VISUAL HIDE AND SEEK

Anonymous authors
Paper under double-blind review

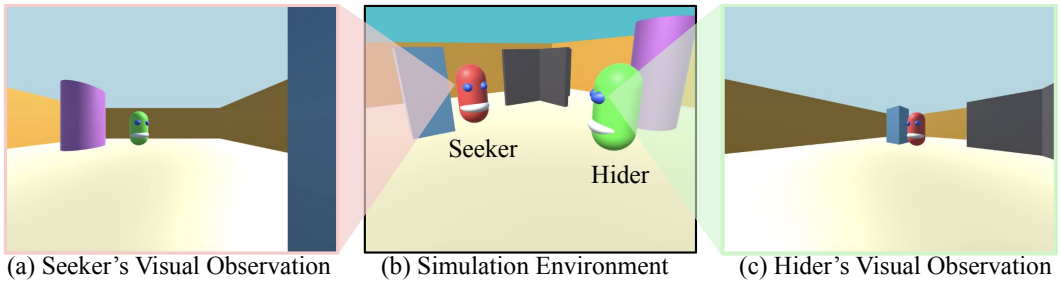
ABSTRACT

We train embodied agents to play Visual Hide and Seek where a prey must navigate in a simulated environment in order to avoid capture from a predator. We place a variety of obstacles in the environment for the prey to hide behind, and we only give the agents partial observations of their environment using an egocentric perspective. Although we train the model to play this game from scratch without any prior knowledge of its visual world, experiments and visualizations show that a representation of other agents automatically emerges in the learned representation. Furthermore, we quantitatively analyze how agent weaknesses, such as slower speed, effect the learned policy. Our results suggest that, although agent weaknesses make the learning problem more challenging, they also cause useful features to emerge in the representation.

1 INTRODUCTION

We introduce the task of *Visual Hide and Seek* where a neural network must learn to steer an embodied agent around its environment in order to avoid capture from a predator. We designed this game to mimic the typical dynamics between predator and prey. For example, we place a variety of obstacles inside the environment, which create occlusions that the agent can leverage to hide behind. We also only give the agents access to the first-person perspective of their three-dimensional environment. Consequently, this task is a substantial challenge for reinforcement learning because the state is both visual (pixel input) and partially observable (due to occlusions). Figure 1 illustrates the problem setup.

Our hypothesis, which our experiments suggest, is that learning to play this game will cause useful representations of multi-agent dynamics to emerge. We train the agent to navigate through its environment to maximize its survival time, which the model successfully learns to do. However, since we train the model from scratch to directly map pixels into action, the network can learn a representation of the environment and its dynamics. We probe the internal representations, and our experiments quantitatively suggest that the model automatically learns to recognize the viewpoint of the predator in order to hide best.



(a) Seeker's Visual Observation (b) Simulation Environment (c) Hider's Visual Observation

Figure 1: **Visual Hide-and-Seek:** We train models to play a game of visual hide and seek and analyze the dynamics that automatically emerge. We show that, although agent weaknesses make the learning problem more challenging, they collaterally encourage the learning of rich representations for the scene dynamics.

Embodied agents with extensive training experience are increasingly able to solve a large number of problems across manipulation, navigation, and game-playing tasks (Mnih et al., 2015; Gu et al., 2017; Zhu et al., 2017b; Silver et al., 2017; Kahn et al., 2018; Kalashnikov et al., 2018; Mirowski et al., 2018). Extensive work has demonstrated that, after learning with indirect supervision from a reward function, rich representations for their task automatically emerge (Bansal et al., 2017; Lowe et al., 2017; Liu et al., 2019; Jaderberg et al., 2019). We also observe similar characteristics. However, the precise process through which representations emerge often remains unclear (Rahwan et al., 2019; Shevlin & Halina, 2019). Therefore, we take a step back to investigate the underlying reasons *why* strategies emerge in the first place.

What intrinsic structure in the game, if any, caused this strategy to emerge? Or was the model just lucky in discovering these representations? We believe analyzing these fundamental questions will not only lead to more accurate, robust, and trustworthy models, but also provide insight to improve emergent representations from self-supervised learning. We quantitatively compare a spectrum of hide and seek games where we perturb the abilities of agents and the environmental complexity. Our experiments show that, although impediments to the agent, such as slower speed, make the learning problem more challenging, they also cause the model to learn more useful representations of its environmental dynamics. We provide empirical evidence for a "sweet spot" where the weakness is strong enough to cause useful strategies to emerge without derailing the learning process.

This paper makes three principal contributions to embodied agents. Firstly, we introduce the problem of visual hide-and-seek where an agent receives a partial observation of its visual environment and must navigate to avoid capture. Secondly, we empirically demonstrate that this task causes representations of other agents in the scene to emerge. Thirdly, we analyze the underlying reasons why these representations emerge, and show they are due to imperfections in the agent's abilities. The rest of this paper analyzes these contributions in detail. We plan to release all software, data, environments, and models publicly to promote further progress on this problem.

2 Related Work

Our paper contributes to a rapidly growing area to learn representations from indirect supervision. Early work has studied how features automatically emerge in convolutional networks for image recognition (Zhou et al., 2014; Zeiler & Fergus, 2014). Since direct supervision is often expensive to collect, there has been substantial work in learning emergent representations across vision (Doersch et al., 2015; Vondrick et al., 2018), language (Kottur et al., 2017; Radford et al., 2017), sound (Owens et al., 2016; Aytar et al., 2016), and interaction (Aytar et al., 2018; Burda et al., 2018). We also study how representations emerge. However we investigate the emergent dynamics in the two-player game of visual hide and seek. Through systematic analysis, we characterize why representations emerge, which we believe can refine the field's understanding of self-supervised learning dynamics.

