success Rate at the end of the Training Cycles

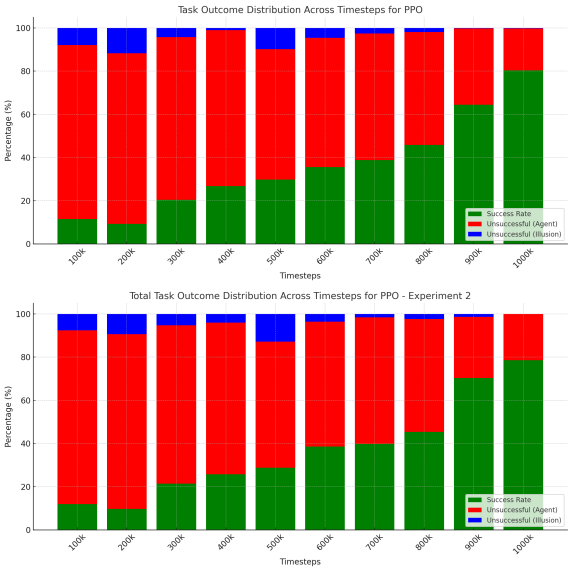

Figure 3: PPO Algorithm Experiment Results

cube is introduced as a distractor. Initially, when training timesteps are low (e.g., 100k), the success rate (green segment) is minimal, and failures due to agent-related errors (red segment) dominate. This shows that PPO initially struggles to distinguish the target from the distracting cube. As training progresses and timesteps increase to 1M, the success rate rises steadily, indicating that PPO gradually learns to ignore the distracting cube and focus on the actual target. Interestingly, the failures caused by the illusion itself (blue segment) remain small throughout the experiment, suggesting that while PPO may initially falter due to decision-making inefficiencies, it does not significantly misinterpret the illusion as the goal.

In contrast, the second plot, titled "Total Task Outcome Distribution Across Timesteps for PPO - Experiment 2", highlights PPO's performance in the sphere illusion scenario. This experiment introduces dimly visible spheres near the target, which distort depth perception and create a more challenging perceptual environment. In this case, the success rate starts low at shorter timesteps and improves more slowly compared to the cube illusion scenario. The blue segment, representing failures caused by the illusion, is noticeably larger in this experiment, especially at lower timesteps. This indicates that PPO is more susceptible to misjudging the depth-related visual cues introduced by the spheres. As training timesteps increase, the success rate improves, and illusion-induced failures gradually decrease, though it takes longer for PPO to achieve a comparable level of robustness to the cube illusion experiment.

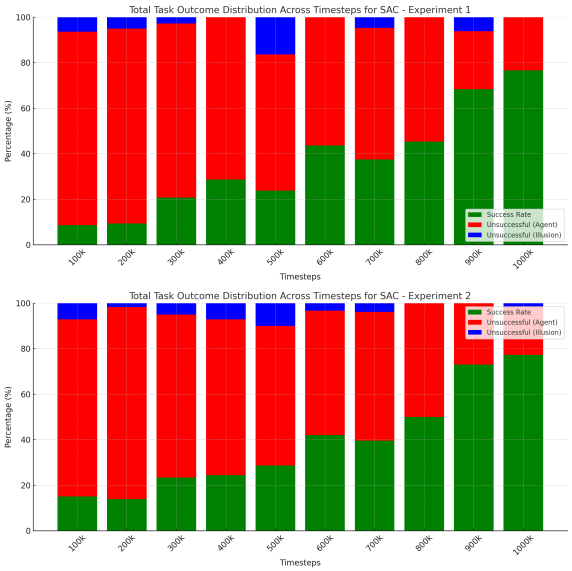

Figure 4: SAC Algorithm Experiments Results

3.1.5 Result for SAC

From Figure 4, the first plot, titled "Total Task Outcome Distribution Across Timesteps for SAC - Experiment 1", represents the SAC agent's performance under the cube illusion scenario, where a semi-transparent cube acts as a distractor. Initially, at lower timesteps (100k–300k), the success rate (green segment) is low, with most failures attributable to agent-related errors (red segment). This indicates that SAC, in its early training stages, struggles to differentiate between the target and the distracting cube. However, as training progresses and timesteps increase, the success rate grows steadily, showing SAC's capability to learn and adapt. Interestingly, the illusion-related failures (blue segment) are minimal across all timesteps, highlighting that SAC rarely confuses the cube itself with the target but instead struggles with broader decision-making challenges caused by the distractor.

The second plot, "Total Task Outcome Distribution Across Timesteps for SAC - Experiment 2", illustrates SAC's performance under the sphere illusion scenario, where dimly lit spheres distort depth perception. Here, the success rate (green segment) also starts low at earlier timesteps but increases significantly with extended training. However, compared to the cube illusion, the illusion-related failures (blue segment) are more pronounced in this scenario, particularly at earlier timesteps. This indicates that SAC agents are initially more susceptible to depth-related visual ambiguities. Over time, the illusion-related failures decrease, but this reduction is slower than in Experiment 1, suggesting that depth distortions present a greater challenge for SAC than dealing with a distracting object.

3.2 Conclusion

The analyses of PPO and SAC under cube and sphere illusion scenarios reveal distinct learning dynamics and challenges for each algorithm, highlighting their strengths and areas for improvement.

PPO demonstrates steady improvement in handling both types of visual illusions with extended training. The algorithm adapts more efficiently to the cube illusion, as seen in the quicker rise in success rates and the minimal impact of illusion-related failures. However, PPO struggles more with depth-related ambiguities introduced by the sphere illusion, where failures caused by the illusion persist longer and require significantly more training to overcome. This suggests that while

PPO excels in scenarios involving clear distractors, it faces limitations in environments where depth perception plays a critical role.

