Learning from My Partner's Actions:
Roles in Decentralized Robot Teams

Dylan P. Losey*
Stanford University
dlosey@stanford.edu

Mengxi Li*
Stanford University
mengxili@stanford.edu

Jeannette Bohg
Stanford University
bohg@stanford.edu

Dorsa Sadigh
Stanford University
dorsa@cs.stanford.edu

Abstract: When teams of robots collaborate to complete a task, communication is often necessary. Like humans, robot teammates should *implicitly* communicate through their *actions*: but interpreting our partner's actions is typically difficult, since a given action may have many different underlying reasons. Here we propose an alternate approach: instead of not being able to infer whether an action is due to exploitation, information giving, or information gathering, we define separate *roles* for each agent. Because each role defines a distinct reason for acting (e.g., only exploit), teammates can now correctly interpret the meaning behind their partner's actions. Our results suggest that leveraging and alternating roles leads to performance comparable to teams that explicitly exchange messages.

Keywords: Multi-Agent Systems, Decentralized Control, Robot Learning

1 Introduction

When teams of robots are deployed in our homes, warehouses, and roads, often these robots must collaborate to accomplish their task. Imagine two robots working together to move a heavy table across a room (see Fig. 1). Due to occlusions, each agent can only see some of the obstacles within this room. Thus, the robots need to *communicate* to inform their partner about the obstacles they see. One option is for the robots to explicitly communicate by directly sending and receiving messages: i.e., we could tell our teammate where the obstacles are. But humans utilize more than just explicit communication—we also implicitly communicate through our actions. For example, if our partner guides the table away from our intended trajectory, we might infer that we were moving towards an obstacle, and that there is a better path to follow. Collaborative robot teams—like humans—should also leverage the information contained within their partner's actions to learn about the world.

Unfortunately, interpreting the meaning behind an action is hard. Robots can take actions for *many different reasons*: to exploit what they know, actively give information to their partner, or actively gather information about the environment. So when we observe our partner applying some force to the table, what (if anything) should we learn from that action? And how do we select actions that our partner can also interpret? In this paper, we show that assigning roles alleviates these challenges:

> *Teams of robots can correctly interpret and learn from each other's actions when the team is* separated *into* roles, *and each role provides the robots with a distinct reason for acting.*

Returning to our example, imagine that our partner's role is to exploit their current information: they move the table towards the goal while avoiding the obstacles that they can observe. If we know this role, we can now interpret their actions: when our partner applies an unexpected force, it must be because of some obstacle that we did not see. Hence, assigning roles enables us to *learn* from our partner's actions and update our estimate of the system state *naturally*, without requiring additional, explicit communication. In this paper, we make the following contributions:

Roles in Two-Player Teams. We focus on decentralized two-player teams where each agent sees part of the current state, and together the agents observe the entire state. We show that—without
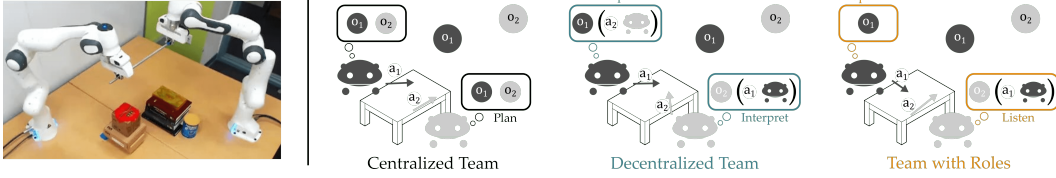

Figure 1: (Left) Our problem setting, where two robots must work together to complete a task. Each robot observes different parts of the true system state. (Right) Unlike **centralized teams**, **decentralized teammates** should implicitly communicate through actions: here each agent sees one obstacle, and tries to infer where the other obstacle is based on their partner's actions. Interpreting our partner's actions is hard: there are many reasons why an agent might choose an action. We introduce speaker and listener **roles** so that each agent has a distinct reason for acting, enabling the robots to learn from and implicitly communicate through their actions.

roles—the agent policies are interdependent, and interpreting the actions of our partner leads to infinite recursion (what do you think I think you think, etc.). To remove this interdependence and reach interpretable actions, we introduce two classes of policies: a *speaker* role, where agents exploit what they know, and a *listener* role, where agents learn by modeling their partner as a speaker.

