EMERGENT TOOL USE FROM MULTI-AGENT AUTOCURRICULA

Bowen Baker[*]
OpenAI
bowen@openai.com

Ingmar Kanitscheider[*]
OpenAI
ingmar@openai.com

Todor Markov[*]
OpenAI
todor@openai.com

Yi Wu[*]
OpenAI
jxwuyi@openai.com

Glenn Powell[*]
OpenAI
glenn@openai.com

Bob McGrew[*]
OpenAI
bmcgrew@openai.com

Igor Mordatch[*][†]
Google Brain
imordatch@google.com

ABSTRACT

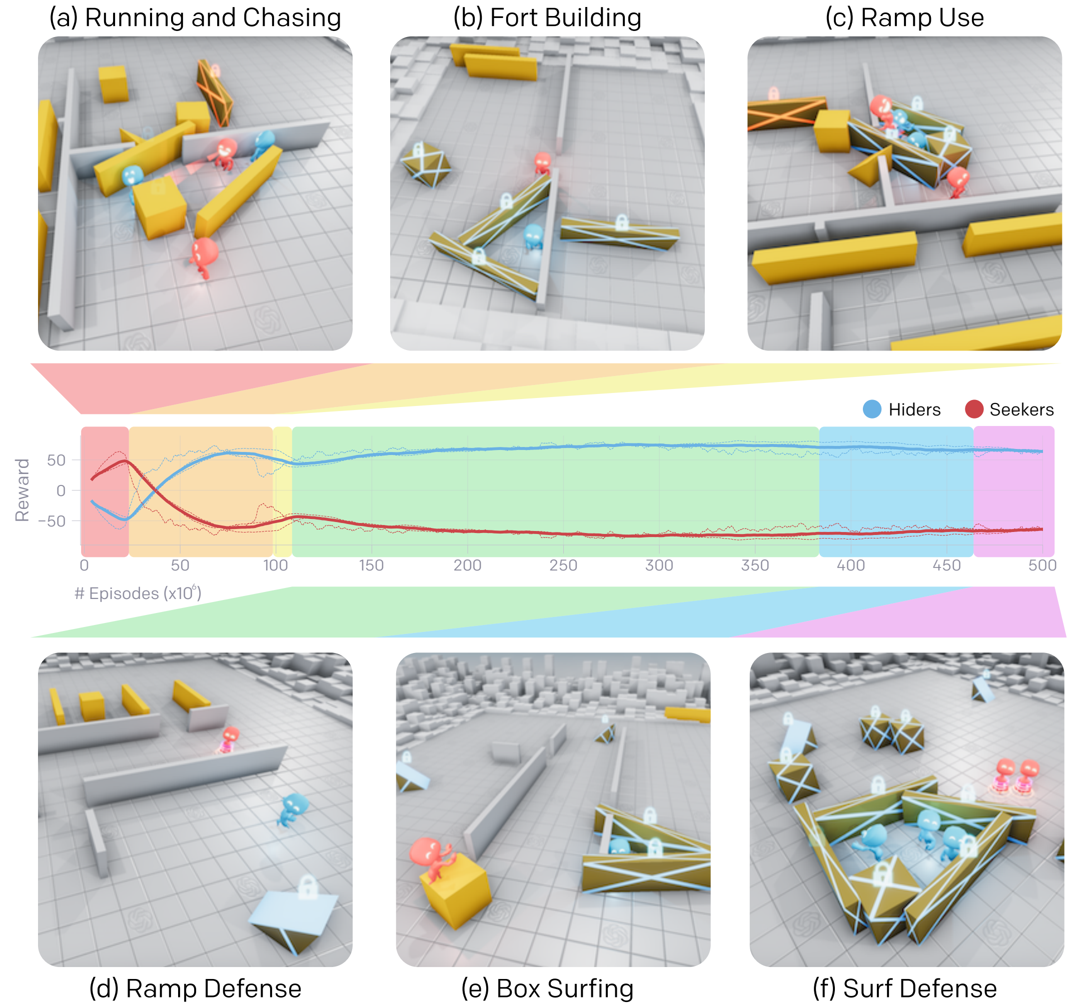
Through multi-agent competition, the simple objective of *hide-and-seek*, and standard reinforcement learning algorithms at scale, we find that agents create a self-supervised autocurriculum inducing multiple distinct rounds of emergent strategy, many of which require sophisticated tool use and coordination. We find clear evidence of six emergent phases in agent strategy in our environment, each of which creates a new pressure for the opposing team to adapt; for instance, agents learn to build multi-object shelters using moveable boxes which in turn leads to agents discovering that they can overcome obstacles using ramps. We further provide evidence that multi-agent competition may scale better with increasing environment complexity and leads to behavior that centers around far more human-relevant skills than other self-supervised reinforcement learning methods such as intrinsic motivation. Finally, we propose transfer and fine-tuning as a way to quantitatively evaluate targeted capabilities, and we compare hide-and-seek agents to both intrinsic motivation and random initialization baselines in a suite of domain-specific intelligence tests.