SAC, on the other hand, shows a similar pattern of improvement across both experiments but with a slower initial adaptation phase. SAC's ability to handle cube illusions improves with training, with minimal failures due to misinterpreting the illusion itself. However, the algorithm encounters greater difficulty with the sphere illusion, where depth-related ambiguities cause higher rates of illusion-related failures early in training. Despite these challenges, SAC's off-policy learning mechanism enables it to achieve competitive performance at higher timesteps, indicating its capacity to generalize and adapt given sufficient training.

In comparing the two algorithms, PPO shows faster adaptation to simpler distractor scenarios but struggles more with complex perceptual ambiguities. SAC, while slower to adapt initially, demonstrates greater robustness over time, especially in environments requiring nuanced decision-making and generalization. These findings suggest that SAC's off-policy approach and exploration-driven strategies may offer advantages in handling visually challenging tasks, while PPO's simplicity and consistency make it a strong contender for tasks with less complex visual ambiguities.

Both algorithms exhibit clear potential for improvement, particularly in enhancing their ability to process and interpret depth-related cues. Future work could focus on integrating advanced perception mechanisms, such as multi-modal sensor fusion or attention-based architectures, to further enhance their robustness against visual illusions. This would enable both PPO and SAC to better tackle real-world scenarios where visual ambiguities are prevalent.

4 Challenge & Future Direction

One of the key challenges in evaluating the robustness of PPO and SAC under visual illusions lies in the issue of spurious correlations due to the non-deterministic nature of these algorithms. Reinforcement learning agents rely heavily on stochastic policies and randomness in exploration, which can result in inconsistent behaviors across different training runs. This stochasticity complicates the interpretation of results, as observed successes or failures may not consistently reflect the true impact of the illusion. For example, an agent may appear to handle an illusion effectively in some runs while failing in others, raising questions about whether the observed behaviors stem from robust learning or chance correlations in the training data. This inherent randomness makes it challenging to distinguish genuine adaptation from spurious correlations, particularly when evaluating the effects of specific environmental modifications such as cube and sphere illusions.

Another challenge lies in determining the significance of the illusion's impact, even when failures can be attributed to it. While the blue segment in the bar plots represents illusion-related failures, accurately quantifying the effect of the illusion requires disentangling it from broader agent-related errors. In some cases, illusion-induced failures might occur due to indirect influences, such as compounding decision errors from prior actions, rather than the illusion itself. This makes it difficult to measure the true degree to which the illusion hinders agent performance. Developing robust metrics to evaluate the direct contribution of illusions to agent failures is a critical yet unresolved challenge in this domain.

Future work can extend these experiments by incorporating real optical illusions into testing environments, moving beyond simulated visual challenges like transparent cubes and dim spheres. Real optical illusions, such as those involving complex patterns, ambiguous shapes, or misleading depth cues, would provide a more realistic benchmark for assessing an agent's ability to handle perceptual ambiguity. For instance, introducing illusions like the Ames room or Ponzo illusion could test the limits of an agent's visual processing capabilities and its reliance on depth perception. Such tests would offer valuable insights into how well reinforcement learning algorithms generalize to complex real-world scenarios with inherent perceptual ambiguities.

Another promising direction is the creation of a comprehensive framework for training agents against environmental ambiguity, with a particular focus on visual illusions. This framework could

involve systematically introducing controlled levels of ambiguity into training environments, allowing agents to learn robust policies that are less prone to failures caused by illusions. Techniques like domain randomization and adversarial training could be employed to expose agents to a diverse set of ambiguous scenarios, improving their ability to generalize beyond the specific conditions encountered during training. For example, an agent could be trained with varying object transparencies, conflicting depth cues, or shifting light conditions, enabling it to learn robust representations of the environment that are resilient to such perturbations.

Additionally, multi-modal learning approaches, such as combining visual inputs with tactile or depth sensors, could be integrated into this framework to help agents resolve ambiguities. By leveraging information from multiple sensory modalities, agents could mitigate the impact of visual illusions and make more reliable decisions. Moreover, the framework could incorporate attention mechanisms or feature attribution methods to identify and prioritize relevant environmental features, further enhancing the agent's robustness against ambiguity.

By addressing these challenges and pursuing these future directions, reinforcement learning research can advance toward developing agents capable of operating reliably in environments filled with visual ambiguities and illusions, ultimately improving their real-world applicability.

5 Github Link

https://github.com/HVKHCM/ILR.git

References

[1] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[2] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30. IEEE, 2017.

[3] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[4] C. F. Gauss and C. H. Davis. Theory of the motion of the heavenly bodies moving about the sun in conic sections. *Gauss's Theoria Motus*, 76(1):5–23, 1857.

[5] J.-L. Lagrange. *Mécanique Analytique*. Desaint, Paris, 1788.

[6] K. Liu, Z. Wang, Y. Zhou, H. Wang, X. Zhao, Y. Xu, and X. Chang. An empirical study and analysis of learning generalizable manipulation skill in the sapien simulator. https://openreview.net/forum?id=STfl2VTNy-9, 2022. Accessed: 2024-12-01.

[7] A. Zhan, M. Jin, and D. Held. Learning visual robotic control efficiently with contrastive pretraining and data augmentation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5115–5122. IEEE, 2022.

[8] R. Calandra, A. Owens, D. Jayaraman, J. Lin, W. Yuan, J. Malik, E. H. Adelson, and S. Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018.

[9] H. Lab. Baselines algorithm. *UCSD*, 2024.