Visual Hide and Seek

Boyuan Chen¹, Shuran Song¹, Hod Lipson¹ and Carl Vondrick¹

¹Columbia University, New York, NY 10027
bchen@cs.columbia.edu

Abstract

We train embodied agents to play Visual Hide and Seek to study the relationship between agent behaviors and environmental complexity. In Visual Hide and Seek, a prey must navigate in a simulated environment in order to avoid capture from a predator, only relying on first-person visual observations. By probing different environmental factors, agents exhibit diverse hiding strategies and even the knowledge of its own visibility to other agents in the scene. Furthermore, we quantitatively analyze how agent weaknesses, such as slower speed, affect the learned policy. Our results suggest that, although agent weaknesses make the learning problem more challenging, they also cause more useful features to be learned. Our project website is available at http://www.cs.columbia.edu/bchen/visualhideseek/.

Introduction

With the recent developments in Artificial Intelligence, machines now can perform various tasks with high success rate. Normally, a machine is trained under some provided training environment or dataset, and a cost function is specified for the learning system to minimize to achieve the final goal. As a natural observation, various training environments can induce different biases into the learning system, which could lead to diverse behaviors. However, the relationship between the training environment and the learned behavior is still not well-understood.

In this paper, we introduce the task of Visual Hide and Seek, where a neural network must learn to steer an embodied agent around its environment to avoid capture from a predator. Hide and seek is one of the most important activities played between predators and preys every day in nature. We develop a simulation game setting that mimics the dynamics in typical hide and seek activity. Our main goal is to carefully study the emergent behaviors and representations of multi-agent dynamics through this game.

For a simple version of this game, we train a hider agent to navigate through its environment to maximize its survival time with Reinforcement Learning. Our learning agent only observes the environment through a first-person camera and it needs to map the visual observations directly to actions.

The end-to-end learning system will hence learn a representation of the environment and its dynamics. Figure 1 illustrates the basic setup.

To systematically study the cause of emergent behaviors and agent dynamics from the perspective of environmental complexity, we probe different environmental factors as well as the agents’ abilities such as obstacle layouts, the speed of the agents and awareness of visibility as an auxiliary reward. In addition, we propose three evaluation matrices to quantify the agent dynamics and behaviors, which further help us understand the learned agents.

Our experiments quantitatively suggest that diverse behaviors and dynamics emerge out of different environment setup. Each hider agent chooses its own specific strategy to avoid capture. Moreover, our analysis indicates that although agent weakness, 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 show there is 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 behavioral study of embodied agents. Firstly, we introduce the problem of visual hide-and-seek where an agent receives
a partial observation of its visual environment and learn to navigate to avoid capture. Secondly, we propose three matrices to quantify the learned behaviors and agent dynamics in this game. We further show that diverse behaviors and visual representations of other agents in the scene emerge. Thirdly, we analyze the underlying reasons why these representations emerge, and show they are due to imperfections in the agent’s abilities. The reset 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.

Related Work

Our paper contributes to a rapidly growing area to learn embodied agents that learn to navigate and manipulate environments. 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). 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; Dafry et al., 2016; Mirowski et al., 2016; Brahmbhatt and 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). For a full review of multi-agent reinforcement learning, please see (Panait and Luke, 2005; Bu et al., 2008; Tuyls and Weiss, 2012; Shoham et al., 2007; Hernandez-Leal et al., 2018).

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 and 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. We characterize why representations emerge, which we believe can refine the field’s understanding of self-supervised learning dynamics.

This paper is concurrent to (Baker et al., 2019), and we encourage 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 during both training and testing. 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.

Hide and Seek

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

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 square of 14 unit x 14 unit, and any real value position inside the square without being occupied any obstacle is a valid game state. The speed of the hider is two units per second while the speed of the seeker is one and a half units per second. Our simulation runs in 50 frame per second. The agents can turn 3.3 degree per frame. Each agent has a diameter of one unit. In contrast to other multiplayer games (Sukhbaatar et al., 2016; Moravčík et al., 2017; Lample and Chaplot, 2017; MacAlpine and Stone, 2017; Lowe et al., 2017; Foster 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 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. The hider policy contains two components: a visual feature extractor and a policy network. We represent the visual feature extractor at time $t$ with $\phi(o_t; \theta_f)$ where $o$ is a first-person observation of the agent’s environment. We use a convolutional neural network with skip connections for $\theta_f$. We parametrize the policy $\pi(\phi(o_t; \theta_f); \theta_p)$ with a deep neural network with parameters $\theta_p$. The final action of the hider agent is sampled from the output distribution of $\pi(\phi(o_t; \theta_f); \theta_p)$ where we have $a_t \sim \pi(\phi(o_t; \theta_f); \theta_p)$.

