ColorGrid: A Multi-Agent Non-Stationary Environment for Goal Inference and Assistance

Anonymous submission

Abstract

Autonomous agents' interactions with humans are increasingly focused on adapting to constantly changing preferences to improve assistance in real-world tasks. Effective agents must learn to accurately infer human goals, which are often hidden, to collaborate well. However, existing Multi-Agent Reinforcement Learning (MARL) environments lack the necessary attributes required to rigorously evaluate these agents' learning capabilities. To this end, we introduce COLORGRID, a novel MARL environment with customizable non-stationarity, asymmetry, and reward structure. We investigate the performance of state-of-the-art (SOTA) MARL algorithms within COLORGRID, conducting extensive ablations which reveal that COLORGRID, particularly with simultaneous non-stationary and asymmetric goals between a "leader" agent representing a human and a "follower" assistant agent, presents a significant challenge unsolved by SOTA MARL algorithms. One notable ablation demonstrates that the cost of exploration directly affects an agent's ability to adapt.

1 Introduction

The integration of autonomous assistants into real-world tasks with humans necessitates robust guidance and correction. Nascent work is examining how to train autonomous agents to collaborate with humans of diverse skill levels, yet these studies always assumed a fixed objective. In the real world, human goals change – consider a surgeon encountering unexpected complications during a surgery. A robotic surgery assistant must seamlessly follow suit, adapting on-the-fly. As such, we contend that training collaborative agents to adaptively help humans in real-world tasks requires adapting to the human's hidden and changing goals. We develop a benchmark that tests this aspect of human-AI coordination, creating a test-bed that enables focusing on the core problems of inferring goals in real-time. Agents that can make progress on this benchmark take a step toward helping humans in the real-world.

Many prior works explore zero-shot coordination with other agents and with humans (Dennis et al. 2021; Strouse et al. 2022; Jeon, Losey, and Sadigh 2020; Nikolaidis, Hsu, and Srinivasa 2017), but their goals are typically modelled as static throughout a given episode. Other works study social agents and their collaboration when augmented with message passing (Weil et al. 2024; Shrestha and Moore 2014), but in reality we seek autonomous agents that do not require explicit messages to seamlessly and quickly transition between goals when assisting humans with arbitrary tasks. In settings such as elder care, it is unrealistic to expect humans to describe new goals in detail to the assistant, since the tasks may be time sensitive and require immediate attention.

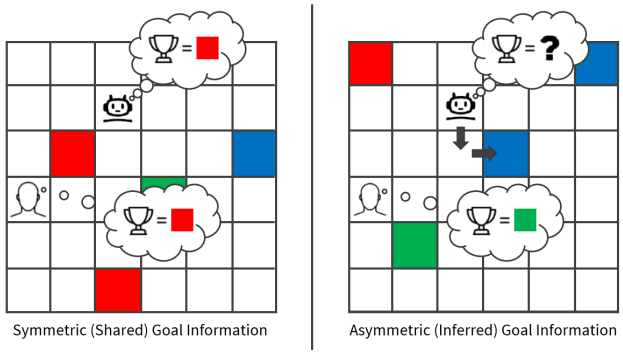

Figure 1: COLORGRID visualization distinguishing agent learning in a symmetric vs. asymmetric scenario.

In competitive Multi-Agent Reinforcement Learning (MARL) domains, self-play (Silver et al. 2017) has been shown to reach human-level play without human data using model-free reinforcement learning. Self-play excels in competition, but works poorly for *cooperation*. It produces agents that can only coordinate with themselves, and thus break or display "stubbornness" when needing to work with other agents who have different behavior. In contrast, population play (Jaderberg et al. 2018) trains a group of agents who play in teams or against each other; however, works using these methods typically treat the high-level goal as static throughout an episode and focus on coordination at the strategy level.

In this work, we present our two-fold contribution to address the lack of focus on learning to infer non-stationary goals: (1) Releasing COLORGRID, a novel MARL environment with configurable levels of non-stationarity, reward sparsity, and asymmetry, and reward structure and (2) Exploring how Independent Proximal Policy Optimization (IPPO) a SOTA MARL algorithm (Schulman et al. 2017) performs in COLORGRID with broad ablations on envi-

ronment configurations, reward shaping, and learning parameters. We demonstrate that with non-stationary goals and without providing explicit goal information to the "follower" autonomous assistant, COLORGRID is a challenging MARL environment currently unsolved by IPPO which has been shown to perform well in collaborative settings (Yu et al. 2021). Particularly, we find that IPPO is insufficient when asymmetry regarding goal information is present but also in the symmetric setting when the cost to explore is immediately set to the maximum, for an episode, rather than a gradual increase.