Mimicking Explicit Communication. We explore how roles can be used to make our decentralized team behave like a *centralized* team that communicates explicitly. We find that decentralized teammates which alternate between roles can match the centralized team, but if the agents always maintain the same roles, the team may become unstable. We also reveal that speakers trade-off between stochasticity and communication: to improve overall team performance, speakers should choose more deterministic actions than the centralized policy to clearly convey their observations.

Comparing Implicit to Explicit. We implement roles both in a simulated game and a two-robot manipulation task. Our simulations compare implicitly communicating through actions to explicitly communicating by sending messages, and demonstrate that—when robots leverage roles—implicit communication is almost as effective as explicit communication. In robot experiments, teams that alternated roles successfully communicate their observations and collaborate to avoid obstacles.

Overall, this work is a step towards learning from our partner's actions in decentralized robot teams.

2 Related Work

Multi-Agent Teams. Teams of robots have performed tasks such as localization [1], navigation [2], and manipulation [3, 4]. While many works rely on centralized coordination [5], decentralized multi-agent teams are receiving increasing attention [6, 2, 7, 8]. Within *decentralized control*, the problem is reformulated as a decentralized partially-observable Markov decision process (Dec-POMDP) [9, 10] or coordinator-POMDP [11]. In practice, solving these optimization problems is frequently intractable: related works instead rely on high-level abstractions [12], sparse interactions between agents [13], specific problem instances [14], or control approximations [6]. Alternatively, with multi-agent *reinforcement learning* (MARL), agents can learn decentralized policies from trial-and-error [15]. Today's MARL approaches often leverage actor-critic methods to scale to high dimension state-action spaces [16, 17, 18]; however, these methods require offline, centralized training, and must deal with the credit assignment problem (i.e., who is contributing to the team's success?).

Communication. Across both control and learning approaches, communication between agents is key to effective collaboration. The protocol that the agents use to communicate can be either based on pre-defined triggers [19, 20] or learned from training data [21, 22, 23]. But the method that the agents use to communicate is generally *explicit*: the agents have a separate channel with which they directly broadcast and receive messages from their teammates [24]. At the other end of the spectrum, [25, 26, 27] study cooperative robots that do *not* attempt to exchange information. Unlike these recent works that either have access to explicit communication or omit communication entirely, we will focus on leveraging *implicit* communication through actions.

Roles in Human Teams. Our research is inspired by human-human teams, where different roles naturally emerge in collaborative tasks [28, 29]. For example, when two humans are working together to physically manipulate an object, the agents can identify and assume complementary roles based on their partner's force feedback alone [30]. Roles have also been applied to human-robot interaction: here the robot dynamically adjusts its level of autonomy (i.e., becomes a leader or follower) in response to the force feedback from the human partner [31, 32, 33]. We will *extend* these ideas to robot-robot teams, where we believe that roles can similarly improve collaboration.

3 Interpreting Actions via Roles

Consider a decentralized, two-agent team where the agents share a common objective. Each agent observes part of the system state. To make good decisions, both agents need an accurate estimate of the entire state; for instance, we need to know whether there is an obstacle behind us, if there is free space to our right, etc. While we know the aspects of the state that we directly observe, how can we estimate the parts of the system state that our partner sees? One natural solution is to learn about the environment based on the *implicit* communication contained within our partner's *actions*. In this section, we show that correctly interpreting the meaning behind our partner's actions is challenging when both agents try to learn from their partner and exploit what they observe at the same time. Returning to the table moving example: when our partner applies a force, is this because of what they have learned from our own actions, because of an obstacle behind us, or some combination of both? To correctly infer the unobserved state, each agent must reason over their partner's behavior, and this behavior may in turn depend on the first agent's actions. Accordingly, we here introduce *speaker* and *listener* roles to remove this interdependency: the speaker implicitly communicates relevant parts of the state that they observe to the listener, who learns a more accurate state estimate.

