Simultaneous Policy and Discrete Communication Learning for Multi-Agent Cooperation

Benjamin Freed[1], Guillaume Sartoretti[1], and Howie Choset[1]

***Abstract*— Decentralized multi-agent reinforcement learning has been demonstrated to be an effective solution to large multi-agent control problems. However, agents typically can only make decisions based on local information, resulting in suboptimal performance in partially-observable settings. The addition of a communication channel overcomes this limitation by allowing agents to exchange information. Existing approaches, however, have required agent output size to scale exponentially with the number of message bits, and have been slow to converge to satisfactory policies due to the added difficulty of learning message selection. We propose an independent bitwise message policy parameterization that allows agent output size to scale linearly with information content. Additionally, we leverage aspects of the environment structure to derive a novel policy gradient estimator that is both unbiased and has a lower variance message gradient contribution than typical policy gradient estimators. We evaluate the impact of these two contributions on a collaborative multi-agent robot navigation problem, in which information must be exchanged among agents. We find that both significantly improve sample efficiency and result in improved final policies, and demonstrate the applicability of these techniques by deploying the learned policies on physical robots.**

I. INTRODUCTION

As the required complexity of robot-robot interactions in domains such as autonomous transportation and manufacturing increases, so will the need for scalable and effective multi-agent control strategies. Reinforcement learning has shown great success recently in complex multi-agent control problems [1], [2], allowing groups of agents to automatically learn policies superior to hand-crafted controllers. Many multi-agent reinforcement learning (MARL) approaches have focused on completely decentralized policies, meaning each agent selects actions independently of all other agents using only local information from its own observations [3], [4], [5], [6], [7]. Such decentralized policies offer major scalability and parallelizability advantages over centralized planning approaches, often at the cost of optimality or joint action coordination. Because actions are selected independently, the computational burden scales linearly with the team size (as opposed to exponentially, as for typical centralized planners), and action selection for each agent can occur in parallel [8].

One important drawback to decentralized MARL is that each agent's policy is conditioned only on its own observations, as opposed to all information available to the group. This lack of information sharing results in suboptimal policies and possibly poor performance at the group level [8].

Benjamin Freed, Guillaume Sartoretti, and Howie Choset are with the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA 15213, USA. {bfreed,gsartore,choset}@cs.cmu.edu

Recently, this problem has been addressed by allowing agents to selectively exchange information using a communication channel, to supply each other with the information required to make more informed local decisions, ultimately allowing superior global performance [6], [9], [4], [10]. However, this additional capability poses additional challenges, such as how to select the best information to send, how to encode it into a message, and how to interpret messages from other agents. Past approaches generally fall into one of two categories: 1) reinforced communications learning, in which messages are treated as another possible action, and standard RL approaches are applied to optimize message selection [6], [10], and 2) differentiable communications learning, in which sending and receiving agents are treated as one neural network and are trained with backpropagation. [6], [9].

We chose to focus on reinforced communications learning, as it is more general than differentiable communications learning. Reinforced communications learning naturally handles discrete, non-differentiable communication channels with unknown channel noise [4]. We feel that discrete, noisy channels are representative of many real-world robotics applications, but, to the best of our knowledge, is currently beyond the capability of differentiable communications approaches. There are, however, two significant problems with past approaches to reinforced communications learning, which we address in this work. The first is that, in past approaches, message selection has been framed as a choice of a single message from the entire set of possible messages [6], [9], [4], [10], the size of which scales exponentially with amount of message information. Practically, this means that each agent's output size must scale exponentially with the number of message bits, which quickly becomes infeasible for even modest message sizes. Secondly, adding communications learning increases the difficulty of the learning task because trajectories are now sampled from a much larger space (the joint state-action-*message* space as opposed to just the joint state-action space), meaning more episodes are required to adequately cover the space.

To address the scaling issue mentioned above, we propose an independent bitwise message policy parameterization, as described in Section III-A, from which individual message bits are independently sampled. This parameterization allows agent output sizes to scale linearly with the number of message bits, rather than exponentially. We also present a technique to combat the added difficulty of learning a message policy by deriving an improved policy gradient estimator (section III-B), which we theoretically prove yields a lower variance message gradient contribution than a typical

policy gradient estimator. We evaluate the practical impact of these contributions in a set of experiments involving a simple but illustrative partially observable multi-agent control task (section IV). Building from our prior work on distributed learning of collaborative policies [11], [12], [13], agents share policy parameters, allowing them to contribute learning updates to the same parameter set, thereby improving learning efficiency compared to heterogeneous policies. We then deploy simulation-trained policies onto a pair of autonomous ground vehicles.

II. BACKGROUND

In this section, we detail how the problem of decentralized control of a group of agents in an unknown environment is modeled as a multi-agent Markov decision process (MDP). We then extend this model to show how agents can utilize a (possibly noisy or intermittent) communication channel made available to them to exchange information.