The inability of SOTA MARL algorithms to address the problems of coordinating with a partner that has non-stationary hidden goals highlight the fact that much additional research is needed into MARL algorithms that can truly assist human partners. We hope that, by sharing the code for this benchmark, we can spur more research progress towards real-world human-AI cooperation

2 Related Work

2.1 Zero-shot coordination with humans

To encourage strategic diversity while still averting the need to collect human data, fictitious co-play (Strouse et al. 2022) trains agents against a population of self-play agents and prior training checkpoints in the game Overcooked (Carroll et al. 2019). Several works in robotics have also modeled shifting human goals to improve robot-human cooperation. For instance, Jeon, Losey, and Sadigh (2020) develop a shared autonomy method for precise robotic manipulation, reinterpreting user inputs in a latent space based on a confidence-conditioned prediction of the human's goal. In the setting of human-robot cooperation where the two agents may prefer different strategies to achieve a given task, Nikolaidis, Hsu, and Srinivasa (2017) model the human's adaptability as a latent variable in a POMDP (Partially Observable Markov Decision Process) (Smallwood and Sondik 1973) and adjust the robot's weighting between its and the human's preferred strategies to maximize the team's rewards depending on the human's capability to switch modes. Shared autonomy, a robot estimating operator intent based on operator inputs, can benefit from modeling those intents as Bayesian (Jain and Argall 2018).

Moreover, prior work such as Yell At Your Robot (Shi et al. 2024) explore real-time corrections of low-level robot actions through natural language. While providing impressive adaptability, this approach requires a human to constantly monitor the robot for potential mistakes, which is unrealistic in some downstream applications. Most of these works model human intent explicitly with hand-crafted heuristics. Our contribution is a step towards extending this adaptability to implicit higher-level goals in truly cooperative environments where both humans and autonomous agents collaborate without the need for explicit feedback.

2.2 Multi-Agent Environments

Distinct multi-agent environments have been designed to understand the challenges of agent interactions. This section outlines the main benefits of COLORGRID over other environments. The popular Overcooked environment (Carroll et al. 2019) is where agents (autonomous and human) use shared ingredients to produce and deliver soup. Across different kitchen layouts, collaboration between agents is explicitly required to complete the task, including both high-level strategic and low-level mobility collaboration. In contrast, COLORGRID involves the complexity of having a dynamic objective via goal block switching, within an episode, rather than agents coordinating how to share resources, which could potentially be more complex.

The Multi-Agent Particle Environments (MPE) introduced by Mordatch and Abbeel (2018) initialize every agent with one or more private, physically-grounded goals that may require cooperation from other agents. Examples of goals in this environment include going to a landmark and looking at a location. Additionally, there is a channel through which agents can broadcast messages to others for a small cost. MPE use a continuous representation of physical locations while representing time as discrete. Distinct from MPE's vector world, level-based foraging environments (Albrecht and Ramamoorthy 2015) are represented in a discrete grid-world. Our mechanism for assigning penalty fundamentally differs from level-based foraging environments (Albrecht and Ramamoorthy 2015), which only give positive rewards from collecting items but impose an additional penalty every step to incentivize rapid collection. The Multi-Robot Warehouse environment (Papoudakis et al. 2021) challenges agents to learn long chains of actions to complete deliveries and earn sparse rewards. While these environments have implicit non-stationarity from containing multiple agents, COLORGRID introduces non-stationarity from the actual goal changing. COLORGRID also doesn't have a communication channel between agents, i.e. the "follower" agent isn't notified when the "leader" "learns" of the goal switching since in real life scenarios it's not always possible to communicate, due to inaccessibility for instance.

The Coingrid environment (Raileanu et al. 2018) evaluates two symmetric agents on a fully cooperative task. Specifically, there is an 8×8 grid containing 4 coins for each of 3 colors. The two agents are each randomly assigned a color, with the goal of collecting either its assigned color or its partner agent's color to collectively maximize the reward without picking up non-assigned colors. While Coingrid does have hidden goals, our approach focuses on the case where those goals can change (and can also be hidden).

The Watch-And-Help (WAH) challenge (Puig et al. 2021) tests how much an agent Bob can accelerate task completion after observing another agent, Alice, demonstrate the task end-to-end. WAH explicitly enumerates the goals of a task as a set of predicates and their counts. In contrast, our motivation is to solve a continuous task without an explicit description. Rather than clearly delineating the start and end of a task, we design COLORGRID to encourage a helper agent to act as soon as the intended goal is clear. Furthermore, observing a task to infer the goal and then helping Alice finish the same task as fast as possible in a different episode is not an evaluation that supports our goal of real-time, continuous adaptation within an episode.

3 Method

3.1 Environment

We introduce COLORGRID, a novel environment to evaluate multi-agent reinforcement learning without explicit message passing. This environment features complex agent learning interactions and state dynamics and is implemented using PettingZoo (Terry et al. 2021), a platform that supports creating environments for MARL problems.