Our work builds on research for embodied agents that learn to navigate and manipulate environments. Several recent works have created 3D embodiment simulation environment (Kolve et al., 2017; Brodeur et al., 2017; Savva et al., 2017; Das et al., 2018; Xia et al., 2018; Savva et al., 2019) for navigation and visual question answering tasks. To train these models, visual navigation is often framed as a reinforcement learning problem (Chen et al., 2015; Giusti et al., 2015; Oh et al., 2016; Abel et al., 2016; Bhatti et al., 2016; Daftry et al., 2016; Mirowski et al., 2016; Brahmbhatt & Hays, 2017; Zhang et al., 2017a; Gupta et al., 2017a; Zhu et al., 2017a; Gupta et al., 2017b; Kahn et al., 2018). Moreover, by incorporating multiple embodied agents into the environment, past work has explored how to learn diverse strategies and behaviors in multi-agent visual navigation tasks (Jaderberg et al., 2019; Jain et al., 2019). Our paper is also related to multi-agent reinforcement learning. For a full review of multi-agent reinforcement learning, please see (Panait & Luke, 2005; Bu et al., 2008; Tuyls & Weiss, 2012; Shoham et al., 2007; Hernandez-Leal et al., 2018).

This paper is concurrent to (Baker et al., 2019), and we urge readers to watch their impressive results on learning to play hide and seek games. However, there are a few key differences between the two papers that we wish to highlight. Firstly, in contrast to (Baker et al., 2019), we focus on hide and seek in partially observable environments. Our environment is three-dimensional, and agents only receive an egocentric two-dimensional visual input, which creates situations abundant with occlusions. Secondly, the input to our model is a visual scene, and not the state of a game engine. The learning problem is consequently very challenging because the model must learn perceptual

representations in addition to its policy. Our experiments suggest this happens, but the richness of the visual representation depends on the impediments to the model. Finally, we focus our investigation on analyzing the underlying reasons *why* different behaviors emerge, which we believe will refine the field's insight into self-supervised learning approaches.

3 HIDE AND SEEK

We first present our environment and learning problem, then describe our interventions to understand the cause of different emergent behaviors.

3.1 ENVIRONMENT AND LEARNING

We created a 3D simulation for hide-and-seek using the Unity game engine, which we use throughout this paper. There are two agents in this game: the hider and the seeker. Each agent receives only a first-person visual observation of the environment with 120-degree field of view, and navigates around the environment by selecting actions from a discrete set (move forward, move backward, turn left, turn right, stand still). The environment is a 14 x 14 square, and any real-value position inside the square is a valid game state. The speed of the hider is two units per step while the speed of the seeker is one and a half units per step. In contrast to other multiplayer games (Sukhbaatar et al., 2016; Moravčík et al., 2017; Lample & Chaplot, 2017; MacAlpine & Stone, 2017; Lowe et al., 2017; Foerster et al., 2018), the egocentric perspective of our problem setup makes this task very challenging. We place obstacles throughout the environment to create opportunities for occlusion.

Seeker Policy: We use a simple deterministic policy for the seeker, which is as follows. If the seeker can see the hider, move towards it. If the seeker cannot see the hider, move towards the hider's last known position. If the seeker still cannot find the hider, then it will randomly explore the environment by waypoints in a round-robin fashion. The game episode concludes when the seeker successfully "catches" the hider. We define catching the hider as a collision between the two agents.

Hider Policy: Given the environment and the seeker algorithm, we train a policy for the agent to hide. Given a first-person observation of the agent's environment, we estimate the action to execute with a convolutional neural network. We train the model to maximize its reward function using PPO (Schulman et al., 2017). Each time step, we provide a small reward ($+0.001$) for each living step, and a large negative reward (-1) once captured. We call this model **basic**.

Initialization: The starting positions of the hider and the seeker are randomly sampled on a continuous grid for each episode. We also specify their initial orientations so that they always start by looking at each other (even if there is an obstacle between them).

Our intention is that, by learning to hide, the model will learn features for the affordances of objects in its visual field, such as obstacles and other agents. We fit a linear regression from the learned features to classify different objects and game states, and we are often able to find a strong correlation, suggesting the emergence of useful features. However, why should these features emerge in the first place? In the next section, we present interventions to understand emergent behavior.

3.2 SYSTEMATIC INTERVENTIONS

To understand the mechanisms behind emergent features, we systematically intervene on the learning process. We chose these interventions because they either give the agent an advantage or weakness. We summarize these variations in Table 1. By training the models from scratch with different interventions and analyzing the learned representation, we can analyze how environmental features cause different strategies to be learned.

Speed: The first intervention we will make is manipulating the relative speed of the hider and seeker. We explore two variations. In the first variant, we slow down the hider. Likewise, in the second variant, we speed up the hider. We call these agents **slowerhider** and **fasterhider** respectively. The conditions are otherwise the same as the basic model.

Self-Visibility: Awareness of your own visibility is a crucial feature for successfully surviving during hide and seek (Russell et al., 2012). We explicitly incorporate this into the reward function as an auxiliary dense reward signal. Along with the sparse reward above, this agent receives an additional

reward of 0.001 if the seeker does not see it at the current time step. Likewise, the agent receives a punitive reward of -0.001 if it is currently visible by the seeker. We refer to this agent as **visibilityreward**. Furthermore, we also use a variant where the hider has a faster speed, which we call **visibilityreward+faster**.