A. Multi-agent Markov Decision Process

In the development of our theoretical results, we choose to model the environment as an MDP containing M agents, each selecting actions independently (a multi-agent MDP), with full state observability. A multi-agent MDP is defined by $(S, A, P, r, \rho_0, \gamma)$, where S is the state space, A is the joint action space, consisting of every possible combination of individual agents' actions, P denotes the state transition probability function, r denotes the reward function, ρ_0 denotes the initial state distribution, and $\gamma \in (0, 1]$ denotes a discount factor. At each timestep, agents select individual actions according to independent state-conditioned policies, denoted $\pi^{(1)}(a_t^{(1)}|s_t;\theta_1), ..., \pi^{(M)}(a_t^{(M)}|s_t;\theta_M)$, where θ_i signifies the parameters for the ith agent. We wish to find the policy parameters $\theta = \{\theta_1, ...\theta_M\}$ that maximize expected cumulative reward, $J = \mathbb{E}_{p(\tau;\theta)}[r(\tau)] = \mathbb{E}_{p(\tau;\theta)}\left[\sum_{t=0}^{T} \gamma^t R_t\right]$, where $\tau = \{s_0, a_0, ..., s_T, a_T\}$ denotes the trajectory (all states encountered and action taken during an episode, where an episode terminates when a terminal state is reached), $s_0 \sim \rho_0$, $a_t \sim \prod_{j=1}^{M} \pi^{(j)}(a_t^{(j)}|s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$, and R_t denotes the reward received at time t [14].

Our approach (described below for the case of full-state observability) can be generalized to partially-observable environments, where agents receive only local observations rather than the complete state. This generalization can be accomplished by conditioning agent policies on the entire sequence of local observations available to that agent. Such a decentralized partially-observable model is referred to as a decentralized partially-observable Markov decision process (dec-POMDP), and is described in more detail in [15]. We assume a dec-POMDP world model in our experiments.

B. Multi-Agent MDP with a Communication Channel

We can extend the MMDP model to contain a communication channel that agents can use to pass messages to each other. We assume that: 1) messages sent through the channel do not directly influence the state of the environment. 2) The communication channel is discrete, and has finite channel capacity. This modeling assumption is made to reflect real-world communication networks, which typically transfer digital information at a finite rate. 3) Messages sent through the channel may be corrupted with arbitrary unknown channel noise, and the set of agents in communication with a particular agent may change between timesteps. 4) Messages sent from a given agent will be received by other agents at the following time step. This assumption is introduced to model the latency in the channel, and can be generalized to arbitrary delays. The communications-enabled model we use is very similar to the dec-POMDP-com model described in [7], however we do not assume an explicit cost of communication, and instead assume this cost is incorporated into the standard reward function.

In addition to an action policy, we now allow each agent to maintain a message policy, or set of message policies, indicating the probability with which an agent will sample a particular message at a given the current observations. Each (possibly noise-corrupted) message becomes an input to an arbitrary set of other agents at the next timestep. For simplicity, we generally consider that each agent maintains one message policy, denoted $P_m^{(j)}$ for the jth agent, and sampled messages are sent to at most one other agent. We show in the supplementary material how this may be generalized to situations in which multiple message policies are maintained, and each agent sends messages to multiple other agents. At each timestep t for each agent j, we sample an action $a_t^{(j)} \sim \pi^{(j)}(\cdot|s_t, m_t^{\text{rec},j}; \theta_j)$ and a message $m_t^{(j)} \sim P_m^{(j)}(\cdot|s_t, a_t^{(j)}, m_t^{\text{rec},j}; \theta_j)$, where $m_t^{\text{rec},j}$ denotes the messages received by agent j at time t.

Denoting the joint action and joint message respectively to be the set of all actions and messages selected at a given time step, we can express the joint action policy as $\pi(a_t|s_t, m_t^{\text{rec}}; \theta) = \prod_{i=1}^{M} \pi^{(i)}\left(a_t^{(i)}|s_t, m_t^{\text{rec},i}; \theta_i\right)$, and the joint message policy as $P_m(m_t|s_t, a_t, m_t^{\text{rec}}; \theta) = \prod_{i=1}^{M} P_m^{(i)}\left(m_t^{(i)}|s_t, a_t^{(i)}, m_t^{\text{rec},i}; \theta_i\right)$. As in the case without messages, the objective function is the expected cumulative episode rewards, however now we take the expectation across trajectories that contain messages, i.e. $\tau = (a_0, s_0, m_0, ..., .a_T, s_T, m_T)$.