In COLORGRID, agents are situated in a 32×32 grid-world, populated with three colors of "blocks" that yield a reward for an agent when it moves into its grid cell; agents can move up, down, left, and right. At every timestep, one of the three colors gives a $+1$ reward (the "goal" block), while the other two yield a -1 reward ("incorrect" blocks). Also, the "goal" block switches with some probability at each timestep, introducing non-stationarity but it is slow enough such that the "follower" agent can reach it if it switches. Overall, the objective is for the follower to learn to travel to the nearest but different goal block grid square from the leader.

Whenever a block is collected in COLORGRID, a new block of the same color is spawned in a random empty cell, maintaining a constant block density and uniform color distribution throughout time. We find that sparse rewards and the incorrect block penalty make this environment difficult for agents to learn due to the cost of exploration.

Customizability Users of COLORGRID configure the environment size, the reward and penalty ascribed by the goal and incorrect blocks, the density of blocks filling the grid, and the reward shaping functions applied. The latter two affect reward sparsity, which defaults to filling 10% of the grid with three colors uniformly at random. The probability of the goal block changing is also customizable.

	Observability	Non-Stationary Environment	Reward Sparsity	Symmetric Goals	Hidden Goals
COLORGRID	Partial/Full	Yes	Sparse	Yes	Yes
Multi-Agent Particle	Partial/Full	Yes	Dense	No	No
Level-Based Forage	Partial/Full	No	Sparse	Yes	No
Multi-Robot Warehouse	Partial	No	Sparse	Yes	No
Overcooked	Full	No	Sparse	Yes	No
Coingrid	Full	No	Sparse	Yes	No

Table 1: Comparisons between COLORGRID and related environments. "Non-stationary Environment" refers to whether the core environment changes over time, without considering the non-stationarity introduced by multiple learning agents. In our case, blocks reappear in the environment, and the goal block color may switch. "Hidden Goals" refers to agents receiving different amounts of state information; in our case, providing the goal block color to the leader but not the follower. Environment attributes for rows 2, 3, and 4 are referenced from Papoudakis et al. (2021).

Asymmetry In the multi-agent setting of COLORGRID, we define one agent to be the "leader" and the other to be the "follower", to simulate the scenario where a robot follower must assist a human leader. The leader is always informed which block color is the goal, but this is hidden from the follower when the asymmetry option is activated. Adding asymmetry makes this environment significantly more challenging, as the follower must learn to infer the leader's goal from the leader's trajectory only.

State Representation We represent states in a grid representation format, with 5 channels of 32×32 matrices. Each channel is a binary mask for every cell on the 32×32 board, representing whether it contains the color or agent (3 block position and 2 agent position masks result in 5 channels). Additionally, the goal color is provided as a one-hot vector where each index represents a color present in COLORGRID. Exposing options to toggle non-stationarity and asymmetry make COLORGRID a valuable resource for comparing reinforcement learning algorithms between multiple levels of environment complexity.

3.2 Agent Implementation

We implement our agents as Actor-Critic neural networks (Haarnoja et al. 2018), with a shared portion that computes features using convolutional and linear layers before feeding into the separate policy and value networks. We use stride size of 1 for all convolutions, Leaky ReLU (Maas et al. 2013) as the activation function following all convolution layers, and TanH (LeCun 1989) following all linear layers. We referenced Ndousse et al. (2021) for our initial architecture implementation, modifying it to use a stride of 1 for all convolutions (since our state representation is symbolic rather than pixel-based), concatenate a vector to signal the goal block, and add an auxiliary objective network as described in **Experiments**. Our exact network architecture is as follows

As our goal is to produce agents which can cooperate effectively, the performance metric is the sum of rewards achieved by each agent. We train agents using Independent Proximal-Policy Optimization (Schulman et al. 2017), to ensure that the individual rewards observed by each agent are correlated with its own actions. For this reason, we don't use Multi-Agent PPO (MAPPO) because the value and policy networks between the leader and follower can't be shared.

3.3 Baseline Agents

The main baseline we experiment with is having the leader use the A* search algorithm, where the cost of a path is the length of the shortest path to a goal block (ties are broken arbitrarily) [1]. The follower also uses a greedy A* policy, but only updates its current goal with the true reward color when the leader collects a block[2].

4 Experiments

4.1 Experimental Design

Here we describe the set of environment and training parameters to conduct the experiments described in **Results**.

[1]This is an admissible heuristic, so this implementation of A* is guaranteed to perform optimally.

[2]The follower's first action of an episode is to move to any neighboring empty space until the leader collects a block.

We run all experiments with a random seed of 0. We focus on completing a breadth of experiments with the resources available rather than running fewer experiments with many seeds. We fix a set of standard hyperparameters for the IPPO algorithm, shown in Appendix section A.2, and maintain a constant learning rate rather than annealing over time. For all experiments, we use a block density of 10% unless otherwise mentioned.