Two-Agent Team. We formulate our two-player team as a decentralized Markov decision process (Dec-MDP) [9]. Let the state space be $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2$ and let the action space be $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$, where $\mathcal{S}_i$ and $\mathcal{A}_i$ are the state and action space for agent i. The team has dynamics $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ and receives reward $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. At the current timestep t, agent i observes its own state component s_i^t, and collectively the team observes $s^t = (s_1^t, s_2^t)$. We therefore have that the state is *jointly* fully observable: s^t is known given the current observations of both agents, s_1^t and s_2^t. When making decisions, agent i has access to its history of observations, $s_i^{0:t} = (s_i^0, s_i^1, \ldots, s_i^t)$, as well as the history of actions taken by both agents, $a^{0:t-1}$. For simplicity, we assume that each agent observes its partner's current action selection (our results still hold if they observe the previous action). Hence, the agents have policies of the form $\pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t)$ and $\pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t)$.

Mimicking Centralized Teams. One approach to control decentralized teams is solving this Dec-MDP; however, the problem is NEXP-complete [10], and often intractable for continuous state and action spaces. We study a different approach: we find policies for both decentralized agents to collectively *mimic* the behavior of a *centralized* team [34, 23]. Imagine that—when moving a table—both teammates know exactly what their partner sees; when we explicitly communicate our observations, we can solve the problem together, and collaborate perfectly to carry the table. We will treat this centralized policy that uses explicit communication as the *gold standard* for our decentralized team, while recognizing that the decentralized team only has access to implicit communication, and may not be able to completely match the centralized team. Importantly, the optimal centralized policy is the solution to an MDP, and can be tractable computed offline (P-complete) [10].

Interdependent Policies. Let the centralized policies be $\pi_1^*(a_1^t \mid s_1^t, s_2^t)$ and $\pi_2^*(a_2^t \mid s_1^t, s_2^t)$. When both decentralized agents choose their actions to best mimic this centralized behavior, we reach:

$$\pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t) \propto \sum_{s_2^{0:t}} \pi_1^*(a_1^t \mid s_1^t, s_2^t) \cdot \pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \cdot P(s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1})$$

$$\pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \propto \sum_{s_1^{0:t}} \pi_2^*(a_2^t \mid s_1^t, s_2^t) \cdot \pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t) \cdot P(s_1^{0:t} \mid s_2^{0:t}, a^{0:t-1}) \tag{1}$$

See the appendix for our complete derivation. We cannot solve Eqn. (1) as is because the policies are interdependent: π_1 and π_2 both appear in each other's policy, so that solving for π_1 requires solving π_2 for all possible $s_2^{0:t}$, which requires an inner loop that solves for π_1 over all possible $s_1^{0:t}$, and so on. Going back to our table example: imagine that our partner observes whether there is an obstacle behind us, and we want to infer the likelihood of this obstacle from their actions. This is easy when our partner's actions are only based on this obstacle—but if our partner's behavior is also a response to our own actions, how to we know which aspects of our partner's behavior to learn from? To break this interdependence and recover interpretable actions, we separate our team into two roles: an agent that exploits what it observes (the *speaker*), and an agent that learns from its partner's (the *listener*).

Speaker. A speaker is an agent that makes decisions by purely exploiting their observations $s_i^{0:t}$. Let Agent 1 be the speaker; the speaker policy that best matches π_1^* is:

$$\pi_1(a_1^t \mid s_1^{0:t}) = \sum_{s_2} \pi_1^*(a_1^t \mid s_1^t, s_2^t) \cdot P(s_2^t \mid s_1^{0:t}) \tag{2}$$

More generally, any policy of the form $\pi_i(a_i^t \mid s_i^{0:t})$ is a speaker. Because speakers react to their observations, these actions convey information about the parts of the state they see to their teammates.

Listener. A listener makes decisions based on its own observations while also learning an improved state estimate from its partner's behavior. If Agent 2 is the listener, the policy that matches π_2^* is:

$$\pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \propto \sum_{s_1^{0:t}} \pi_2^*(a_2^t \mid s_1^t, s_2^t) \cdot \pi_1(a_1^t \mid s_1^{0:t}) \cdot P(s_1^{0:t} \mid s_2^{0:t}, a^{0:t-1}) \tag{3}$$

Compared to (1), now the listener models its partner as a speaker, and solving for π_1 (the speaker policy) does not depend on π_2 (the listener policy). More generally, we consider any policy of the form $\pi_i(a_i^t \mid s_i^{0:t}, a^{0:t-1}, a_j^t)$ as a listener if it interprets its teammate's actions with $\pi_j(a_j^t \mid s_j^{0:t})$.