At time step $t$, we assign a small reward ($+0.001$) to the reward function $r_t$ for each living step, and a large negative reward ($-1$) once captured. Our goal is to optimize $\theta_f$ and $\theta_p$ jointly to maximize the accumulated reward as shown in Equation 1. We optimize the model using PPO (Schulman et al., 2017). We call this model basic.

$$\max_{\theta_f, \theta_p} \mathbb{E}_{\pi(\phi(o_t; \theta_f); \theta_p)} \left[ \sum_t r_t \right]$$

(1)

**Starting Positions:** 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 towards 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 recognition of objects in its visual field, such as itself, other agents and obstacles. To analyze this, we fit a linear regression from the learned features to classify different objects and game states. We encode object and game states as discrete variables, and use the classification accuracy as a measure of how well the features are encoding different states. Our experiments will use this approach to probe the learned features.

**Systematic Interventions**

We systematically intervene on the learning process to understand the mechanisms behind emergent features. 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; Reinhold et al., 2019). 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.

<table>
<thead>
<tr>
<th>Hider Name</th>
<th>Speed</th>
<th>Seeker Policy</th>
<th>Maps</th>
<th>Visibility Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>2</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>None</td>
</tr>
<tr>
<td>fasterhider</td>
<td>A</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>None</td>
</tr>
<tr>
<td>slowerhider</td>
<td>1</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>None</td>
</tr>
<tr>
<td>stochasticseeker</td>
<td>1</td>
<td>Deterministic</td>
<td>Stochastic</td>
<td>None</td>
</tr>
<tr>
<td>stochasticmaps + stochasticseeker</td>
<td>1</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Yes</td>
</tr>
<tr>
<td>visibilityreward</td>
<td>2</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>Yes</td>
</tr>
<tr>
<td>visibilityreward + faster</td>
<td>A</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 1: Model Variations: 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.

**Implementation Details**

We implement our simulation using the Unity game engine along with ML-Agents (Juliani et al., 2018) and PyTorch (Paszke et al., 2019). 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 successful across reinforcement learning tasks (Schulman et al., 2017). We optimize the objective using Adam optimizer (Kingma and Ba, 2014) for $8 \times 10^6$ steps with a learn-
ing 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 (Minh et al., 2015), we use a 6-step repeated action, which helps the agent explore the environment.

Experiments

The goal of our experiments is to analyze the learned capabilities of the policies, exhibit what behaviors are emerged and characterize why different strategies emerge in the 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 some agents automatically learn to recognize whether itself is visible or not. We also found that with judicious weakness, the model learns to overcome its disadvantage by learning rich features of the environment. The rest of this section investigate these learning dynamics.

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 the 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 to play this game. As one might expect, the stronger the agent or the easier the environment, the better the agent performed: the faster agents frequently outperforms the slower ones, and the agents trained and tested in fixed environments consistently achieve higher scores than the ones with stochastic opponents or random obstacle layouts.

Another thing to note is the effect of adding the auxiliary reward of visibility. By comparing the performance of “basic” and “visibilityreward”, the scores are decreased after adding the auxiliary reward. This is because the optimization of the hider policy becomes more difficult. While moving at the same speed with the seeker, it is hard for the hider to turn to be invisible from being visible to the seeker to receive positive visibility reward. According to the seeker policy, the seeker will keep following the hider once the hider is visible. Hence, the hider agents receives more negative rewards due to this limit. Once the hider is free from the speed limit as shown in “visibilityreward + faster”, the hider reaches comparable scores with the “basic” policy.

However, this does not mean adding visibility reward is an undesired choice. Rather as we will show in the next section, making the problem harder sometimes leads to stronger capabilities on related downstream tasks, which further makes the problem interesting.

What visual concepts are learned?

Learning to recognize the predator and whether the agent itself is visible to the predator are important prerequisites for hiding strategies. In this experiment, we want to investigate which visual concepts are encoded in the latent representations from policies trained under different variants. Consequently, we design two corresponding tasks to study the encoded visual concepts from the trained hider policies.

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. Mean and standard deviation are presented.