Environmental Complexity: We also intervene on the complexity of the environment, which lets us analyze the impact of the environment on the emergent features. We use two variations. Firstly, we use a stochastic policy for the seeker agent (instead of deterministic). Specifically, the stochastic seeker randomly visits locations in the map until the hider is within its field of view, at which point it immediately goes towards it. We name this variation **stochasticseeker**. Secondly, we use a stochastic map where we randomly select a number of objects and also randomly position throughout the environment. Consequently, the difficulty of the maps will change between easy (many occlusions) to difficult (few to none occlusions). We name this variation **stochasticmaps**.

Hider Name	Speed	Seeker Policy	Maps	Visibility Reward
basic	2	Deterministic	Deterministic	None
fasterhider	A	Deterministic	Deterministic	None
slowerhider	1	Deterministic	Deterministic	None
stochasticseeker	2	Stochastic	Deterministic	None
stochasticmaps + stochasticseeker	2	Stochastic	Stochastic	None
visibilityreward	2	Deterministic	Deterministic	Yes
visibilityreward + faster	A	Deterministic	Deterministic	Yes

Table 1: **Interventions:** We perturb the learning process in order to understand the causes of different strategies during visual hide and seek. When the speed is "A", the agent has allowed to accelerate to reach higher speeds at a rate of two units per time period squared.

3.3 Implementation Details

We implement our simulation using the Unity game engine along with ML-Agents (Juliani et al., 2018) and PyTorch (Paszke et al., 2017). We plan to publicly release our framework. Given an image as input, we use a four-layer convolutional network as a backbone, which is then fed to a two-layer fully connected layers for the actor and critic models. We use LeakyReLU (Maas et al., 2013) as activation function throughout the network. The input pixel values are normalized to a range between 0 and 1. We train the network using Proximal Policy Optimization, which has been widely successful across reinforcement learning tasks (Schulman et al., 2017). We optimize the objective using Adam optimizer (Kingma & Ba, 2014) for $8 * 10^6$ steps with a learning rate $3.0e-4$ and maximum buffer size of 1,000. We then rolled out the policies for 100 episodes using the same random seed across all the variants. Each episode had 1,000 number of steps at maximum. Following (Mnih et al., 2015), we use a 6-step repeated action, which helps the agent explore the environment. Please see the appendix for full details.

4 Experiments

The goal of our experiments is to analyze the learned capabilities of the polices, quantify the utility of the learned representations, and characterize why strategies emerge in hide and seek games.

Overall, our experiments show that the hiding agent is able to learn to avoid capture. We show that each policy learns a different strategy depending on its environmental abilities. During the process of learning to hide, our results suggest that the agent automatically learns to recognize whether itself is visible or not. However, by comparing performance on the training task versus performance on the visual perception task, we show that the strength of the emergent representation depends on the agent weaknesses. If the agent is too strong, the model does not need to learn useful representations. However, with judicious weakness, the model learns to overcome its disadvantage by learning rich features of the environment. The rest of this section investigates these learning dynamics.

Environment	Average of Living Steps (trained / random)	Success Rate (trained / random)
basic	500 / 41	45.36% / 0
fasterhider	526 / 42	34.02% / 0
visibilityreward + faster	406 / 35	32.99% / 0
stochasticseeker	277 / 45	22.68% / 0
stochasticmaps + stochasticseeker	291 / 48	17.53% / 0
visibilityreward	236 / 43	18.56% / 0
slowerhider	73 / 46	1.03% / 0

Table 2: **Hider Performance:** We show the success rate and average number of living steps of different Hider agents and environments. The maximum number of steps for each episode is 1,000.

4.1 Does the agent learn to solve its training task?

We first directly evaluate the models on their training task, which is to avoid capture from the seeker. We quantify performance using two metrics. Firstly, we use the average number of living steps across all testing episodes. Secondly, we use the success rate, which is the percentage of games that the hider avoids capture throughout the entire episode. Table 2 reports performance, and shows that all agents solve their task better than random chance, suggesting the models successfully learned this play game. As one might expect, the stronger the agent, the better the agent performed: the faster agent frequently outperforms the slower agent.

4.2 What visual concepts are learned in representations?

In this experiment, we want to investigate which visual concepts are encoded in the latent representation. For example, learning to recognize the predator and whether the agent is itself visible to the predator are important prerequisites for hide and seek strategies. Consequently, we design two corresponding tasks to study the encoded visual concepts from the trained policies. The first task analyzes whether the learned features are predictive of the seeker (**Seeker Recognition**). The second analyzes whether if the hider is able to infer the visibility of itself to the seeker (**Awareness of Self-Visibility**). These two tasks can be categorized as two binary classification downstream tasks. To do this, we extract the mid-level features from the learned policy network in different environments, and train a binary logistic regression model on these two proposed tasks.

Environment	Seeker Recognition (/random init policy)	Awareness of Self-Visibility (/random init polcy)
basic	90.83% / 76.83%	75.17% / 64.42%
fasterhider	89.67% / 72.92%	77.92% / 60.17%
visibilityreward + fasterhider	78.08% / 72.23%	63.17% / 60.00%
stochasticseeker	96.42% / 77.83%	94.58% / 64.92%
stochasticmaps + stochasticseeker	95.92% / 79.08%	95.00% / 71.17%
visibilityreward	95.75% / 77.50%	94.92% / 66.17%
slowerhider	83.50% / 78.33%	81.08% / 68.92%

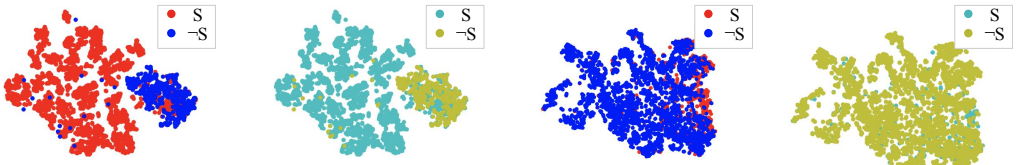
Table 3: **Downstream Visual Perception Tasks:** the table shows the classification accuracies on two important visual perception tasks using mid-level features from all the trained policies. Regardless of the advantages of the hider agent and high performance in the hiding task, the "visibilityreward + faster" policy has the worst performance on both of the two tasks.