Architectural

Auxiliary Supervised Loss We add an auxiliary supervised prediction task, in which a two-layer MLP predicts the goal block color from an input of a feature representation output by the shared network. Auxiliary model-based loss objectives have been extensively studied in the reinforcement learning literature, and in line with previous works we find that this component improves training performance and stability (Ndousse et al. 2021; Ke et al. 2019; Weber et al. 2018; Shelhamer et al. 2017; Jaderberg et al. 2016; Krupnik, Mordatch, and Tamar 2019; Hernandez-Leal, Kartal, and Taylor 2019). Unless specified otherwise, we use a default coefficient of $\kappa = 0.2$ to add the supervised cross-entropy loss to the main PPO loss. With this additional objective, the full loss is described by the below equation, where $c_i = 1$ if the i'th color is the current goal, or 0 otherwise.

$$L = L_{\text{PPO}} - \kappa \sum_{i=1}^{3} c_i \cdot \log \hat{c}_i \tag{1}$$

Goal Color Concatenation We experiment with the neural network architecture, specifically with where in the forward pass to append a one-hot vector describing the current color of the goal block. We either concatenate this vector "Early" (immediately after flattening the output from the convolutional network) or "Late" (after projecting the flattened output into a lower-dimensional space). See A.1 for the full architecture including the concatenation points.

Environmental

Cost of Exploration We consider three cases of COLORGRID with an expected reward that is either positive, zero, or negative (in expectation, varying the cost of exploration). See A.4 for the specific reward values.

Penalty Annealing To lower the cost of exploration at the beginning while the agents are still learning, we linearly raise the reward penalty for collecting blocks of incorrect colors, i.e. the block penalty coefficient increases from 0 to 1 between timesteps 4M and 10M, of the 80M total timesteps.

Distance Reward Shaping We introduce a constant penalty when the follower is too close to the leader, incentivizing the follower to collect blocks not targeted by the leader. We use a threshold of 10 Manhattan distance, roughly $\frac{1}{3}$ of the environment length.

Potential Field Reward Shaping We introduce another reward shaping term, inspired by electric potential fields. Just like charged particles, blocks emit a negative or positive value, which increases in magnitude as an agent approaches them. This eliminates sparse rewards, while maintaining an order of magnitude greater reward for collecting a goal block.

Goal Block Switch Probability. Each environment step, COLORGRID has a chance to switch the color of the goal block, which simulates the leader changing their goal. The default block switching probability parameter is $\frac{2}{3} \cdot \frac{1}{32} = \frac{1}{48}$. The $\frac{2}{3}$ is motivated by the $\frac{1}{3}$ chance to randomly select the same color, while the $\frac{1}{32}$ gives the leader enough time steps to collect a new goal block color before switching again (in expectation, it takes 32 environment steps before the goal block switches, and 32 steps is enough to cross half the default 32x32 environment).

4.2 Results and Findings

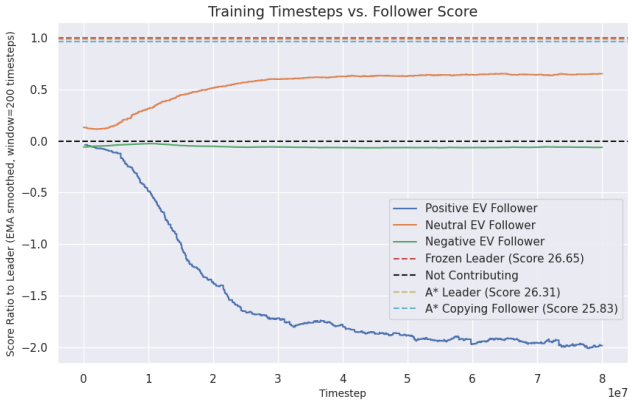

Figure 2: Using a frozen expert leader trained with IPPO, we train a cold-started follower varying the training reward structure such that random block collection would have positive, neutral, and negative expected value (EV). Dotted lines are baselines of the expert leader, A* search leader, and A* search follower which routes to the last color picked up by the A* leader. We use seed 0 for these comparisons, except for the A* agent scores which are averaged across 100 seeds.

Cost of Exploration Figure 2 shows the effect of training an IPPO follower in asymmetric environments with a positive, zero, and negative expected cost of exploration, learning with a frozen leader policy trained with the best hyperparameters from 2 in **IPPO Findings**. We train the follower without an auxiliary objective or penalty annealing to purely explore the effect of varying the reward structure.

Upon visualizing the environments, we observe the follower agent behavior:

1. In an environment with a positive expected value of exploration, the follower learns to pick up all blocks, whether they are the color of the goal or not.
2. In an environment with a negative expected value of exploration, the follower learns to avoid picking up any blocks, and does not distinguish which are the goal.

3. Only in an environment with an expected cost of exploration of zero does the follower learn to pick up the correct block color often, and less frequently collecting incorrect blocks.

Note that in these experiments, the goal block color switches randomly, implying that the follower in the Neutral environment occasionally learned to correctly observe the leader's behavior to inform which goal to collect.