1 INTRODUCTION

Creating intelligent artificial agents that can solve a wide variety of complex human-relevant tasks has been a long-standing challenge in the artificial intelligence community. Of particular relevance to humans will be agents that can sense and interact with objects in a physical world. One approach to creating these agents is to explicitly specify desired tasks and train a reinforcement learning (RL) agent to solve them. On this front, there has been much recent progress in solving physically grounded tasks, e.g. dexterous in-hand manipulation (Rajeswaran et al., 2017; Andrychowicz et al., 2018) or locomotion of complex bodies (Schulman et al., 2015; Heess et al., 2017). However, specifying reward functions or collecting demonstrations in order to supervise these tasks can be

[*] This was a large project and many people made significant contributions. Bowen, Bob, and Igor conceived the project and provided guidance through all stages of the work. Bowen created the initial environment, infrastructure and models, and obtained the first results of sequential skill progression. Ingmar obtained the first results of tool use, contributed to environment variants, created domain-specific statistics, and with Bowen created the final environment. Todor created the manipulation tasks in the transfer suite, helped Yi with the RND baseline, and prepared code for open-sourcing. Yi created the navigation tasks in the transfer suite, intrinsic motivation comparisons, and contributed to environment variants. Glenn contributed to designing the final environment and created final renderings and project video. Igor provided research supervision and team leadership.

[†] Work performed while at OpenAI


(a) Running and Chasing (b) Fort Building (c) Ramp Use

(d) Ramp Defense (e) Box Surfing (f) Surf Defense

Figure 1: Emergent Skill Progression From Multi-Agent Autocurricula. Through the reward signal of hide-and-seek (shown on the y-axis), agents go through 6 distinct stages of emergence. (a) Seekers (red) learn to chase hiders, and hiders learn to crudely run away. (b) Hiders (blue) learn basic tool use, using boxes and sometimes existing walls to construct forts. (c) Seekers learn to use ramps to jump into the hiders' shelter. (d) Hiders quickly learn to move ramps to the edge of the play area, far from where they will build their fort, and lock them in place. (e) Seekers learn that they can jump from locked ramps to unlocked boxes and then *surf* the box to the hiders' shelter, which is possible because the environment allows agents to move together with the box regardless of whether they are on the ground or not. (f) Hiders learn to lock all the unused boxes before constructing their fort. We plot the mean over 3 independent training runs with each individual seed shown with a dotted line. Please see openai.com/blog/emergent-tool-use for example videos.

time consuming and costly. Furthermore, the learned skills in these single-agent RL settings are inherently bounded by the task description; once the agent has learned to solve the task, there is little room to improve.

Due to the high likelihood that direct supervision will not scale to unboundedly complex tasks, many have worked on unsupervised exploration and skill acquisition methods such as intrinsic motivation. However, current undirected exploration methods scale poorly with environment complexity and are drastically different from the way organisms evolve on Earth. The vast amount of complexity and diversity on Earth evolved due to co-evolution and competition between organisms, directed by natural selection (Dawkins & Krebs, 1979). When a new successful strategy or mutation emerges, it changes the implicit task distribution neighboring agents need to solve and creates a new pressure for adaptation. These evolutionary arms races create implicit *autocurricula* (Leibo et al., 2019a) whereby competing agents continually create new tasks for each other. There has been much success in leveraging multi-agent autocurricula to solve multi-player games, both in classic discrete games such as Backgammon (Tesauro, 1995) and Go (Silver et al., 2017), as well as in continuous real-time domains such as Dota (OpenAI, 2018) and Starcraft (Vinyals et al., 2019). Despite the impressive emergent complexity in these environments, the learned behavior is quite abstract and disembodied from the physical world. Our work sees itself in the tradition of previous studies that showcase emergent complexity in simple physically grounded environments (Sims, 1994a; Bansal et al., 2018; Jaderberg et al., 2019; Liu et al., 2019); the success in these settings inspires confidence that inducing autocurricula in physically grounded and open-ended environments could eventually enable agents to acquire an unbounded number of human-relevant skills.

We introduce a new mixed competitive and cooperative physics-based environment in which agents compete in a simple game of hide-and-seek. Through only a visibility-based reward function and competition, agents learn many emergent skills and strategies including collaborative tool use, where agents intentionally change their environment to suit their needs. For example, hiders learn to create shelter from the seekers by barricading doors or constructing multi-object forts, and as a counter strategy seekers learn to use ramps to jump into hiders' shelter. Moreover, we observe signs of dynamic and growing complexity resulting from multi-agent competition and standard reinforcement learning algorithms; we find that agents go through as many as six distinct adaptations of strategy and counter-strategy, which are depicted in Figure 1. We further present evidence that multi-agent co-adaptation may scale better with environment complexity and qualitatively centers around more human-interpretable behavior than intrinsically motivated agents.