The labels are from ground-truth game states. We emphasize that we do not use any of these labels during learning. We only rely on them for studying the representations after the training is completed. Throughout the paper, we use "S" and "¬S" to represent the game state of whether the seeker is visible to the hider, and "H" and "¬H" to represent the game state of whether the hider is visible to the seeker. Therefore, the inputs of these two tasks are the visual observations of the hider agent, and the labels are "H"/"¬H" and "S" /"¬S" respectively. We then use the classifica-

tion accuracy on the test images as a measurement on how well the latent representations encode corresponding visual concepts.

Table 3 shows the results for both tasks. The faster agent, especially "visibilityreward + faster", performs worst on these two perception tasks, even though it has the most advantages among all the variants. Since the faster agents performed better at the survival task (Table 2), this suggests that the performance in the training task does not directly cause the performance in other downstream tasks. To further demonstrate this, we visualize the embedding mid-level features of "visibilityreward" and "visibilityreward+faster" policy with respect to the labels of the above two tasks using t-SNE (Maaten & Hinton, 2008) in Figure 2. The features for "visibilityreward" policy is more separable than the features for "visibilityreward + faster". This qualitative evidence supports that the "visibilityreward + faster" policy is worse at downstream visual perception tasks even though the hider in this case can moves a lot faster.

In the second task, we analyze whether the hider is able to infer its own visibility to the seeker. The features from the stochastic variants and the "visibilityreward" are the most predictive of the hiders own visibility. This could be because the hider agent is encouraged to pay more frequent attention to the seeker agent in order to get higher rewards.



(A) visibilityreward (B) visibilityreward + fasterhider

Figure 2: **t-SNE Embedding** of mid-level features colorized by the labels of the two visual perception tasks from Section 4.2.

4.3 What causes useful features to emerge?

To analyze the relationship between training task difficulty and emerging skills, Figure 3 compares the classification accuracy of using the features to predict self-visibility versus the agent's survival time. Each dot in the plot represents one variant of the environments specified in Table 1. If the agent is not able to avoid capture at all, then the features are clearly poor. However, if the agent is able to completely evade capture, then the features are also poor! Instead, there is a concave relation that suggests agent weaknesses are actually advantageous for learning rich representations. We believe this is the case because by giving the model a disadvantage, the learning process will compensate for the weakness, which it does by learning stronger features. In other words, this suggests that judicious agent weaknesses act as a regularization for learning useful features for hide and seek.

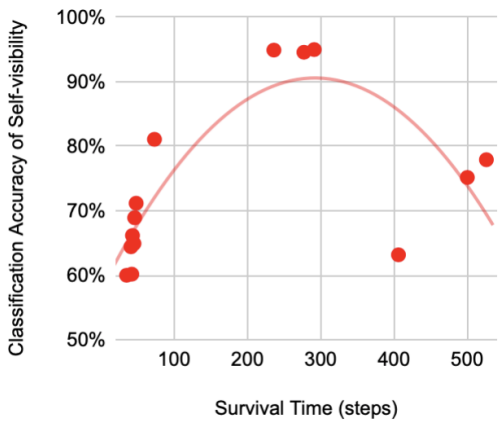


Figure 3: **Quality of Representation vs. Survival Time**: We compare the performance of models on their survival time versus how well the internal representation is predictive of downstream recognition tasks. Each red dot represents one policy trained with different advantages or disadvantages. The curve is the parabolic best fit. Interestingly, improved task performance (survival time) does not always create stronger representations. Instead, the best representations arise from the models with intermediate disadvantages. When the model has a weakness, the model learned to overcome it by instead learning better features.