We hypothesize the Pessimistic follower avoids collecting any blocks due to a sparsity of reward: it is rare that the leader collects the goal block, and the follower randomly chooses the goal block shortly after. To maximize reward in an environment where uninformed block pickup occurs, the follower avoids picking up blocks and incurring the penalty.

On the contrary, the Optimistic follower collects every block in its path. We infer the follower learns a local optimum that collecting blocks uniformly at random yields positive reward, and thus never learns the global optimum of only collecting the goal block.

Even in a Neutral environment, which we find to be the best environment for learning a follower agent using the SOTA IPPO algorithm, the agent still collects incorrect blocks. In a real-world scenario with high consequences for mistaken actions, this would be unacceptable for a human helper. Though the Negative EV follower never collects incorrect blocks, it also never collects goal blocks, which is harmless but not helpful.

We believe the real world is a Pessimistic (Negative EV) environment if pursuing random goals, where many actions exist that cause harm, and only a few are helpful. This, combined with the SOTA IPPO algorithm learning poorly in a Pessimistic environment, motivates our study of how to get a follower to learn in a Pessimistic environment for the rest of the paper.

We observe that the frozen leader score is slightly above the baseline A* leader score, showing that the SOTA IPPO algorithm learns a policy achieving a score on-par with classical heuristic-based search algorithms. The A* copying follower performance, on the other hand, significantly outperforms all follower agents trained with IPPO, demonstrating that even with an expected cost of exploration of zero, IPPO is insufficient to learn to infer the leader's goal.

Auxiliary Loss κ	**Penalty Annealing**	**Converged** Σ **Reward**
0.2	4M-10M	**48.8**
0.2	None	-0.7
0	None	-1.0
0	4M-10M	32.8

Table 2: "Converged Sum Reward" is the final net reward over 128 timesteps averaged over 16 environments. All runs use 10% block density, non-stationary goal block color with 2% chance of switching at each step, and *symmetric* information between the leader and follower. We take the converged reward values after 80M train timesteps and average across 3 seeds.

IPPO Findings

Annealing the incorrect block penalty enables learning through early exploration. Early exploration is key for the agents to learn. We find that training with a constant incorrect penalty results in a negative reward after 80M timesteps (see 2). Without penalty annealing, because the environment is pessimistic with negative expected value, agents converge adversely toward avoiding blocks altogether whereas introducing penalty annealing creates a better learning dynamic.

Supervised goal prediction stabilizes training with sparse rewards. We find that there is a clear benefit of the auxiliary goal prediction objective through an ablation with 5% block density, the sparsest environment configuration that we explore. In this experiment we use symmetric information, providing the current goal information to both the leader and follower agents. With such sparse reward signal, naive PPO training collapses to inaction once the block penalty starts increasing at 4M timesteps. Similar to the case when training without penalty annealing, these agents degenerate to a behavior of avoiding collecting any blocks at all, minimizing the penalties incurred but failing to achieve any positive reward.

However, when including the supervised auxiliary objective with $\kappa = 0.2$, the agents are able to recover and learn to collect the current goal blocks while avoiding the other colors. Intuitively, the supervised loss forces the feature representation output by the shared network to encode the goal information, thereby allowing the policy network to perform optimally given the current goal. We additionally conduct an experiment with a higher coefficient $\kappa = 0.4$. We find that weighting the auxiliary loss too heavily reduces performance, demonstrating the importance of tuning this hyperparameter.

Unbalanced learning collapses to a single-agent solution. We observe that when one agent learns faster than the other, it collects goal blocks which are close to the other agent, removing opportunities to receive positive reward signal, resulting in a single-agent solution where the slower agent does not collect any blocks. As an attempt to get agents to learn in the asymmetric setting, we introduced distance reward shaping, penalizing the leader from being close to the follower. While this does allow the follower to explore during the portion of training with no block penalty, once the penalty starts increasing it again collapses to not collecting any blocks.

Dense reward shaping does not solve goal inference for Pessimistic environments. To combat reward sparsity, we experimented with applying a reward shaping term which provides rewards based on proximity to goal blocks and non-goal blocks, similar to an electric potential field where goal blocks represent positive potential and incorrect blocks represent negative potential. Still, we see that the follower agent performance is poor; we hypothesize this to be the case because despite denser rewards, the negative expected cost of exploration still heavily punishes exploration.

5 Discussion & Future Work

Avenues for future work broadly fall into three categories: environmental, agentic, and algorithmic design. All three support developing multi-agent systems that collaborate more effectively with humans and other agents in zero-shot settings.

5.1 Environmental Developments

COLORGRID is populated with three colors of blocks. This paper used a uniform distribution of blocks, experimenting with the distribution of blocks, and thus reward signal frequency, is a natural direction. This paper also assigned a fixed reward value upon block collection by an agent, experimenting with sampling from a reward distribution, and thus reward signal magnitude, is another natural direction.