III. THEORY

We detail our approach to simultaneously learning multi-agent behavioral and communication policies, including our independent bitwise message policy, and an improved policy gradient estimator. We would like to emphasize that while we assume for simplicity that each agent maintains one message policy, and samples exactly one message at each timestep that is sent to at most one other agent throughout this section, we detail in the supplementary material how our approach can be generalized to the case in which each agent maintains multiple message policies, and sends to multiple different agents. This generalization allows our approach to be applied to problems with any number of agents and any communication topology. In this work, we do not address the issue of how each agent selects recipients for its messages,

and instead assume that some mechanism for determining message recipients is already in place, either as part of the environment or as part of the agents' action spaces.

A. Independent Bitwise Policy Parameterization

Existing approaches to reinforcement-learned communication frame message selection as a choice of a single message from the set of possible messages M. In the Q-learning approach taken by [6], a separate Q-value for each message in M is learned, and at each timestep the message with the highest Q value is selected. In the policy-gradient approach taken by [10], messages are selected from a learned message policy distribution over M. Both approaches require each agent to have $|A| + |M|$ outputs, where $|A|$ denotes an agent's action-space size. Because $|M|$ scales exponentially with message size (measured in bits), such approaches are not feasible for even a modest message sizes. For this reason, we parameterize our message policy as a set of independent probability distributions over each message bit. This allows the size of each agent's output to scale linearly with message size, because only one additional output is required for each additional message bit. One potential drawback to this approach is that because each bit is sampled independently, it does not capture the same level of dependency between message bits as a fully joint message distribution would. However, because a bitwise factorizable stochastic message policy can represent any deterministic message policy, if one assumes the optimal communication policy is deterministic (which is likely the case, since a deterministic message policy can be proven to allow agents to communicate more information compared to a non-deterministic one), such a bitwise message policy is capable of representing the optimal policy. In other words, inter-bit dependencies are not necessary in cases where the message policy converges to determinism. In addition to the scalability advantage the independent bitwise representation, in practice we observe a large increase in learning efficiency compared to learning a full distribution across M, probably owing to the drastically reduced output dimensionality (fig. 3).

B. Policy gradient for MARL with Communication

In this section, we derive a policy gradient estimator for MARL with communication. Additionally, we show how one can take advantage of the property that messages only influence the world via the actions of the recipient agent at the next time step to derive a more accurate baseline and therefore reduce the variance of our policy gradient estimator, compared to what would be possible if messages were assumed to have arbitrary influence on the environment.

For convenience, we introduce the following notion: The augmented state, representing the concatenation of the state at time t and all messages received at time t, will be denoted $\bar{s}_t = [s_t, m_t^{\text{rec}}]$. The augmented state visitation frequency under joint policy π and joint message distribution P_m will be denoted $\rho_{\bar{s}}(\bar{s}) = \sum_{t=0}^{T} \gamma^t p(\bar{s}_t = \bar{s}|\pi, P_m)$. For convenience, $\nabla_{\theta_j} \log \pi^{(j)}(a_t|s_t, m_t^{\text{rec}}; \theta)$ will be abbreviated to $\nabla \log \pi$, and $\nabla_{\theta_j} \log P_m^{(j)}(m_t|s_t, a_t, m_t^{\text{rec}}; \theta)$ will be abbreviated to $\nabla \log P_m$.

Applying the policy gradient theorem [16], and treating the joint action-message distribution $P(a_t, m_t|s_t, m_t; \theta)$ as a traditional policy, we have that the gradient of the objective function with respect to θ_j is:

$$\begin{aligned}\nabla_{\theta_j} J &= \mathbb{E}_{\rho_{\bar{s}}}\left[\nabla_{\theta_j} \log p(a_t, m_t|s_t, m_t^{\text{rec}}; \theta) Q^{\pi}(s_t, a_t, m_t)\right], \\ &= \mathbb{E}_{\rho_{\bar{s}}}\Big[\big(\nabla_{\theta_j} \log \pi(a_t|s_t, m_t^{\text{rec}}; \theta) + \\ &\quad \nabla_{\theta_j} \log P_m(m_t|s_t, a_t, m_t^{\text{rec}}; \theta)\big) Q^{\pi}(s_t, a_t, m_t)\Big], \\ &= \mathbb{E}_{\rho_{\bar{s}}}\Big[\big(\nabla \log \pi^{(j)} + \nabla \log P_m^{(j)}\big) Q^{\pi}(s_t, a_t, m_t)\Big].\end{aligned} \tag{1}$$

where $Q^{\pi}(s_t, a_t, m_t) = \mathbb{E}_{\tau}\left[\sum_{k=t}^{T} \gamma^{k-t} R_t|s_t, a_t, m_t\right]$ is a natural extension of the typical Q-function that now depends additionally on the selected messages at time t.