However, as environments increase in scale and multi-agent autocurricula become more open-ended, evaluating progress by qualitative observation will become intractable. We therefore propose a suite of targeted intelligence tests to measure capabilities in our environment that we believe our agents may eventually learn, e.g. object permanence (Baillargeon & Carey, 2012), navigation, and construction. We find that for a number of the tests, agents pretrained in hide-and-seek learn faster or achieve higher final performance than agents trained from scratch or pretrained with intrinsic motivation; however, we find that the performance differences are not drastic, indicating that much of the skill and feature representations learned in hide-and-seek are entangled and hard to fine-tune.

The main contributions of this work are: 1) clear evidence that multi-agent self-play can lead to emergent autocurricula with many distinct and compounding phase shifts in agent strategy, 2) evidence that when induced in a physically grounded environment, multi-agent autocurricula can lead to human-relevant skills such as tool use, 3) a proposal to use transfer as a framework for evaluating agents in open-ended environments as well as a suite of targeted intelligence tests for our domain, and 4) open-sourced environments and code[1] for environment construction to encourage further research in physically grounded multi-agent autocurricula.

2 Related Work

There is a long history of using self-play in multi-agent settings. Early work explored self-play using genetic algorithms (Paredis, 1995; Pollack et al., 1997; Rosin & Belew, 1995; Stanley & Miikkulainen, 2004). Sims (1994a) and Sims (1994b) studied the emergent complexity in morphology and behavior of creatures that coevolved in a simulated 3D world. Open-ended evolution was further explored in the environments Polyworld (Yaeger, 1994) and Geb (Channon et al., 1998), where agents compete and mate in a 2D world, and in Tierra (Ray, 1992) and Avida (Ofria & Wilke, 2004), where computer programs compete for computational resources. More recent work attempted to formulate necessary preconditions for open-ended evolution (Taylor, 2015; Soros & Stanley, 2014). Co-adaptation between agents and environments can also give rise to emergent complexity (Florensa et al., 2017; Sukhbaatar et al., 2018; Wang et al., 2019). In the context of multi-agent RL, Tesauro (1995), Silver et al. (2016), OpenAI (2018), Jaderberg et al. (2019) and Vinyals et al. (2019) used self-play with deep RL techniques to achieve super-human performance in Backgammon, Go, Dota, Capture-the-Flag and Starcraft, respectively. Bansal et al. (2018) trained agents in a simulated 3D physics environment to compete in various games such as sumo wrestling and soccer goal shooting. In Liu et al. (2019), agents learn to manipulate a soccer ball in a 3D soccer environment and discover

[1] Code can be found at `github.com/openai/multi-agent-emergence-environments`.

emergent behaviors such as ball passing and interception. In addition, communication has also been shown to emerge from multi-agent RL (Sukhbaatar et al., 2016; Foerster et al., 2016; Lowe et al., 2017; Mordatch & Abbeel, 2018).

Intrinsic motivation methods have been widely studied in the literature (Chentanez et al., 2005; Singh et al., 2010). One example is count-based exploration, where agents are incentivized to reach infrequently visited states by maintaining state visitation counts (Strehl & Littman, 2008; Bellemare et al., 2016; Tang et al., 2017) or density estimators (Ostrovski et al., 2017; Burda et al., 2019b). Another paradigm are transition-based methods, in which agents are rewarded for high prediction error in a learned forward or inverse dynamics model (Schmidhuber, 1991; Stadie et al., 2015; Mohamed & Rezende, 2015; Houthooft et al., 2016; Achiam & Sastry, 2017; Pathak et al., 2017; Burda et al., 2019a; Haber et al., 2018). Jaques et al. (2019) consider multi-agent scenarios and adopt causal influence as a motivation for coordination. In our work, we utilize intrinsic motivation methods as an alternative exploration baseline to multi-agent autocurricula. Similar comparisons have also been made in Haber et al. (2018) and Leibo et al. (2019b).

Tool use is a hallmark of human and animal intelligence (Hunt, 1996; Shumaker et al., 2011); however, learning tool use in RL settings can be a hard exploration problem when rewards are unaligned. For example, in Forestier et al. (2017); Xie et al. (2019) a real-world robot learns to solve various tasks requiring tools. In Bapst et al. (2019), an agent solves construction tasks in a 2-D environment using both model-based and model-free methods. Allen et al. (2019) uses a combination of human-designed priors and model-based policy optimization to solve a collection of physics-based puzzles requiring tool use. However, in each of these works, agents were explicitly incentivized to interact with and use tools, whereas in our environment agents implicitly create this incentive through multi-agent competition.

3 Hide and Seek

Agents are tasked with competing in a two-team hide-and-seek game in a physics-based environment. The *hiders* are tasked with avoiding line of sight from the *seekers*, and the seekers are tasked with keeping vision of the hiders. There are objects scattered throughout the environment that the agents can grab and also lock in place. There are also randomly generated immovable rooms and walls that the agents must learn to navigate. Before the game of hide-and-seek begins, the hiders are given a *preparation phase* where the seekers are immobilized, giving the hiders a chance to run away or change their environment.