A reward signal occurs each time an agent collects a block. Rather than populating COLORGRID uniformly with 3% of each color, generating more challenging environments (varying density by color to simulate goal rarity, evaluating on dense and structured mazes, disjoint hemispheres where blocks collected in one respawn in the other, and varying the probability of the leader changing the goal block). To aid training, we could use POET (Wang et al. 2019) to generate a curriculum of environments that adversarially challenge the follower.

Every reward signal in COLORGRID is currently a fixed value, determined by block color and the current goal. A more realistic model would be to sample the returned reward from a distribution, with mean and variance determined by color. This further increases follower robustness, as the expected reward of a block no longer represents the guaranteed reward received. We expect this to incentivize subtle tradeoffs between block selection, depending on the implicitly learned block swap probability. This could also help understand the answer to which learning frequencies are difficult to learn from.

Finally, we motivated our study of Pessimistic environments by the real-world analogue of a critical situation where mistakes are costly. We recognize that all experiments may be repeated on Optimistic and Neutral environments. An even more representative study would repeat our experiments across multiple seeds and plot standard error bars to strengthen our claims about differences in learned behavior.

5.2 Agentic Developments

In **Cost of Exploration**, we run all follower training runs without an auxiliary supervised objective or penalty annealing; a clear next step is to run hyperparameter experiments in this setting similar to those in **IPPO Findings** to find the optimal configuration.

COLORGRID can be used to train more than one follower, where they learn from more than one leader. In the real world, we expect humans to outnumber autonomous agents, motivating the need for agents trained to prioritize which leader they assist. In this setting, some followers are helping a greater number of leaders, where different leaders have different goal blocks. A follower assisting two leaders with conflicting goals is another niche but critical scenario to study.

A significant limitation of our agents is the requirement to define an explicit set of possible goals. Instead of a separate channel representations for each block, a single channel with discrete integers would better handle a large and fluctuating number of potential goals in the environment. This would enable training an agent in a COLORGRID instance where the distribution of goals is changing (e.g. there are mostly red blocks at the start, and mostly blue blocks at the end). To flexibly handle introduced and removed goals, Thought Cloning (Hu and Clune 2024) is well-suited to describe goals, embed them, and condition on those embeddings.

5.3 Algorithmic Developments

COLORGRID's novel and challenging multi-agent scenarios further emphasize the need for algorithmic developments beyond IPPO. The initial Optimistic, Neutral, and Pessimistic environments all led to imperfect learned behavior by the IPPO-driven follower agent. One direction is to warm-start the follower by behavior cloning the A* copying baseline. Another is to use the diverse policy data collected from baseline A* agents to perform Implicit Q-Learning (IQL), and use an objective inspired by COMA (Foerster et al. 2017): $Q(a_R, s) = Q(a_H, a_R, s) - Q(a_H, s)$, where a_R is a robot action and a_H is the human action, which would enable calculating the value for a robot helper. We expect this to help the follower separate its unique contribution to the reward from the leader's, allowing it to learn more effective behaviors.

Another promising algorithmic direction is to use an inverse reinforcment learning (IRL) method, such as BASIS (Abdulhai, Jaques, and Levine 2022), such that the follower actively infers the leader's current goal with IRL in an online setting. In this method, the follower would explicitly learn a model of the leader's reward function and infer its goal rather than implicitly encoding goal inference into the policy. Online IRL might provide similar benefits to our auxiliary supervised objective, but may prove to be a more effective method of goal inference due to more explicitly modeling the leader's reward function.

6 Conclusion

We present COLORGRID, a novel multi-agent environment with changing and hidden goals. We find that it is a challenging MARL environment that IPPO cannnot solve out-of-the-box, and analyze several architectural, environmental, and algorithmic factors that make the environment easier or harder for agents. Notably, we see that the cost of exploration directly affects an agent's ability to learn while collaborating. By making a simplified gridworld environment to focus on studying the keys aspects of this problem from a cooperation perspective and while abstracting away details like image understanding or low-level motor control, COLORGRID is a first step towards human-AI cooperation and social learning. We hope this research helps spur other works to gain more insights into how costlier individual exploration raises the incentive to rely on social learning and

enables the development of AI agents that can adaptively help humans without explicit feedback in real-time and realistic settings where not always known goals are constantly changing. This can impact a wide range of fields and industries including robotic surgery and household assistance. More specifically, a robotic surgery assistant can better respond to patient-specific comfort levels, which could lead to improved patient outcomes, or a robot could help another human set the table while not getting in the way.