This gradient can be approximated by the following unbiased gradient estimator:

$$g_j = \left(\nabla \log \pi^{(j)} + \nabla \log P_m^{(j)}\right) \hat{Q}_t(s_t, a_t, m_t), \tag{2}$$

where $\hat{Q}_t(s_t, a_t, m_t) = \sum_{k=t}^{T} \gamma^{k-t} R_t|s_t, a_t, m_t$. The variance of this estimator can be further reduced while keeping it unbiased by subtracting a baseline from the sum of future rewards [16], yielding the following gradient estimator (where the arguments of $\hat{Q}_t$ are omitted for brevity):

$$g'_j = \nabla \log \pi^{(j)} \left(\hat{Q}_t - b_t^{(a,j)}\right) + \nabla \log P_m^{(j)} \left(\hat{Q}_t - b_t^{(m,j)}\right). \tag{3}$$

In general, to achieve an unbiased gradient estimator, the action baseline $b_t^{(a,j)}$ should not depend on the action, i.e. should only depend on the state . However, recent work showed that if the policy is factorizable, as is the case here because each agent chooses actions (and messages) conditionally independently from all other agents, the baseline for each policy factor (in this case corresponding to each agent's policy) can be conditioned on the samples drawn from all other policy factors [17]. As a result, each agent can have its own action baseline, and that baseline can depend on all other agents' actions, as well as all other agents' messages. In other words, $b_t^{(a,j)}$ can be of the form $b_t^{(a,j)} = b(s_t, a_t^{\neq j}, m_t^{\neq j})$, where $a_t^{\neq j}$ and $m_t^{\neq j}$ are the set of all actions and messages, respectively, sampled at timestep t, excluding those of agent j.

Taking advantage of similar conditional-independence properties present in our communications-enable model, we demonstrate that specifically in the case of multi-agent RL with communication, it is possible to condition the message baseline on much more information, while keeping the resulting gradient estimator unbiased. Specifically, we show that, because all actions, messages from other agents, and rewards at time t are conditionally independent of the message $m_t^{(j)}$, as well as all actions and messages at time $t+1$ except $a_{t+1}^{(x)}$ and $m_{t+1}^{(x)}$, where agent x is the recipient of $m_t^{(j)}$, the message baseline for agent j at time t can take the form $b_t^{(m,j)} = b(s_t, a_t, m_t^{\neq j}, R_t, s_{t+1}, a_{t+1}^{\neq x}, m_{t+1}^{\neq x})$, while keeping the gradient estimator unbiased (see supplementary material). This baseline is equivalent to one of the

form $b_t^{(m,j)} = b(R_t, s_{t+1}, a_{t+1}^{\neq x}, m_{t+1}^{\neq x})$, because the future trajectory is independent of s_t, a_t, $m_t^{\neq j}$ given s_{t+1}, $a_{t+1}^{\neq x}$, and $m_{t+1}^{\neq x}$. Intuitively speaking, the utility of conditioning baselines on additional information is that by doing so we may hope to further reduce the variance of the gradient estimator, because the baseline can be made to absorb more of the fluctuations inherent to future rewards. This concept is explored further in Section III-C.

C. Learned Value Function Baseline

In practice, it is common to use a learned value function as a baseline. Such algorithms are known as actor-critic [18]. Typically, such baselines are functions of only the current state, i.e. of the form $b(s_t) = \mathbb{E}_{a_t}\left[Q^\pi(s_t, a_t)\right] = V(s)$. Wu et al. propose the following state-action-dependent baseline for agent j (for an MDP that contains no messages) [17]: $b\left(s, a^{\neq j}\right) = \mathbb{E}_{a_t^{(j)}}[Q^\pi(s_t, a_t)]$.

We extend this approach to the case with messages, and use the following for our action baseline:

$$\begin{aligned} b_t^{a,j} &= \mathbb{E}_\tau\left[\sum_{k=t}^T \gamma^{k-t} R_k \middle| s_t, a_t^{\neq j}, m_t^{\neq j}\right] \\ &= \mathbb{E}_{a_t^{(j)}, m_t^{(j)}}\left[Q^\pi(s_t, a_t, m_t)\right]. \end{aligned} \tag{4}$$

In a similar vein, we propose the following message baseline:

$$\begin{aligned} b_t^{m,j} &= \mathbb{E}_\tau\left[\sum_{k=t}^T \gamma^{k-t} R_k \middle| s_t, a_t, m_t, R_t, s_{t+1}, a_{t+1}^{\neq x}, m_{t+1}^{\neq x}\right] \\ &= R_t + \gamma \mathbb{E}_{a_{t+1}^{(x)}, m_{t+1}^{(x)}}\left[Q^\pi(s_{t+1}, a_{t+1}, m_{t+1})\right]. \end{aligned} \tag{5}$$

where agent x receives agent i's message from time t.