There are no explicit incentives for agents to interact with objects in the environment; the only supervision given is through the hide-and-seek objective. Agents are given a *team based* reward; hiders are given a reward of 1 if all hiders are hidden and -1 if any hider is seen by a seeker. Seekers are given the opposite reward, -1 if all hiders are hidden and +1 otherwise. To confine agent behavior to a reasonable space, agents are penalized with a reward of -10 if they go too far outside of the play area (outside an 18 meter square). An episode lasts 240 timesteps, the first 40% of which are the preparation phase where all agents are given zero reward.

We simulate the environment in the MUJOCO physics engine (Todorov et al., 2012). The world is populated with 1 to 3 hiders, 1 to 3 seekers, 3 to 9 movable boxes of which at least 3 are elongated, 2 movable ramps, and randomly generated static walls and rooms. We also experiment with a simpler, less randomized environment described in Appendix A.3. Agents observe the position, velocity, and size (in the case of the randomly shaped boxes) of objects and other agents. If entities are not in line-of-sight of the agent or not in a 135 degree cone in front of the agent, then they are masked out in the policy. Agents also have 30 range sensors arrayed evenly around them, similar to a lidar. In addition, each agent observes its own team and other agents' teams as well as how much time is left in the preparation phase.

Agents are simulated as spherical objects and have 3 action types that can be chosen simultaneously at each time step. They may *move* by setting a discretized force along their x and y axis and torque around their z-axis. They have a single binary action to *grab* objects, which binds the agent to the closest object while the action is enabled. Agents may also *lock* objects in place with a single binary action. Objects may be unlocked only by agents on the team of the agent who originally locked the object. Agents may only grab or lock objects that are in front of them and within a small radius.

4 Policy Optimization

Agents are trained using self-play, which acts as a natural curriculum as agents always play opponents of an appropriate level.

Agent policies are composed of two separate networks with different parameters – a policy network which produces an action distribution and a critic network which predicts the discounted future returns. Policies are optimized using Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Generalized Advantage Estimation (GAE) (Schulman et al., 2015), and training is performed using *rapid* (OpenAI, 2018), a large-scale distributed RL framework. We utilize decentralized execution and centralized training. At execution time, each agent acts given only its own observations and memory state. At optimization time, we use a centralized omniscient value function for each agent, which has access to the full environment state without any information masked due to visibility, similar to Pinto et al. (2017); Lowe et al. (2017); Foerster et al. (2018).

In all reported experiments, agents share the same policy parameters but act and observe independently; however, we found using separate policy parameters per agent also achieved all six stages of emergence but at reduced sample efficiency.

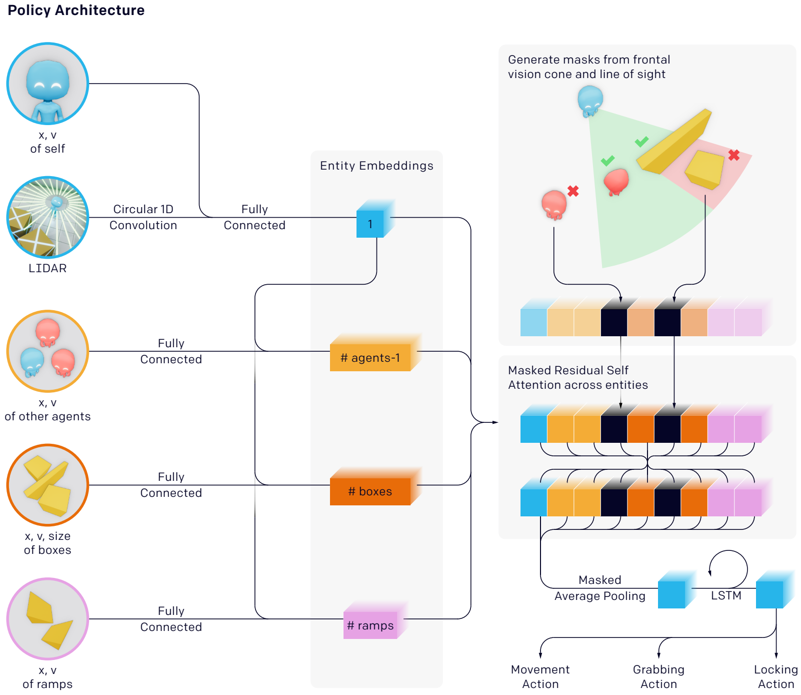


Figure 2: Agent Policy Architecture. All entities are embedded with fully connected layers with shared weights across entity types, e.g. all box entities are encoded with the same function. The policy is ego-centric so there is only one embedding of "self" and $(\#\text{agents} - 1)$ embeddings of other agents. Embeddings are then concatenated and processed with masked residual self-attention and pooled into a fixed sized vector (all of which admits a variable number of entities). x and v stand for state (position and orientation) and velocity.