A Appendix

A.1 Network Architecture

Shared Convolutional Network:

- Conv2d: 5 input channels, 32 output channels, kernel size of 3
- Conv2d: 32 input channels, 64 output channels, kernel size of 3
- Conv2d: 64 input channels, 64 output channels, kernel size of 3

Shared Projection and Feature Network:

- Linear: Input dimension 43264 + 3, output dimension 192
- Linear: Input dimension 192, output dimension 192
- Linear: Input dimension 192, output dimension 192
- (If asymmetric) LSTM: Input dimension 192, output dimension 192

The first linear layer adds 3 to the input dimension to concatenate the one-hot vector of goal information, or zeros in the case of an asymmetric follower. In **Experiments** we also discuss an alternative architecture where the goal information vector is concatenated in the input to the second linear layer.

Value Network:

- Linear: Input dimension 192, output dimension 64
- Linear: Input dimension 64, output dimension 64
- Linear: Input dimension 64, output dimension 1

Policy Network:

- Linear: Input dimension 192, output dimension 64
- Linear: Input dimension 64, output dimension 64
- Linear: Input dimension 64, output dimension 4

Finally, we also include an auxiliary network to predict the current goal color, which we find in **Experiments** is key to enabling agents to learn under non-stationary goals:

- Linear: Input dimension 192, output dimension 64
- Linear: Input dimension 64, output dimension 3

A.2 PPO Parameters

Parameter	Value
LR	1e-4
N. Envs	16
N. Rollout Steps	128
Gamma	0.99
GAE Lambda	0.95
N. Minibatches	4
Update Epochs	4
Clip Param	0.2
Entropy Coef	0.01
Value Coef	0.5
Target KL	0.01

A.3 Early vs. Late Goal Concatenation

Concatenating goal information earlier in the forward pass is insignificant. We find that, in an environment with 10% block density, concatenating the one-hot goal information vector immediately after flattening the output from the convolutional network doesn't have a substantially higher shared reward than concatenating the vector after projecting the flattened output into a lower-dimensional space. We hypothesize that the difference is negligible because either way, there are enough layers after appending to appropriately encode the goal information into the feature representation.

A.4 Cost of Exploration Reward Values

Goal Reward	**Incorrect Reward**
+4	-1
+2	-1
+1	-1

Table 3: In the Cost of Exploration section, we discuss three cases of COLORGRID that are differentiated by having a positive (optimistic), zero (neutral), or negative (pessimistic) expected value for collecting a random block. Here are the reward values for the three cases, respectively.

A.5 Warmstarting Experiment Parameter Details

For warmstarting, we apply the distance penalty reward shaping term for a distance threshold of 10, penalty 0.25, for 20M timesteps, and penalty annealing from 10M to 20M timestep. We also try penalty 0.5 for distance threshold 10 for 40M time steps and penalty annealing for 4M to 10M.

References

Abdulhai, M.; Jaques, N.; and Levine, S. 2022. Basis for Intentions: Efficient Inverse Reinforcement Learning using Past Experience. arXiv:2208.04919.

Albrecht, S. V.; and Ramamoorthy, S. 2015. A Game-Theoretic Model and Best-Response Learning Method for Ad Hoc Coordination in Multiagent Systems. arXiv:1506.01170.

Carroll, M.; Shah, R.; Ho, M. K.; Griffiths, T. L.; Seshia, S. A.; Abbeel, P.; and Dragan, A. D. 2019. On the Utility of Learning about Humans for Human-AI Coordination. *CoRR*, abs/1910.05789.

Dennis, M.; Jaques, N.; Vinitsky, E.; Bayen, A.; Russell, S.; Critch, A.; and Levine, S. 2021. Emergent Complexity and Zero-shot Transfer via Unsupervised Environment Design. arXiv:2012.02096.

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2017. Counterfactual Multi-Agent Policy Gradients. arXiv:1705.08926.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. arXiv:1801.01290.

Hernandez-Leal, P.; Kartal, B.; and Taylor, M. E. 2019. Agent Modeling as Auxiliary Task for Deep Reinforcement Learning. arXiv:1907.09597.

Hu, S.; and Clune, J. 2024. Thought Cloning: Learning to Think while Acting by Imitating Human Thinking. arXiv:2306.00323.

Jaderberg, M.; Czarnecki, W. M.; Dunning, I.; Marris, L.; Lever, G.; Castañeda, A. G.; Beattie, C.; Rabinowitz, N. C.; Morcos, A. S.; Ruderman, A.; Sonnerat, N.; Green, T.; Deason, L.; Leibo, J. Z.; Silver, D.; Hassabis, D.; Kavukcuoglu, K.; and Graepel, T. 2018. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364: 859 – 865.

Jaderberg, M.; Mnih, V.; Czarnecki, W. M.; Schaul, T.; Leibo, J. Z.; Silver, D.; and Kavukcuoglu, K. 2016. Reinforcement Learning with Unsupervised Auxiliary Tasks. arXiv:1611.05397.

Jain, S.; and Argall, B. 2018. Recursive Bayesian Human Intent Recognition in Shared-Control Robotics. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3905–3912.