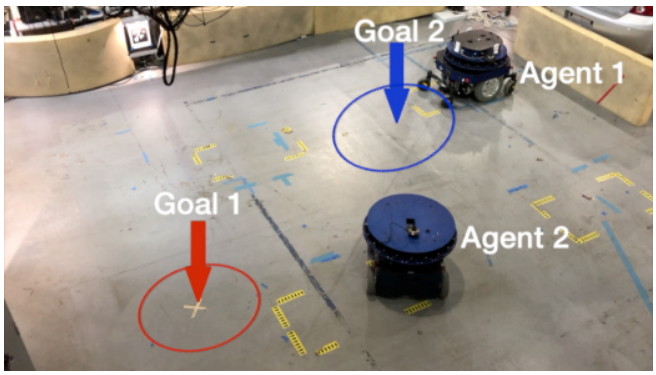
Defining the action and message advantage estimates for agent j at time t as $\hat{A}_t^{a,j} = \hat{Q}_t - \mathbb{E}_{a_t^{(j)}, m_t^{(j)}}\left[Q^\pi(s_t, a_t, m_t)\right]$ and $\hat{A}_t^{m,j} = \hat{Q}_{t+1} - \mathbb{E}_{a_{t+1}^{(x)}, m_{t+1}^{(x)}}\left[Q^\pi(s_{t+1}, a_{t+1}, m_{t+1})\right]$, we see that the message advantage for agent j at time t is equivalent to the action advantage for agent x at the next time step, i.e. $\hat{A}_t^{m,j} = \hat{A}_{t+1}^{a,x}$.

Expressed in terms of advantage estimates, the gradient estimator then becomes:

$$g_j' = \nabla \log \pi^{(j)} \hat{A}_t^{a,j} + \nabla \log P_m^{(j)} \gamma \hat{A}_{t+1}^{a,x}. \tag{6}$$

The advantage of an action is a measure of how much additional reward was achieved after its selection compared to the expected future reward from following the current policy from the current state. Because a message only influences the future reward via the recipient agent's action at the next time step, it is intuitive that in evaluating a particular message, we need only consider the advantage of the influenced action, discounted by 1 time step. We analytically prove in the supplementary material the contribution of the message policy to this gradient estimator is lower variance than one in which the same advantage estimate is used for both the action and the message. Additionally, we show how such an approach can be generalized to the case in which agents send messages to multiple different recipient agents, possibly sampled from multiple distinct message policies.

In practice, we found it feasible to use a neural network to approximate these value functions. However, rather than learning a complete Q-function and taking the expectation with respect to $a_t^{(j)}$ and $m_t^{(j)}$, we simply learned an analog of a traditional value function $V^{\pi,j}(s_t, a_t^{\neq j}, m_t^{\neq j}) = \mathbb{E}_\tau\left[\sum_{k=t}^T \gamma^{(k-t)} R_k | s_t, a_t^{\neq j}, m_t^{\neq j}\right]$ by regressing the discounted sum of future rewards obtained in a rollout, $\hat{Q}_t(s_t, a_t, m_t)$, against s_t, $a_t^{\neq j}$, and $m_t^{\neq j}$.



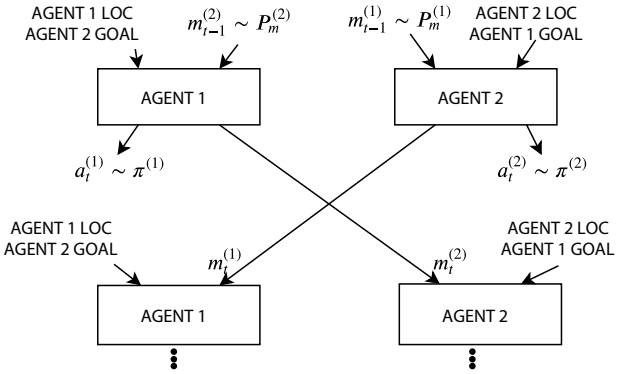


Fig. 1. Experimental setup. Top: two agents navigate in a grid world, and observe their own location and the others agent's goal. Agents do not observe their own goal, and must therefore communicate it to each other. Bottom: At each timestep, agents output an action and message policy, based on their environmental observation and message input. An action and message are sampled from the action and message policies, respectively, and each agent's message becomes an input to the other at the next timestep.

IV. EXPERIMENTS

We demonstrate the applicability of our approach to real robotic systems using a simple but illustrative experiment, in which robots must jointly learn behavioral and communication policies. In our experiment, two robots in a grid world are each tasked with navigating to a randomly selected goal cell, which remains static throughout an episode. Robots are only able to directly observe their location, and their neighbor's goal. Therefore, the robots must exchange information in order to arrive at their goals with probability higher than what random exploration would allow. At each timestep, agents select one out of five possible actions from their stochastic action policy, corresponding to moving in the four cardinal directions, or remaining stationary. Additionally, agents both select a message from their stochastic message policies, which becomes the input to the other agent at the next timestep (fig 1). Robots have no persistent state, mean-

ing they cannot remember past observations or messages, thus necessitating communication at each timestep.