We utilize entity-centric observations (Džeroski et al., 2001; Diuk et al., 2008) and use attention mechanisms to capture object-level information (Duan et al., 2017; Zambaldi et al., 2018). As shown in Figure 2 we use a self-attention (Vaswani et al., 2017) based policy architecture over entities, which is permutation invariant and generalizes to varying number of entities. More details can be found in Appendix B.

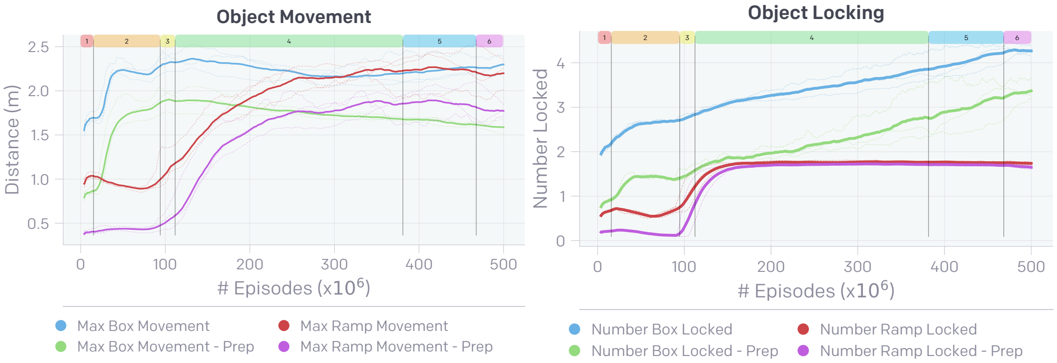


Figure 3: Environment specific statistics used to track stages of emergence in hide-and-seek. We plot the mean across 3 seeds with each individual seed shown in a dotted line, and we overlay the 6 emergent phases of strategy: (1) Running and Chasing, (2) Fort Building, (3) Ramp Use, (4) Ramp Defense, (5) Box Surfing, (6) Surf Defense. We track the maximum movement of any box or ramp during the game as well as during the preparation phase (denoted with "Prep"). We similarly track how many objects of each type were locked at the end of the episode and preparation phase. As agents train, their interaction with the tools in their environment changes. For instance, as the agents learn to build forts they move boxes and lock boxes much more during the preparation phase.

5 Auto-Curricula and Emergent Behavior

As agents train against each other in hide-and-seek, as many as six distinct strategies emerge, each of which creates a previously non-existing pressure for agents to progress to the next stage. Note that there are no direct incentives for agents to interact with objects or to explore, but rather the emergent strategies are solely a result of the autocurriculum induced by multi-agent competition. Figure 1 shows the progression of emergent strategies agents learn in our environment (see Appendix A.1 for trajectory traces of each strategy[2]).

Initially, hiders and seekers learn to crudely run away and chase. After approximately 25 million episodes of hide-and-seek, the hiders learn to use the tools at their disposal and intentionally modify their environment. They begin to construct secure shelters in which to hide by moving many boxes together or against walls and locking them in place. After another 75 million episodes, the seekers also learn rudimentary tool use; they learn to move and use ramps to jump over obstacles, allowing them to enter the hiders' shelter. 10 million episodes later, the hiders learn to defend against this strategy; the hiders learn to bring the ramps to the edge of the play area and lock them in place, seemingly removing the only tool the seekers have at their disposal.

We originally believed defending against ramp use would be the last stage of emergence in this environment; however, we were surprised to find that yet two more qualitatively new strategies emerged. After 380 million total episodes of training, the seekers learn to bring a box to the edge of the play area where the hiders have locked the ramps. The seekers then use the ramp to move on top of the box and *surf* it to the hiders' shelter. This is possible because the agents' movement action allows them to apply a force on themselves regardless of whether they are on the ground or not; if they do this while grabbing the box under them, the box will move with while they are on top of it. In response, the hiders learn to lock all of the boxes in place before building their shelter.[3]

In all stages of strategy agents must learn to coordinate within their team. Similar to Liu et al. (2019), we use team-based rewards such that agents are required to collaborate in order to succeed; however, in our work we require neither population-based training (Jaderberg et al., 2017) or evolved dense rewards (Jaderberg et al., 2019). Notably, hiders learn efficient division of labor; for instance, when constructing shelter they often separately bring their own box to the construction area. Furthermore,

[2] See openai.com/blog/emergent-tool-use for sample videos.

[3] Note that the discovery of a new skill does not necessarily correlate to the reward of a team changing. For example, the hider reward still decreases even after the discovery of surf defense, which is likely because teams with one or two hiders often do not have enough time to lock all of the boxes in play.

as hiders attempt to defend against box surfing by locking boxes in place during the preparation phase, we find that 2 and 3 hider teams are able to lock 25% and 36% more boxes, respectively, than a single hider team, indicating that larger teams are able to divide labor and accomplish an objective a smaller team could not.