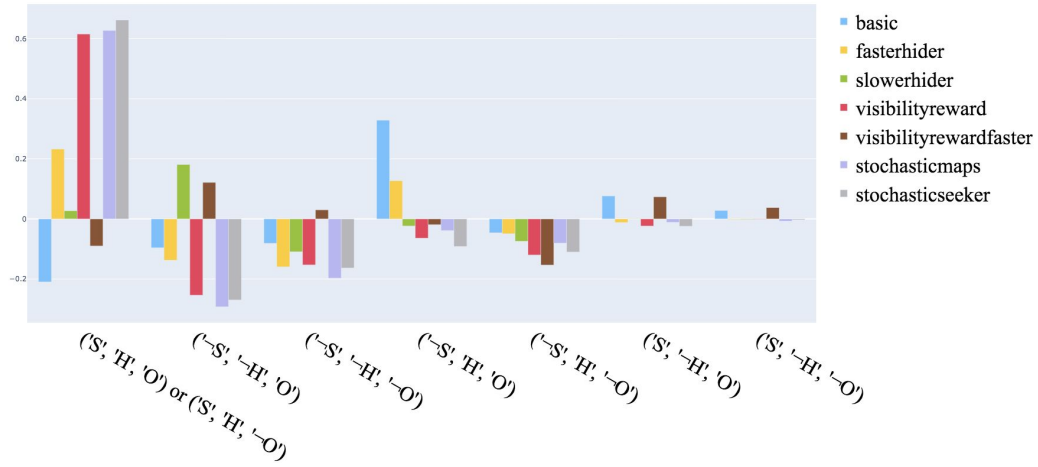


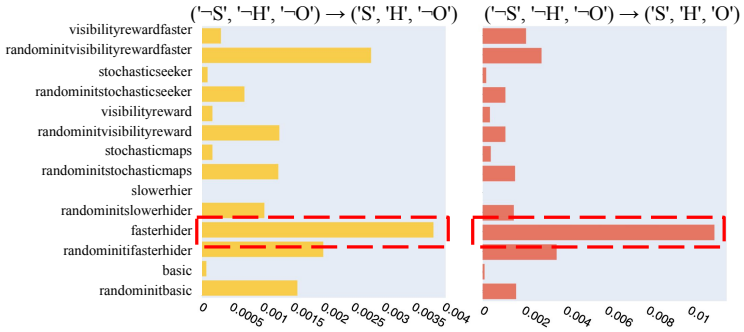
Figure 4: **Frequencies of Visual States.** We show the relative frequency over chance that each visual perception states are visited. "S" stands for whether the Seeker is visible to the Hider, "H" stands for whether the Hider is visible to the Seeker, and "O" means whether there is any obstacle visible to the Hider. "¬" denotes the opposite condition. This plot demonstrates a quantitative way to measure different behaviors using visiting frequencies among important states. For example, we can tell that "basic" agent learns to turn away from the seeker and then run away from it as shown in the high blue bar for state ('¬S', 'H', 'O'). The "basic" hider learns to always run away because its similar speed with the seeker. The "slowerhider" favors to stay at ('¬S', '¬H', 'O') where the hider and the seeker cannot see each other and there is at least one obstacle in front itself. This suggests the slower hider is learning to leverage the environment to compensate for its speed disadvantage. "visibilityrewardfaster" often stays either at the state where they cannot see each other as suggested by the high brown bars of state ('¬S', '¬H', 'O') and ('¬S', '¬H', '¬O'), or stays at the state where the it can see the seeker but the seeker cannot see itself as indicated in the high brown bars at ('S', '¬H', 'O') and ('S', '¬H', '¬O'). This suggests that the "visibilityrewardfaster" can use its speed to hide completely and check the status of the seeker while keeping itself safe.

4.4 How does the trained model hide?

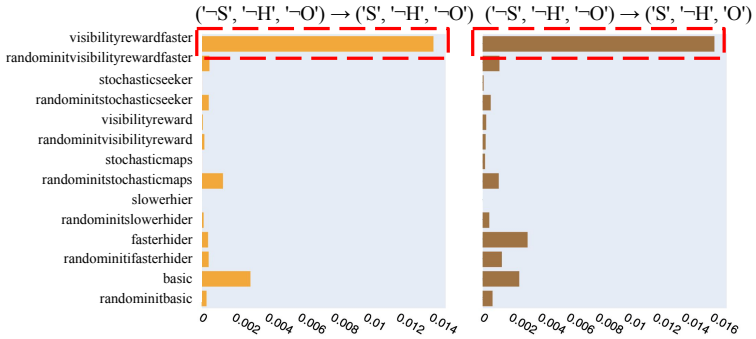
Each agent is trained with varying advantages and disadvantages. We are interested in understanding how these variations affect the behavior of learned policy and how exactly the trained model learns to hide. We quantitatively analyze the *dynamics* of the learned policies by probing the internal game engine state. Note that we only use the game engine states for analysis. During learning, the model is expected to learn to act from pixel input alone (which the experiments above shows happens).

Frequency of Visual States: To analyze states for each policy, we track three visual states in the hide and seek game. 1) **"S" / "¬S"**: whether the hider can see seeker, 2) **"O" / "¬O"**: whether the hider can see any obstacle and 3) **"H" / "¬H"**: whether the hider is visible to the seeker. We rolled out each learned policy for 50,000 steps, and counted how often they enter these game states. Then we plot their relative frequencies by subtracting the frequencies of corresponding random initialized policies from the absolute visiting frequencies.

Figure 4 compares the frequency of states for each model. We observe a few key differences among the agents. When the speeds and abilities of the hider and seeker are the same (basic), the hider learns a policy to first turn away from the seeker then run away, as evidenced by the high blue bar for the state (not S, H, O). However, when the hiding agent is slower than the seeker, the hider frequently enters a state where the two agents cannot see each other, but an obstacle is visible, as evidenced by the high green bar for the state (not S, not H, O). In other words, when the hiding agent has a disadvantage, the model cannot rely on its intrinsic capabilities, and will instead learn to leverage the environment to perform its task. In contrast, when the hiding agent has several advantages such as "visibilityrewardfaster", the policy also learns to make use of them during the training process. As shown in the high brown bars at state ('¬S', '¬H', 'O') and ('¬S', '¬H', '¬O'), as well as ('S', '¬H', 'O') and ('S', '¬H', '¬O'), this hider learns to use its speed and auxiliary visibility reward



(A) The hider and seeker cannot see each other → The hider and the seeker can see each other



(B) The hider and seeker cannot see each other → The hider can see the seeker now while keep itself invisible from the seeker

Figure 5: **Transition Dynamics:** We show the probability of transitions between states across all the variants. All of the four plots share the same y-axis in the table above. Each bar indicates the probability of transiting from one state to the other. The state transitions are labeled on top of each graph, and an explanation of the transition on the bottom of each graph.

to hide completely from the sight of the seeker and monitor the status of the seeker while keeping itself safe at the same time.

Probability of State Transitions: While the state frequency shows the agents' most common states, it does not reflect the *dynamics* of the decision making process. We use the state transitions to further analyze the agent behaviors. For example, a transition from "H" to "¬H" indicates a successful hiding attempt. Following the previous state definitions, there are eight possible combinations out of these three states and $(8 \times 8 - 8) = 56$ transitions in total excluding identity state transitions. We summarize representative cases, and put the full results for all combinations in the Appendix.