To highlight the benefits of a fully reactive, online behavioral and communication policy, we run experiments in which trained policies are deployed on physical robots. These robots have actuation noise not present in our simulation environment, such as delays in action execution; however, the policy is able to deal with this noise without the additional computational cost of replanning trajectories.

We use asynchronous actor-critic to train our agents in simulation, and then deploy their learned policies to perform real-time control of actual robots. For generality, in section 3 we assumed messages could depend on selected actions; however in this experiment we selected messages independently of actions. We compare the following four techniques:

1) **Standard policy gradient, one-hot messages**: the same advantage estimates are used for messages and actions, and one-hot messages are sampled from a complete distribution over the message space.
2) **Standard policy gradient, bitstring messages**: the same advantage estimates are used for messages and actions, but the message policy is parameterized as a set of independent distributions over individual message bits.
3) **Improved policy gradient, one-hot messages**: separate advantage estimates are used for actions and messages, as detailed in section 3.3, and one-hot messages are sampled from a complete distribution over the message space.
4) **Improved policy gradient, bitstring messages**: our full approach, incorporating both separate action/message advantage estimates, and a message distribution parameterized as a set of independent distributions over individual message bits.

Below we detail the specifics of the reward function used, the neural network structure, the specifics of the learning algorithms employed for each of the four cases, and the observed results.

A. Reward Function

To ensure a collaborative joint policy is learned, we use a shared reward function, in which agents are rewarded according to the sum of individual rewards. We use a shaped reward function, as we found that it outperformed a sparse reward function. Individual rewards are computed in the following manner:

- A penalty of -0.3 is imposed for every time step the agent is not on goal
- A penalty of -0.1 is imposed if agents decide not to move when they are not on goal, or select an invalid action.
- A reward of 0.2 is given each timestep agents move closer to their goal.
- A reward of 0 is given for every timestep that finishes with the agent on goal.

The episode terminates either at the first timestep in which both agents are resting on their goals or once the maximum episode length of 256 timesteps is reached, at which point no additional reward is given. The reward function is designed so that it is guaranteed to be negative at every time step; thus, the optimal strategy is for agents to find their goals and end the episode as quickly as possible.

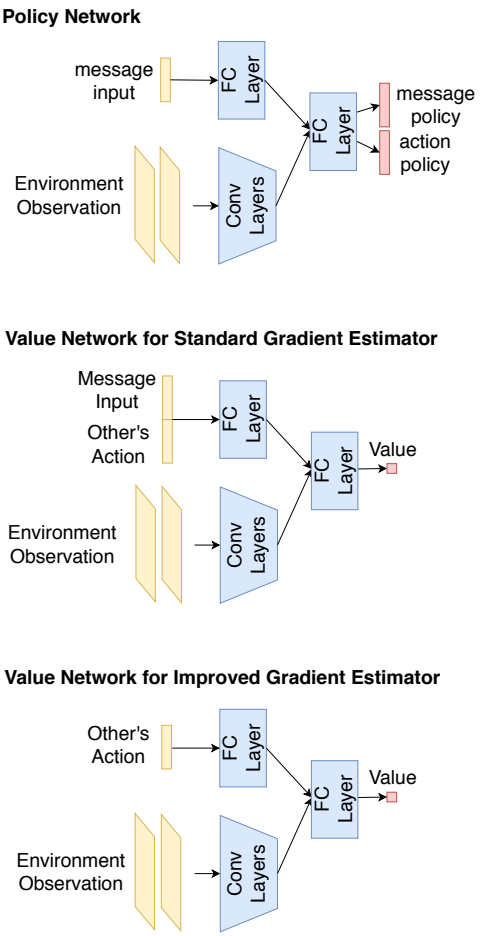


Fig. 2. Policy and value network architectures. Each agent maintains a policy and value network. The policy network (top) is responsible for mapping an environmental observation and the neighboring agent's message to its action and message policies. The value network is responsible for estimating the expected discounted future rewards given a state observation and other inputs. There are two variants of the value network, used as a baseline function in either the standard gradient estimator (center) or improved gradient estimator (bottom). All network architectures process environmental observations using a stack of convolutional layers, and process the message input (and/or neighbor's action) using a separate fully-connected layer, which is then fused with the processed environmental observation and passed through two more fully connected layers.

B. Neural Network Structure

Both agents maintain identical copies of two separate neural networks, the policy network and value network (fig. 2). The policy network is responsible for outputting the action and message policy distributions. The value network is responsible for outputting a value estimate, which is used as a baseline in the policy gradient estimates. Further details on these networks are provided below.