Many stages of emergent strategy can be mapped to behavioral shifts in the way agents interact with the tools in their environment, similar to Leibo et al. (2017); Perolat et al. (2017). We therefore track basic statistics about the agents' interaction with objects during training, shown in Figure 3. For instance, as the hiders learn to build forts, they move and lock boxes much more during the preparation phase. Similarly, as the seekers learn to move and use ramps, the ramp movement in the main phase of the game increases, and as they learn to "box surf" there is a slight increase in the box movement during the main phase of the game. Finally, as the hiders learn to defend against this strategy by locking all boxes in place, the number of locked boxes in the preparation phase increases.

We found that scale plays a critical role in enabling progression through the emergent autocurricula in hide-and-seek. The default model, which uses a batch size of 64,000 and 1.6 million parameters, requires 132.3 million episodes (31.7 billion frames) over 34 hours of training to reach stage 4 of the skill progression, i.e. ramp defense. In Figure 4 we show the effect of varying the batch size in our agents ability to reach stage 4. We find that larger batch sizes lead to much quicker training time by virtue of reducing the number of required optimization steps, while only marginally affecting sample efficiency down to a batch size of 32,000; however, we found that experiments with batch sizes of 16,000 and 8,000 never converged.

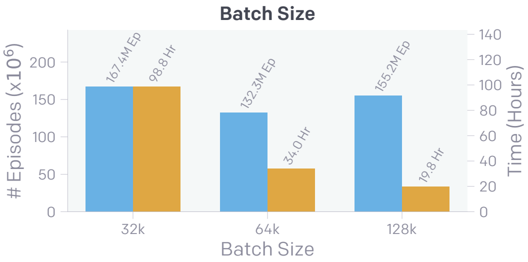


Figure 4: Effect of Scale on Emergent Autocurricula. Number of episodes (blue) and wall clock time (orange) required to achieve stage 4 (ramp defense) of the emergent skill progression presented in Figure 1. Batch size denotes number of chunks, each of which consists of 10 contiguous transitions (the truncation length for backpropagation through time).

We find the emergent autocurriculum to be fairly robust as long as we randomize the environment during training. If randomization is reduced, we find that fewer stages of the skill progression emerges, and at times less sophisticated strategies emerge instead (e.g. hiders can learn to run away and use boxes as moveable shields.); see Appendix A.2 for more details. In addition, we find that design choices such as the minimum number of elongated boxes or giving each agent their own locking mechanism instead of a team based locking mechanism can drastically increase the sample complexity. We also experimented with adding additional objects and objectives to our hide-and-seek environment as well as with several game variants instead of hide-and-seek (see Appendix A.6). We find that these alternative environments also lead to emergent tool use, providing further evidence that multi-agent interaction is a promising path towards self-supervised skill acquisition.

6 Evaluation

In the previous section we presented evidence that hide-and-seek induces a multi-agent autocurriculum such that agents continuously learn new skills and strategies. As is the case with many unsupervised reinforcement learning methods, the objective being optimized does not directly incentivize the learned behavior, making evaluation of those behaviors nontrivial. Tracking reward is an insufficient evaluation metric in multi-agent settings, as it can be ambiguous in indicating whether agents are improving evenly or have stagnated. Metrics like ELO (Elo, 1978) or Trueskill (Herbrich et al., 2007) can more reliably measure whether performance is improving relative to previous policy versions or other policies in a population; however, these metrics still do not give insight into whether improved performance stems from new adaptations or improving previously learned skills. Finally, using environment specific statistics such as object movement (see Figure 3) can also be ambiguous, e.g. the choice to track absolute movement does not illuminate which direction agents moved, and designing sufficient metrics will become difficult and costly as environments scale.

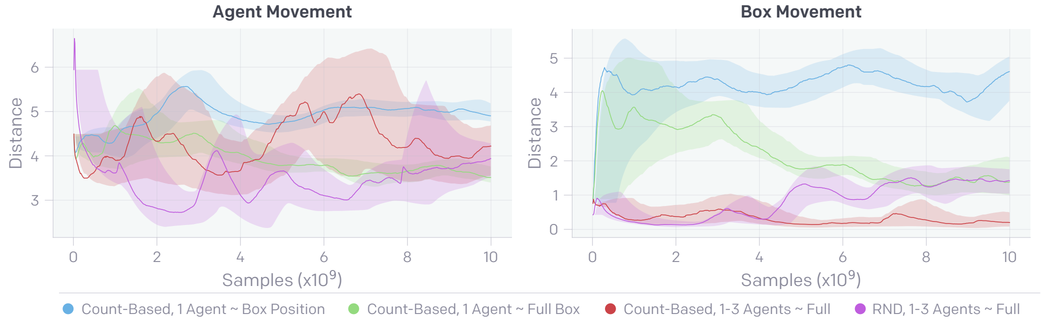


Figure 5: Behavioral Statistics from Count-Based Exploration Variants and Random Network Distillation (RND) Across 3 Seeds. We compare net box movement and maximum agent movement between state representations for count-based exploration: Single agent, 2-D box location (blue); Single agent, box location, rotation and velocity (green); 1-3 agents, full observation space (red). Also shown is RND for 1-3 agents with full observation space (purple). We train all agents to convergence as measured by their behavioral statistics.