4 Leveraging Roles Effectively

Now that we have defined roles, let us return to our table carrying example. Imagine that we are the speaker and our partner is the listener: what is the best speaker policy for us to follow? Should we always remain a speaker, or do we need to switch speaker and listener roles with our teammate? And if our decentralized team uses roles, when can we fully match the behavior of a centralized team? In this section we explore these questions, and analyze how roles operate within simplified settings.

4.1 When Can We Use Roles to Match a Centralized Team?

Roles enable implicit communication through actions. This implicit communication is typically less informative than explicitly sharing observations; but when an agent's actions can completely convey their observed state, robots can leverage roles to fully match the behavior of a centralized team:

Theorem 1. *For continuous systems, if there exist surjective functions $g_1 : \mathcal{A}_1 \to S_1$ and $g_2 : \mathcal{A}_2 \to S_2$, a decentralized team using speaker and listener roles can match a centralized team.*

Proof. The decentralized team matches the centralized team's performance by communicating with actions while rapidly alternating between speaker and listener roles. Define a_1^* as the optimal centralized action for agent 1, and let $\bar{a}_1$ be a naive action that completely conveys what agent 1 observes: $g_1(\bar{a}_1) = s_1$. We choose the speaker action to be $\bar{a}_1$ and the listener action to be $a_2^* + (a_2^* - \bar{a}_2)$, where the listener can compute a_2^* because it observes s_2 and infers s_1 from $g_1(\bar{a}_1)$. The agents change roles: agent 1 is the speaker for time $[t, t + dt)$ and agent 2 is the speaker for time $[t + dt, t + 2dt)$. Taking the limit as $dt \to 0$ (i.e., as the roles change infinitely fast), the team's action a becomes:

$$a^t = \lim_{dt \to 0} \frac{a^t + a^{t+dt}}{2} = \lim_{dt \to 0} \frac{1}{2}\left(\begin{bmatrix} \bar{a}_1^t \\ a_2^{*t} + (a_2^{*t} - \bar{a_2}^t) \end{bmatrix} + \begin{bmatrix} a_1^{*t+dt} + (a_1^{*t+dt} - \bar{a}_1^{t+dt}) \\ \bar{a}_2^{t+dt} \end{bmatrix} \right) = a^{*t}$$

and so the decentralized team's action converges to the optimal action of the centralized team. $\square$

Intuitively, consider the table example with two dynamic obstacles, one of which is observed by each teammate. When we are the speaker, we apply a force with a direction and magnitude that conveys our obstacle's position to the partner; likewise, when our partner is the speaker, their action informs us where their obstacle is. By quickly switching speaker and listener roles we can both understand the state, and match the behavior of a team that explicitly tells one another about the obstacles. Theorem 1 can also be extended to a team of N agents (see appendix for full proof).

4.2 Analyzing Roles in Linear Feedback Systems

To better understand how roles affect decentralized teams, we specifically focus on teams controlled using linear feedback. Here the centralized policy is $a^* = -K^*s$, where K^* is the desired control gain; for instance, this policy could be the solution to a linear-quadratic regulator (LQR). Assuming our decentralized team is similarly controlled with $a = -Ks$, we first determine whether alternating speaker and listener roles is necessary to ensure system stability. We next consider situations where the centralized policy includes stochastic behavior, and we identify how speakers should optimally trade-off between mimicking this desired noise and effectively communicating with the listener.

Do We Need to Change Roles? Imagine that we are the speaker within the table-carrying example. In the best case, our actions completely convey our observations to our partner. But if we are never a listener, we never know what our partner observes; and, if the team's behavior depends on our own actions, this can lead to situations where we are unable to collaboratively accomplish the task:

4

Proposition 1. *If the teammates never change speaker and listener roles, and the team is attempting to mimic a centralized controller $a^* = -K^*s$, there exist controllable system dynamics for which the decentralized team $a = -Ks$ becomes unstable for any choice of K.*

Proof. We prove this by example. Let agent 1 always be the speaker and let agent 2 be the listener. Consider the following controllable system with linear dynamics $\dot{s} = As + Ba$, where $a = -Ks$:

$$\begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} K_{11} & 0 \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \tag{4}$$