1) Policy Network: The policy network takes two separate inputs: an environmental observation, and the message from the other agent at the last timestep. The environmental

observation consists of two 10x10 one-hot matrices, indicating its position in the grid world and its neighbor's goal, and is processed by two convolutional layers into an intermediate representation. The message input is sent through one fully-connected layer before being concatenated with the intermediate representation, which is then sent through two additional fully-connected layers. The policy network outputs: 1) the policy distribution, which is a normalized vector of length 5, indicating the probability of selecting each of the 5 actions, and 2) the message distribution, which is either a 128-length normalize vector, indicating the probability of each possible message, or a vector of length 7 containing elements that range between 0 and 1 and indicate the probability that the corresponding message bit is a 1. The two output sizes are chosen such that both result in the same number of possible messages, and that number is sufficient to uniquely identify one grid cell out of the 100 grid cells in the environment.

The policy network parameters are updated according to one of the following two gradient estimators, depending on whether standard or improved gradient estimators are being used:

Standard gradient estimator for the parameters of agent j:

$$g_j = \sum_{t=0}^{T} \Big[\Big(\nabla \log \pi^{(j)} + \nabla \log P_m^{(j)} \Big) \Big(\hat{Q}_t - V_1(s_t, a_t^{\neq j}, m_{t-1}^{\neq j}) \Big) \Big] \tag{7}$$

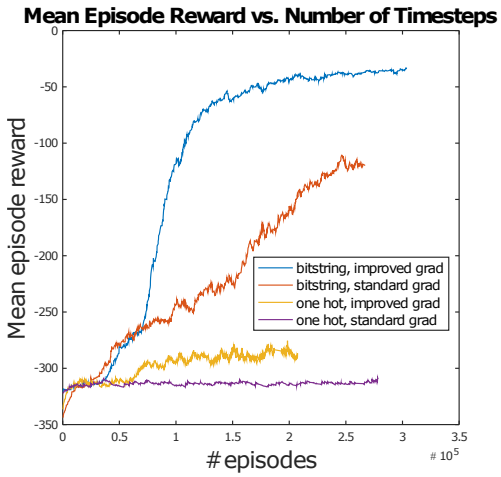
Improved gradient estimator for the parameters of agent j:

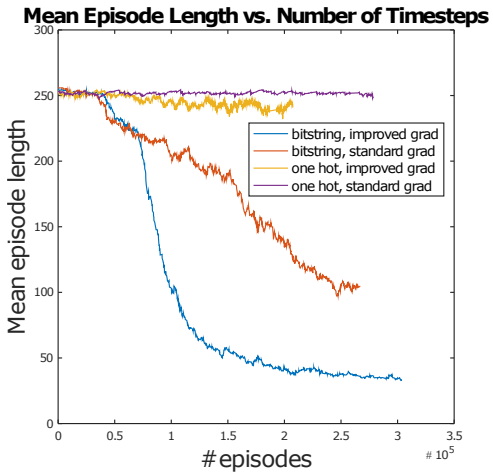
$$g'_j = \sum_{t=0}^{T} \Big[\nabla \log \pi^{(j)} \Big(\hat{Q}_t - V_2(s_t, a_t^{\neq j}) \Big) + \nabla \log P_m^{(j)} \gamma \Big(\hat{Q}_{t+1} - V_2(s_{t+1}, a_{t+1}^{\neq x}) \Big) \Big] \tag{8}$$

where $\hat{Q}_t$ denotes the discounted sum of future rewards starting at time t (i.e. $\hat{Q}_t = \sum_{k=t}^{T} \gamma^{(k-t)} R_t$), V_1 and V_2 respectively denote the value approximations according to the first and second value network variants, as described below, and x denotes the recipient of agent j's message from time t (which in this case is simply the other agent).

2) Value Network: The value network we employ to estimate the value function is similar in architecture to the policy network, but takes different inputs. We use two variants of the value network, one for obtaining standard advantage estimates, and one for obtaining improved advantage estimates. Because the value is not used during execution time, we chose to give both value variants full state observability. The variant we use for obtaining standard advantage estimates takes as input the environmental observation, the message from the other agent at the previous time step, and the other agent's action from the current time step. The value variant used for improved advantage estimates takes as input the environment state and the action from the other agent. This network cannot take as input the message from the other agent at the last time step, since the baseline it generates for the current action becomes the message advantage for the other agent at the last time step, (as explained in section 3.3) but this baseline cannot in turn be conditioned on the other agent's message from the last time step. Although theoretically justified, neither value variant takes the current message as input; we found that this hindered performance, probably because it is not important to estimating the value, but makes the input space larger.

The value network was trained to minimize mean squared error between its value estimate and the Q estimate $\hat{Q}_t$ for each timestep t in the episode.