<table>
<thead>
<tr>
<th>Environment</th>
<th>Average of Living Steps (trained / random)</th>
<th>Success Rate (trained / random)</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>$513 \pm 10 / 43 \pm 2$</td>
<td>$46.02% \pm 2.60% / 0$</td>
</tr>
<tr>
<td>fasterhider</td>
<td>$522 \pm 3 / 43 \pm 5$</td>
<td>$33.27% \pm 2.00% / 0$</td>
</tr>
<tr>
<td>visibilityreward + faster</td>
<td>$422 \pm 24 / 41 \pm 4$</td>
<td>$33.29% \pm 1.33% / 0$</td>
</tr>
<tr>
<td>stochasticseeker</td>
<td>$271 \pm 5 / 43 \pm 2$</td>
<td>$23.75% \pm 1.52% / 0$</td>
</tr>
<tr>
<td>stochasticmaps + stochasticseeker</td>
<td>$281 \pm 12 / 55 \pm 7$</td>
<td>$17.45% \pm 1.59% / 0$</td>
</tr>
<tr>
<td>slowerhider</td>
<td>$248 \pm 13 / 46 \pm 2$</td>
<td>$19.93% \pm 3.55% / 0$</td>
</tr>
<tr>
<td>seeker</td>
<td>$72 \pm 5 / 44 \pm 3$</td>
<td>$1.08% \pm 0.07% / 0$</td>
</tr>
</tbody>
</table>

Figure 2: Game State Descriptions: we explore three important binary game states for quantifying behaviors of the trained agents.

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.

The extracted feature is the activation from the last convolutional layer in the policy network with a dimension of 512. 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
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.

<table>
<thead>
<tr>
<th>Environment</th>
<th>Seeker Recognition ( / random init policy)</th>
<th>Awareness of Self-Visibility ( / random init policy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>91.03 ± 0.21 / 79.25 ± 2.82</td>
<td>75.84 ± 0.66 / 63.92 ± 0.50</td>
</tr>
<tr>
<td>fasterhider</td>
<td>90.01 ± 0.34 / 74.84 ± 1.91</td>
<td>77.88 ± 0.05 / 60.30 ± 0.13</td>
</tr>
<tr>
<td>visibilityreward + fasterhider</td>
<td>79.19 ± 1.11 / 74.18 ± 1.95</td>
<td>65.05 ± 1.88 / 60.84 ± 0.84</td>
</tr>
<tr>
<td>stochasticseeker</td>
<td>96.17 ± 0.29 / 77.54 ± 1.29</td>
<td>94.00 ± 0.56 / 65.05 ± 0.13</td>
</tr>
<tr>
<td>stochasticmaps + stochasticseeker</td>
<td>95.55 ± 0.38 / 80.75 ± 1.67</td>
<td>94.25 ± 0.75 / 70.30 ± 0.38</td>
</tr>
<tr>
<td>visibilityreward</td>
<td>96.21 ± 0.46 / 77.65 ± 0.15</td>
<td>95.30 ± 0.38 / 63.88 ± 2.30</td>
</tr>
<tr>
<td>slowerhider</td>
<td>83.96 ± 0.46 / 77.79 ± 0.54</td>
<td>81.71 ± 0.63 / 68.59 ± 0.34</td>
</tr>
</tbody>
</table>

Figure 3: t-SNE Embedding of mid-level features colorized by the labels of the two visual perception tasks from Table 3

Figure 4: 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 variant of a trained policy or a random policy under different environmental conditions. 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.

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 studying different properties of internal game states. We only use the game states for analysis, not learning.

Visitation Frequency of Visual States: To analyze states for each policy, we track three visual states (Figure 2) in the hide and seek game. 1) “S” / “¬S”: whether the hider can see the seeker, 2) “O” / “¬O”: whether the hider can see any
Figure 5: Frequencies of Visual States. We show the relative frequency over chance that each visual perception states are visited by the hider agent. “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. Figure 2 shows the visual illustrations. We can tell that “basic” agent learns to run from the seeker (¬S) with its back facing the seeker (H') as indicated by the high blue bar at (¬S, H', O). When the hider moves much slower (“slowerhider”), the agent 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 that the slower hider learns to leverage the environment to compensate for its speed limitation. “visibilityrewardfaster” learns to hide by completely evading from the sight of the seeker (¬H) as shown by the high brown bars at (¬S', ¬H', O'), (¬S', ¬¬H', ¬O'), (S', ¬H', O') and (S', ¬¬H', ¬O').

obstacle and 3) “H’ / “¬H”: whether the hider is visible to the seeker. We rolled out each learned policy for 50,000 steps, and computed how often they visit each game state. We plot their relative frequencies by subtracting the frequencies of corresponding random initialized policies from the absolute visiting frequencies.