In Section 6.1, we first qualitatively compare the behaviors learned in hide-and-seek to those learned from intrinsic motivation, a common paradigm for unsupervised exploration and skill acquisition. In Section 6.2, we then propose a suite of domain-specific intelligence tests to quantitatively measure and compare agent capabilities.

6.1 Comparison to Intrinsic Motivation

Intrinsic motivation has become a popular paradigm for incentivizing unsupervised exploration and skill discovery, and there has been recent success in using intrinsic motivation to make progress in sparsely rewarded settings (Bellemare et al., 2016; Burda et al., 2019b). Because intrinsically motivated agents are incentivized to explore uniformly, it is conceivable that they may not have meaningful interactions with the environment (as with the "noisy-TV" problem (Burda et al., 2019a)). As a proxy for comparing meaningful interaction in the environment, we measure agent and object movement over the course of an episode.

We first compare behaviors learned in hide-and-seek to a count-based exploration baseline (Strehl & Littman, 2008) with an object invariant state representation, which is computed in a similar way as in the policy architecture in Figure 2. Count-based objectives are the simplest form of state density based incentives, where one explicitly keeps track of state visitation counts and rewards agents for reaching infrequently visited states (details can be found in Appendix D). In contrast to the original hide-and-seek environment where the initial locations of agents and objects are randomized, we restrict the initial locations to a quarter of the game area to ensure that the intrinsically motivated agents receive additional rewards for exploring.

We find that count-based exploration leads to the largest agent and box movement if the state representation only contains the 2-D location of boxes: the agent consistently interacts with objects and learns to navigate. Yet, when using progressively higher-dimensional state representations, such as box location, rotation and velocity or 1-3 agents with full observation space, agent movement and, in particular, box movement decrease substantially. This is a severe limitation because it indicates that, when faced with highly complex environments, count-based exploration techniques require identifying by hand the "interesting" dimensions in state space that are relevant for the behaviors one would like the agents to discover. Conversely, multi-agent self-play does not need this degree of supervision. We also train agents with random network distillation (RND) (Burda et al., 2019b), an intrinsic motivation method designed for high dimensional observation spaces, and find it to perform slightly better than count-based exploration in the full state setting.

6.2 Transfer and Fine-tuning as evaluation

We propose to use transfer to a suite of domain-specific tasks in order to asses agent capabilities. To this end, we have created 5 benchmark intelligence tests that include both supervised and reinforce-

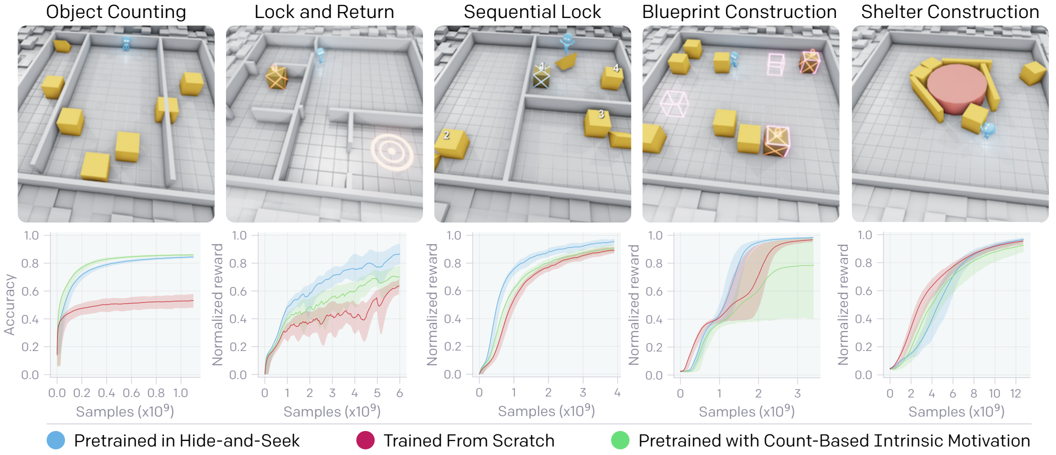


Figure 6: Fine-tuning Results. We plot the mean normalized performance and 90% confidence interval across 3 seeds smoothed with an exponential moving average, except for Blueprint Construction where we plot over 6 seeds due to higher training variance.

ment learning tasks. The tests use the same action space, observation space, and types of objects as in the hide-and-seek environment. We examine whether pretraining agents in our multi-agent environment and then fine-tuning them on the evaluation suite leads to faster convergence or improved overall performance compared to training from scratch or pretraining with count-based intrinsic motivation. We find that on 3 out of 5 tasks, agents pretrained in the hide-and-seek environment learn faster and achieve a higher final reward than both baselines.

We categorize the 5 intelligence tests into 2 domains: cognition and memory tasks, and manipulation tasks. We briefly describe the tasks here; for the full task descriptions, see Appendix C. For all tasks, we reinitialize the parameters of the final dense layer and layernorm for both the policy and value networks.