Approach	Time required to reach goals
Improved policy gradient, bitstring message	43 sec
Standard policy gradient, bitstring message	145 sec
Improved policy gradient, one-hot message	N/A
Standard policy gradient, on-hot message	N/A

Fig. 3. Experimental Results for simulation (left) and actual robots (right). In simulation, we observe learning efficiency improvements both from utilizing an independent bitwise message policy, and our improved policy gradient estimator, with the best results occurring when the two techniques are combined. The policy utilizing a bitwise message policy and improved gradient estimates also allowed actual robots to reach their goals in much less time than the other 3 techniques. Neither of the policies trained with one-hot messages allowed robots to reach their goals.

C. Results

The quantitative performance of the described techniques was evaluated in simulation. Of the four techniques tested, the improved policy gradient/bitstring message approach achieved the highest sample efficiency and ultimate per-

formance. The standard policy gradient/bitstring message approach yielded the second-best results. Both approaches utilizing one-hot messages performed poorly. The version using standard policy gradient estimates failed to improve its policy beyond what is possible using a brute-force search of the space, indicating the agents either failed to communicate useful information, or failed to utilize that information. The version using the improved policy gradient was able to achieve performance superior to a brute-force search, indicating that agents did learn to communicate (fig. 3).

Trials on physical robots corroborated the simulation results. The improved policy gradient/bitstring message approach allowed both robots to find their goals rapidly. The standard policy gradient/bitstring message allowed the robots to find their goals, albeit via a much less efficient route. Finally, neither of the policies that used one-hot messages allowed robots to find their goals, and robots moved around the space in a chaotic manner (fig. 3).

V. CONCLUSIONS

In this paper, we presented two techniques for addressing the current shortcomings in existing approaches to MARL with communication, and have demonstrated their effectiveness on a partially-observable multi-agent problem, on both physical and simulated robots. We presented a novel bitwise message policy parameterization that allows agent output size to scale linearly with message information content, making it applicable to situations in which large amounts of information must be exchanged between agents. In experiments, we found that such a bitwise policy parameterization results in a large improvement in learning efficiency, likely because it allows for much smaller policy network output spaces. Additionally, we described an unbiased policy gradient estimator that has a lower variance message gradient contribution than existing approaches, yielding a large increase in learning efficiency. The techniques presented in this paper can in principle be generalized to arbitrary team sizes and network topologies, as well as problems with unknown channel noise.

Ongoing work is examining the generalizability of these techniques by applying them to complex tasks involving large groups of coordinating agents. We are also investigating the effects of channel noise and changing network topologies on group performance, as well as examining schemes for allowing agents more selectively address recipient agents. Finally, to further improve sample efficiency, we are developing techniques for overcoming limitations present in differentiable communications learning.

REFERENCES

[1] OpenAI, "Openai five," https://blog.openai.com/openai-five/, 2018.

[2] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, *et al.*, "Human-level performance in 3d multiplayer games with population-based reinforcement learning," *Science*, vol. 364, no. 6443, pp. 859–865, 2019.

[3] J. K. Gupta, M. Egorov, and M. J. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *AAMAS Workshops*, 2017.

[4] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[5] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," 2018.

[6] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," pp. 2137–2145, 2016.

[7] F. S. Melo and M. Veloso, "Heuristic planning for decentralized MDPs with sparse interactions," in *DARS*, 2013, pp. 329–343.

[8] L. Busoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," *Innovations in Multi-Agent Systems and Applications-1*, vol. 310, pp. 183–221, 2010. [Online]. Available: http://www.dcsc.tudelft.nl/~bdeschutter/pub/rep/10_003.pdf

[9] S. Sukhbaatar, R. Fergus, *et al.*, "Learning multiagent communication with backpropagation," pp. 2244–2252, 2016.

[10] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," 2018.

[11] G. Sartoretti, W. Paivine, Y. Shi, Y. Wu, and H. Choset, "Distributed learning of decentralized control policies for articulated mobile robots," *IEEE Transactions on Robotics*, pp. 1–14, 2019.

[12] G. Sartoretti, J. Kerr, Y. Shi, G. Wagner, T. K. S. Kumar, S. Koenig, and H. Choset, "PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2378–2385, 2019.

[13] G. Sartoretti, Y. Wu, W. Paivine, T. K. S. Kumar, S. Koenig, and H. Choset, "Distributed reinforcement learning for multi-robot decentralized collective construction," in *DARS 2018 - International Symposium on Distributed Autonomous Robotic Systems*, 2018, pp. 35–49.

[14] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*. Morgan Kaufmann Publishers Inc., 1996, pp. 195–210.

[15] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Mathematics of operations research*, vol. 27, no. 4, pp. 819–840, 2002.

[16] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *NIPS*, 1999.

[17] C. Wu, A. Rajeswaran, Y. Duan, V. Kumar, A. M. Bayen, S. Kakade, I. Mordatch, and P. Abbeel, "Variance reduction for policy gradient with action-dependent factorized baselines," 2018.

[18] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.