Figure 5 quantifies different exploration strategies for each model. For example, the top tow (A) suggests that faster agents tend to monitor the status of the seeker while exposing itself. This likely happens because the faster agents are fast enough that they can immediately react when the seeker becomes too close. Specially, the two red bars in (A) denotes that "fasterhider" policy has the highest probability to transition from the state where the hider and the seek *cannot* see each other to the state where the hider and the seeker *can* see each other. When the status of the seeker becomes clear, the hider in "fasterhider" also exposes itself to the seeker's view.

However, when the faster agent is explicitly penalized for exposing itself ("visibilityreward + fasterhider"), the agent tends to monitor the status of the seeker while keeping itself hidden. This is suggested by the two red bars from Figure 5 (B) where "visibilityreward + fasterhider" policy has the highest probability to transit from the state where the hider and the seeker *cannot* see each other to the state where the hider can see the seeker, yet the seeker *cannot* see the hider. In these cases, the agent has learned to watch the predator, but remain outside of its field of view.

Distance between Agents with respect to Time: We also measure the learned behavior by quantifying distances between the two agents with respect to time. In contrast to previous analysis which focused on short-term dynamics, this lets us quantify long-term dynamics.

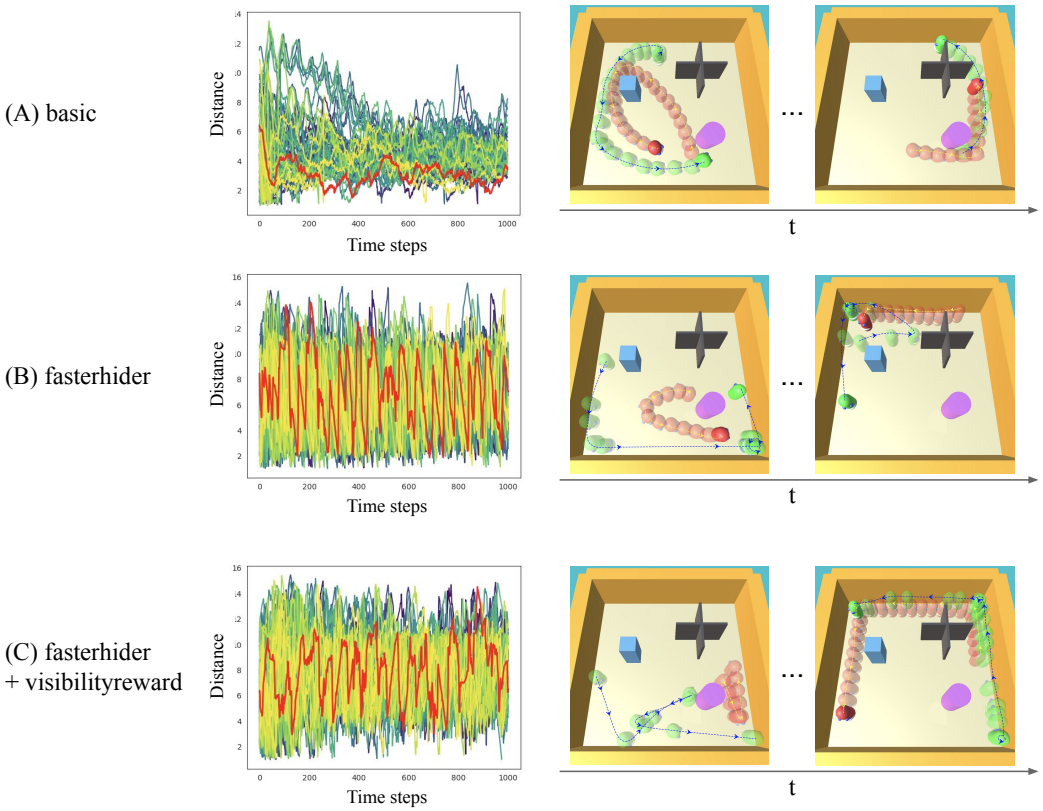


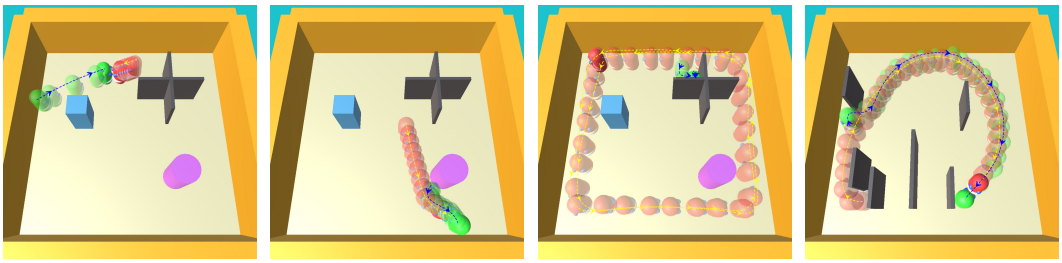
Figure 6: **Distance Versus Time:** We use distance between agents with respect to time steps to quantify long-term dynamics of the game. Hider agents from different policies exhibit diverse pattern in this plot. We further append qualitative demonstrations to explain what dynamics each pattern correspond to on the right side of the plots. The corresponding trajectory is plotted in red line.

Figure 6 plots the distance between agents versus time, alongside representative qualitative cases. When the two agents have the same capabilities (Figure 6a), the agents gradually get closer to each other. However, when one of the hiding agent is faster, the distance between the hider and seeker will significantly oscillate (Figure 6b,c). This is consistent with a reactive strategy where faster agents are able to stand still until they are threatened, at which point they simply run away.