Cognition and memory tasks:

In the *Object Counting* supervised task, we aim to measure whether the agents have a sense of object permanence; the agent is pinned to a location and watches as 6 boxes each randomly move to the right or left where they eventually become obscured by a wall. It is then asked to predict how many boxes have gone to each side for many timesteps after all boxes have disappeared. The agent's policy parameters are frozen and we initialize a classification head off of the LSTM hidden state. In the baseline, the policy network has frozen random parameters and only the classification head off of the LSTM hidden state is trained.

In *Lock and Return* we aim to measure whether the agent can remember its original position while performing a new task. The agent must navigate an environment with 6 random rooms and 1 box, lock the box, and return to its starting position.

In *Sequential Lock* there are 4 boxes randomly placed in 3 random rooms without doors but with a ramp in each room. The agent needs to lock all the boxes in a particular order — a box is only lockable when it is locked in the correct order — which is unobserved by the agent. The agent must discover the order, remember the position and status of visited boxes, and use ramps to navigate between rooms in order to finish the task efficiently.

Manipulation tasks: With these tasks we aim to measure whether the agents have any latent skill or representation useful for manipulating objects.

In the *Construction From Blueprint* task, there are 8 cubic boxes in an open room and between 1 and 4 target sites. The agent is tasked with placing a box on each target site.

In the *Shelter Construction* task there are 3 elongated boxes, 5 cubic boxes, and one static cylinder. The agent is tasked with building a shelter around the cylinder.

Results: In Figure 6 we show the performance on the suite of tasks for the hide-and-seek, count-based, and trained from scratch policies across 3 seeds. The hide-and-seek pretrained policy performs slightly better than both the count-based and the randomly initialized baselines in *Lock and Return*, *Sequential Lock* and *Construction from Blueprint*; however, it performs slightly worse than the count-based baseline on *Object Counting*, and it achieves the same final reward but learns slightly slower than the randomly initialized baseline on *Shelter Construction*.

We believe the cause for the mixed transfer results is rooted in agents learning skill representations that are entangled and difficult to fine-tune. We conjecture that tasks where hide-and-seek pretraining outperforms the baseline are due to reuse of learned feature representations, whereas better-than-baseline transfer on the remaining tasks would require reuse of learned skills, which is much more difficult. This evaluation metric highlights the need for developing techniques to reuse skills effectively from a policy trained in one environment to another. In addition, as future environments become more diverse and agents must use skills in more contexts, we may see more generalizable skill representations and more significant signal in this evaluation approach.

In Appendix A.5 we further evaluate policies sampled during each phase of emergent strategy on the suite of targeted intelligence tasks, by which we can gain intuition as to whether the capabilities we measure improve with training, are transient and accentuated during specific phases, or generally uncorrelated to progressing through the autocurriculum. Noteably, we find the agent's memory improves through training as indicated by performance in the navigation tasks; however, performance in the manipulation tasks is uncorrelated, and performance in object counting changes seems transient with respect to source hide-and-seek performance.

7 Discussion and Future Work

We have demonstrated that simple game rules, multi-agent competition, and standard reinforcement learning algorithms at scale can induce agents to learn complex strategies and skills. We observed emergence of as many as six distinct rounds of strategy and counter-strategy, suggesting that multi-agent self-play with simple game rules in sufficiently complex environments could lead to open-ended growth in complexity. We then proposed to use transfer as a method to evaluate learning progress in open-ended environments and introduced a suite of targeted intelligence tests with which to compare agents in our domain.

Our results with hide-and-seek should be viewed as a proof of concept showing that multi-agent autocurricula can lead to physically grounded and human-relevant behavior. We acknowledge that the strategy space in this environment is inherently bounded and likely will not surpass the six modes presented as is; however, because it is built in a high-fidelity physics simulator it is physically grounded and very extensible. In order to support further research in multi-agent autocurricula, we are open-sourcing our environment code.

Hide-and-seek agents require an enormous amount of experience to progress through the six stages of emergence, likely because the reward functions are not directly aligned with the resulting behavior. While we have found that standard reinforcement learning algorithms are sufficient, reducing sample complexity in these systems will be an important line of future research. Better policy learning algorithms or policy architectures are orthogonal to our work and could be used to improve sample efficiency and performance on transfer evaluation metrics.

We also found that agents were very skilled at exploiting small inaccuracies in the design of the environment, such as seekers surfing on boxes without touching the ground, hiders running away from the environment while shielding themselves with boxes, or agents exploiting inaccuracies of the physics simulations to their advantage. Investigating methods to generate environments without these unwanted behaviors is another import direction of future research (Amodei et al., 2016; Lehman et al., 2018).

Acknowledgments

We thank Pieter Abbeel, Rewon Child, Jeff Clune, Harri Edwards, Jessica Hamrick, Joel Liebo, John Schulman and Peter Welinder for their insightful comments on this manuscript. We also thank Alex Ray for writing parts of our open sourced code.