Figure 5 compares the frequency of states for each model. We observe a few key differences among the agents. When the speed and abilities of the hider and seeker are the same (basic), the agent hides by keeping running from the seeker (H) and facing the seeker with its back ¬S, as evidenced by the high blue bar at (¬S, H, O). However, when the hider is much slower than the seeker, the hider frequently enters a state where the two agents cannot see each other (¬S, ¬H), but an obstacle is in front of it (O), as evidenced by the high green bar at (¬S, ¬¬H, O). In other words, when the hider has a disadvantage, the mode cannot rely on its intrinsic capabilities, and will instead learn to leverage the environment (e.g., obstacle).

In contrast, when the hider has several advantages such as “visibilityrewardfaster”, the policy also learns to make use of them during training. 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 high speed and auxiliary visibility reward to hide completely from the sight of the seeker.

Probability of State Transitions:

Figure 6: Transition Dynamics: We show the probability of transitions between states across all the variants. Each bar indicates the probability of transiting from one state to the other. Each variant shows different transitional probabilities representing different behaviors. “randominit” prefix represents the random initialized policy for each variant.
Figure 7: **Distance Versus Time:** We use the standard deviation of distances between agents through 100 random rollouts versus time to quantify long-term dynamics of the game. Different policies exhibit different stability of the distances over time reflected by the standard deviation plot. We append qualitative demonstrations to explain what dynamics each policy correspond to on the right side of the plots.

While the state visitation 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 will release the full results for all combinations along with our code.

From the plots in Figure 6, we can tell a key difference of exploration behavior between “fasterhider” and “visibilityreward + faster”. The two red bounding boxes in the top row (A) suggests that “fasterhider” policy has the highest probability to transition from the state where the hider and the seeker cannot see each other to the state where the hider and the seeker can see each other. When the status of the seeker is available, the faster hider also exposes itself to the seeker’s view. This observation shows that the faster only agent tends to explore the status of the seeker by exposing itself. This likely happens because the faster agent is fast enough to react immediately when the seeker gets too close.

When the faster agent is explicitly penalized for exposing itself (“visibilityreward + fasterhider”), the agent tends to explore the status of the seeker while keeping itself hidden. This observation is supported by the two red bars from the bottom row (B). “visibilityreward + faster” 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 7 plots the standard deviation of distances between the two agents versus time throughout 100 random rollouts, alongside representative qualitative cases. When the two agents have the same capabilities (Figure 7a), 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 7b,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 as indicated in the visual illustration.

Next to the plots, we also show visual demonstrations that depicts the correlated strategies learned by different hider agents. Another observation is the different behavior 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 watching the seeker from behind. Although the standard deviation plots exhibit similar trend, these two learned polices are still different by looking at the visual demonstrations. This further supports our claim that we need to quantify the agent behaviors with matrices from multiple angles instead of relying on individual case.

**Discussion and Future Work**

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 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 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 and 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. With more complicated environment set up such as more complex room layout, diverse object categories and allowance for agents and objects interaction, we believe more visual representations and different level of game dynamics can be studied in future research.

Furthermore, we hope that our work can also encourage future works on specialized algorithms for visual egocentric multi-agent learning. Group of agents with different goals and capabilities can cause the environment become highly non-stationary. Better policy learning algorithms and feature extraction models are likely to improve the learned strategies and representations.

Finally, while our work offers an approach to study the visual hide and seek game under various changes in the environment and the emergent behaviors, the hide and seek activity in nature is more rich and complex. For example, environment changes over time, and the policies or behaviors of other agents also evolve. In the future, we hope to study how agents’ behaviors will evolve when the environment or other agents become different in a sudden or gradual manner. In the meantime, it is important to consider building artificial agents that can learn and adapt under those changes. For instance, this could be done by training the agents in procedurally generated environments (Shaker et al., 2016; Freiknecht and Effelsberg, 2017) or framing the problem into an open-ended evolutionary process (Stanley, 2019; Sayama, 2018). We hope that our work can step closer to study these future research topics and can open up interesting ideas and directions.

**Acknowledgements**

This research is supported by DARPA MTO grant HR0011-18-2-0020 and NSF NRI 1925157. We would also like to thank NVIDIA for the donation of GPU.

**References**


![Representative Failures](image-url)