Next to the plots, we also show visual demonstrations that depicts the corresponding strategies learned by different hider agents. One key difference is between the faster hiders with and without a visibility dense reward. The "fasterhider" moves towards the seeker while facing it at the same time, then it moves away when they get too close. On the other hand, the "fasterhider + visibilityreward" agent moves towards the seeker by monitoring the seeker from behind.

5 Discussion

Our experiments suggest there are many diverse strategies for learning to hide from a predator. Moreover, during the learning process, the agents learn a visual representation for their task, such as recognizing their own visibility. However, our experiments show that this emergent representation requires a judicious disadvantage to the agent. If the weakness is too severe, then the learning is derailed. If there is no weakness or even an advantage, then the learning just discovers a reactive



(a) **fasterhider** the Hider moved towards the Seeker quickly, then got caught

(b) **slowerhider** the Hider tended to move backward, then died due to no reaction time

(c) **slowerhider** the Hider found a design flaw in the game, then stay there to live forever

(d) **stochasticmaps** the Hider was caught because it did not notice there was an obstacle behind it

Figure 7: **Representative Failures:** We show a few cases where the hider is unable to avoid capture. Many of these failures are due to lack of memory in the model, suggesting improved memory representations will help this task.

policy without requiring strong representations. However, with moderate disadvantages, the model learns to sufficiently overcome its weakness, which it does by learning strong features.

We believe visual hide and seek, especially from visual scenes, is a promising self-supervised task for learning multi-agent representations. Figure 8 shows a few examples where the hider fails to escape from the predator. Many of these failures are due to the lack of memory in the agent, both for the memory of the map and memory of previous predator locations, which invites further research on computational memory models (Milford et al., 2004; Gupta et al., 2017a; Zhang et al., 2017b; Parisotto & Salakhutdinov, 2017; Khan et al., 2017; Wayne et al., 2018). Overall, these results suggests that improving the agent's ability to learn to hide while simultaneously increasing the severity of the disadvantage will cause increasingly rich strategies to emerge.

REFERENCES

David Abel, Alekh Agarwal, Fernando Diaz, Akshay Krishnamurthy, and Robert E Schapire. Exploratory gradient boosting for reinforcement learning in complex domains. *arXiv preprint arXiv:1603.04119*, 2016.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pp. 892–900, 2016.

Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando de Freitas. Playing hard exploration games by watching youtube. In *Advances in Neural Information Processing Systems*, pp. 2930–2941, 2018.

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula, 2019.

Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*, 2017.

Shehroze Bhatti, Alban Desmaison, Ondrej Miksik, Nantas Nardelli, N Siddharth, and Philip HS Torr. Playing doom with slam-augmented deep reinforcement learning. *arXiv preprint arXiv:1612.00380*, 2016.

Samarth Brahmbhatt and James Hays. Deepnav: Learning to navigate large cities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5193–5202, 2017.

Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. Home: A household multimodal environment. *arXiv preprint arXiv:1711.11017*, 2017.

Lucian Bu, Robert Babu, Bart De Schutter, et al. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2722–2730, 2015.

Shreyansh Daftry, J Andrew Bagnell, and Martial Hebert. Learning transferable policies for monocular reactive mav control. In *International Symposium on Experimental Robotics*, pp. 3–11. Springer, 2016.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2054–2063, 2018.

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1422–1430, 2015.

Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Juan P Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2015.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396. IEEE, 2017.

Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2616–2625, 2017a.

Saurabh Gupta, David Fouhey, Sergey Levine, and Jitendra Malik. Unifying map and landmark based representations for visual navigation. *arXiv preprint arXiv:1712.08125*, 2017b.

Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. Is multiagent deep reinforcement learning the answer or the question? a brief survey. *arXiv preprint arXiv:1810.05587*, 2018.

Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019. ISSN 0036-8075. doi: 10.1126/science.aau6249. URL `https://science.sciencemag.org/content/364/6443/859`.

Unnat Jain, Luca Weihs, Eric Kolve, Mohammad Rastegari, Svetlana Lazebnik, Ali Farhadi, Alexander G Schwing, and Aniruddha Kembhavi. Two body problem: Collaborative visual task completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6689–6699, 2019.

Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018.

Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8. IEEE, 2018.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Arbaaz Khan, Clark Zhang, Nikolay Atanasov, Konstantinos Karydis, Vijay Kumar, and Daniel D Lee. Memory augmented control networks. *arXiv preprint arXiv:1709.05706*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Natural language does not emerge'naturally'in multi-agent dialog. *arXiv preprint arXiv:1706.08502*, 2017.

Guillaume Lample and Devendra Singh Chaplot. Playing fps games with deep reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Siqi Liu, Guy Lever, Josh Merel, Saran Tunyasuvunakool, Nicolas Heess, and Thore Graepel. Emergent coordination through competition. *arXiv preprint arXiv:1902.07151*, 2019.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3, 2013.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Patrick MacAlpine and Peter Stone. Ut austin villa: Robocup 2017 3d simulation league competition and technical challenges champions. In *Robot World Cup*, pp. 473–485. Springer, 2017.

Michael J Milford, Gordon F Wyeth, and David Prasser. Ratslam: a hippocampal model for simultaneous localization and mapping. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 1, pp. 403–408. IEEE, 2004.

Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. In *Advances in Neural Information Processing Systems*, pp. 2419–2430, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisỳ, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

Junhyuk Oh, Valliappa Chockalingam, Satinder Singh, and Honglak Lee. Control of memory, active perception, and action in minecraft. *arXiv preprint arXiv:1605.09128*, 2016.

Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. Ambient sound provides supervision for visual learning. In *European conference on computer vision*, pp. 801–816. Springer, 2016.

Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11(3):387–434, 2005.

Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. *arXiv preprint arXiv:1702.08360*, 2017.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*, 2017.

Iyad Rahwan, Manuel Cebrian, Nick Obradovich, Josh Bongard, Jean-François Bonnefon, Cynthia Breazeal, Jacob W Crandall, Nicholas A Christakis, Iain D Couzin, Matthew O Jackson, et al. Machine behaviour. *Nature*, 568(7753):477, 2019.

James Russell, Brioney Gee, and Christina Bullard. Why do young children hide by closing their eyes? self-visibility and the developing concept of self. *Journal of Cognition and Development*, 13(4):550–576, 2012.

Manolis Savva, Angel X. Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv:1712.03931*, 2017.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. *arXiv preprint arXiv:1904.01201*, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Henry Shevlin and Marta Halina. Apply rich psychological terms in ai with care. *Nature Machine Intelligence*, 1(4):165, 2019.

Yoav Shoham, Rob Powers, and Trond Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171(7):365–377, 2007.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, pp. 2244–2252, 2016.

Karl Tuyls and Gerhard Weiss. Multiagent learning: Basics, challenges, and prospects. *Ai Magazine*, 33(3):41–41, 2012.

Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. Tracking emerges by colorizing videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 391–408, 2018.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack Rae, Piotr Mirowski, Joel Z Leibo, Adam Santoro, et al. Unsupervised predictive memory in a goal-directed agent. *arXiv preprint arXiv:1803.10760*, 2018.

Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Jingwei Zhang, Jost Tobias Springenberg, Joschka Boedecker, and Wolfram Burgard. Deep reinforcement learning with successor features for navigation across similar environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2371–2378. IEEE, 2017a.

Jingwei Zhang, Lei Tai, Joschka Boedecker, Wolfram Burgard, and Ming Liu. Neural slam: Learning to explore with external memory. *arXiv preprint arXiv:1706.09520*, 2017b.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 483–492, 2017a.

Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3357–3364. IEEE, 2017b.

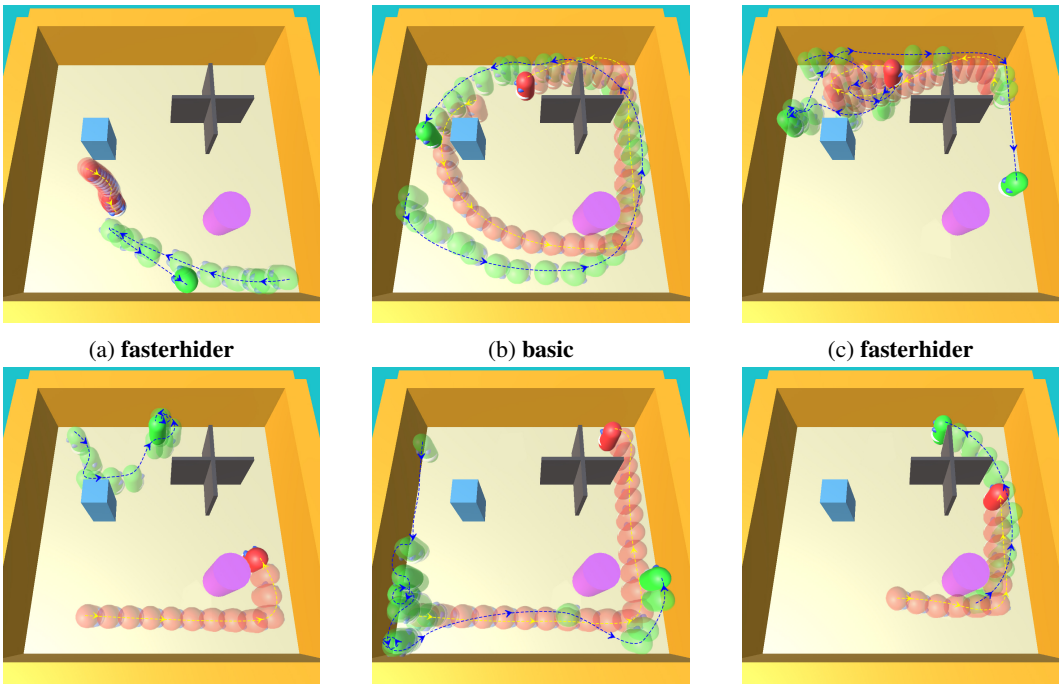
A APPENDIX

A.1 NETWORK ARCHITECTURE

Layer	Kernel Size	Num Outputs	Stride	Padding	Dilation	Activation
Conv1	8	32	4	0	1	LeakyReLU
Conv2	4	64	2	0	1	LeakyReLU
Conv3	3	64	1	0	1	LeakyReLU
FC1	N/A	512	N/A	N/A	N/A	LeakyReLU
Actor-FC2	N/A	512	N/A	N/A	N/A	LeakyReLU
Actor-FC3	N/A	5	N/A	N/A	N/A	LeakyReLU
Critic-FC2	N/A	512	N/A	N/A	N/A	LeakyReLU
Critic-FC3	N/A	1	N/A	N/A	N/A	LeakyReLU

Table 4: **Policy Network Architecture** We list all the parameters used in the policy network.

A.2 MORE VISUALIZATIONS FOR HIDING BEHAVIORS



(a) **fasterhider** (b) **basic** (c) **fasterhider**

(d) **visibilityreward + fasterhider** (e) **visibilityreward + fasterhider** (f) **fasterhider**

Figure 8: **More visual Demonstrations**