Jeon, H. J.; Losey, D. P.; and Sadigh, D. 2020. Shared Autonomy with Learned Latent Actions. arXiv:2005.03210.

Ke, N. R.; Singh, A.; Touati, A.; Goyal, A.; Bengio, Y.; Parikh, D.; and Batra, D. 2019. Learning Dynamics Model in Reinforcement Learning by Incorporating the Long Term Future. arXiv:1903.01599.

Krupnik, O.; Mordatch, I.; and Tamar, A. 2019. Multi-Agent Reinforcement Learning with Multi-Step Generative Models. arXiv:1901.10251.

LeCun, Y. 1989. Generalization and network design strategies.

Maas, A. L.; Hannun, A. Y.; Ng, A. Y.; et al. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, 3. Atlanta, GA.

Mordatch, I.; and Abbeel, P. 2018. Emergence of Grounded Compositional Language in Multi-Agent Populations. arXiv:1703.04908.

Ndousse, K.; Eck, D.; Levine, S.; and Jaques, N. 2021. Emergent Social Learning via Multi-agent Reinforcement Learning. arXiv:2010.00581.

Nikolaidis, S.; Hsu, D.; and Srinivasa, S. S. 2017. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36: 618 – 634.

Papoudakis, G.; Christianos, F.; Schäfer, L.; and Albrecht, S. V. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*.

Puig, X.; Shu, T.; Li, S.; Wang, Z.; Liao, Y.-H.; Tenenbaum, J. B.; Fidler, S.; and Torralba, A. 2021. Watch-And-Help: A Challenge for Social Perception and Human-AI Collaboration. arXiv:2010.09890.

Raileanu, R.; Denton, E.; Szlam, A.; and Fergus, R. 2018. Modeling Others using Oneself in Multi-Agent Reinforcement Learning. arXiv:1802.09640.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Shelhamer, E.; Mahmoudieh, P.; Argus, M.; and Darrell, T. 2017. Loss is its own Reward: Self-Supervision for Reinforcement Learning. arXiv:1612.07307.

Shi, L. X.; Hu, Z.; Zhao, T. Z.; Sharma, A.; Pertsch, K.; Luo, J.; Levine, S.; and Finn, C. 2024. Yell At Your Robot: Improving On-the-Fly from Language Corrections. arXiv:2403.12910.

Shrestha, M.; and Moore, C. 2014. Message-passing approach for threshold models of behavior in networks. *Physical Review E*, 89(2).

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; Lillicrap, T.; Simonyan, K.; and Hassabis, D. 2017. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. arXiv:1712.01815.

Smallwood, R. D.; and Sondik, E. J. 1973. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Oper. Res.*, 21: 1071–1088.

Strouse, D.; McKee, K. R.; Botvinick, M.; Hughes, E.; and Everett, R. 2022. Collaborating with Humans without Human Data. arXiv:2110.08176.

Terry, J.; Black, B.; Grammel, N.; Jayakumar, M.; Hari, A.; Sullivan, R.; Santos, L. S.; Dieffendahl, C.; Horsch, C.; Perez-Vicente, R.; et al. 2021. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 15032–15043.

Wang, R.; Lehman, J.; Clune, J.; and Stanley, K. O. 2019. Paired Open-Ended Trailblazer (POET): Endlessly Generating Increasingly Complex and Diverse Learning Environments and Their Solutions. *CoRR*, abs/1901.01753.

Weber, T.; Racanière, S.; Reichert, D. P.; Buesing, L.; Guez, A.; Rezende, D. J.; Badia, A. P.; Vinyals, O.; Heess, N.; Li, Y.; Pascanu, R.; Battaglia, P.; Hassabis, D.; Silver, D.; and Wierstra, D. 2018. Imagination-Augmented Agents for Deep Reinforcement Learning. arXiv:1707.06203.

Weil, J.; Bao, Z.; Abboud, O.; and Meuser, T. 2024. Towards Generalizability of Multi-Agent Reinforcement Learning in Graphs with Recurrent Message Passing. arXiv:2402.05027.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2021. The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games. arXiv:2103.01955.

B Reproducibility Checklist

Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA)

yes

Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no)

yes

Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper (yes/no)

yes

Does this paper make theoretical contributions? (yes/no)

no

Does this paper rely on one or more datasets? (yes/no)

no

Any code required for pre-processing data is included in the appendix. (yes/partial/no).

yes

All source code required for conducting and analyzing the experiments is included in a code appendix. (yes/partial/no)

yes

All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (yes/partial/no)

yes

All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no)

yes

If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. (yes/partial/no/NA)

yes

This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. (yes/partial/no)

partial

This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. (yes/partial/no)

yes

This paper states the number of algorithm runs used to compute each reported result. (yes/no)

yes

Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. (yes/no)

no

The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). (yes/partial/no)

no

This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. (yes/partial/no/NA)

yes

This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. (yes/partial/no/NA)

partial