Note that $K_{12} = 0$; this is because the speaker makes decisions based purely on their own state, s_1. The listener is able to perfectly infer s_1 since it observes the speaker's action a_1, and $s_1 = -K_{11}^{-1}a_1$. For this team to have stable dynamics the eigenvalues of $A - BK$ must have a strictly negative real part. But here the eigenvalues are: $1 \pm \sqrt{-K_{11}}$. Hence, no matter what speaker and listener gains K_{11}, K_{21}, and K_{22} we choose, the decentralized team becomes unstable. $\square$

How Should We Speak and Listen with Noise? Returning to our table carrying example, we now know that we should alternate speaker and listener roles even when our actions completely convey our observations. But what if we cannot fully observe the actions of our partner (e.g. due to sensor noise)? As long as this estimate is unbiased, we can still match the expected behavior of the centralized team by treating these noisy actions as accurate measurements:

Proposition 2. *If the teammates incorrectly sense their partner's action a_i as $a_i + n_i$, where n_i is unbiased noise, the decentralized team can match the centralized controller action $a^* = -K^*s$ in expectation by speaking and listening as if $a_i + n_i$ were the teammate's true action.*

Proof. Let agent 1 be the speaker, let agent 2 be the listener, and let the decentralized controller be $a = -Ks$, where K is shown in Eqn. (4). The speaker takes action $a_1 = -K_{11}s_1$, but the listener measures $a_1 + n_1$, and infers the speaker's state as $\hat{s}_1 = s_1 - K_{11}^{-1}n_1$. Accordingly, when the listener acts based on $\hat{s}_1$, their action has an additional term $K_{21}K_{11}^{-1}n_1$. Next the agents switch roles, and the first agent (i.e., the listener) acts with an additional term $K_{12}K_{22}^{-1}n_2$. When the agents rapidly alternate between speaker and listener roles, the team's behavior includes these terms:

$$a = -K^*s + \begin{bmatrix} K_{12}^* K_{22}^{*,-1} n_2 \\ K_{21}^* K_{11}^{*,-1} n_1 \end{bmatrix}, \quad K^* = \begin{bmatrix} K_{11}^* & K_{12}^* \\ K_{21}^* & K_{22}^* \end{bmatrix}, \quad \mathbb{E}_n[a] = -K^*s = a^* \tag{5}$$

This matches the centralized team's action in expectation if n_1 and n_2 are unbiased. $\square$

Noisy action measurements are undesirable but often unavoidable. Conversely, there are cases where the centralized policy itself *recommends* stochastic actions: i.e., when an agent sees an obstacle in front of it, it should go around on the right-side 30% of the time, and on the left otherwise. The speaker and listener can *choose* whether or not to incorporate this stochasticity. Intuitively, we might think that both the speaker and listener should match the optimal centralized policy; but, when the speaker's actions become more stochastic, it is harder for the listener to correctly interpret what the speaker has observed. Imagine we are the listener: the more random the speaker's action is, the less information we have about what the speaker sees, which makes it more challenging for us to learn the system state. This leads to a trade-off between speaker noise and overall team performance:

Proposition 3. *If the centralized policy is stochastic, so that $a^* = -K^*s + n$, and n is sampled from a Gaussian distribution $n \sim \mathcal{N}(0, diag(w_1^2, w_2^2))$, the speaker policy that minimizes the Kullback-Leibler (KL) divergence between the decentralized and centralized policies has a variance less than or equal to the corresponding variance of the centralized policy.*

Proof. Let the first agent be the speaker and let the second agent be the listener. The decentralized team takes action $a = -Ks + m$, where $m \sim \mathcal{N}(0, diag(\sigma_1^2, \sigma_2^2))$. We solve for the variances σ_1^2 and σ_2^2 that minimize the KL divergence between the decentralized and centralized policies:

$$\min_{\sigma_1^2, \sigma_2^2} \text{KL}\Big[\mathcal{N}\big(-Ks, diag(\sigma_1^2, \sigma_2^2)\big), \, \mathcal{N}\big(-K^*s, diag(w_1^2, w_2^2)\big)\Big] \tag{6}$$

Selecting K to best match K^*, and taking the expectation of Eqn. (6), the optimal variances are:

$$\sigma_1^2 = \frac{K_{11}^2 w_1^2 w_2^2}{K_{11}^2 w_2^2 + K_{21}^2 w_1^2} \le w_1^2, \quad \sigma_2^2 = w_2^2 \tag{7}$$

See the appendix for our full derivation. Hence, while the listener should match the variance of the corresponding centralized policy, the speaker should intentionally choose a variance $\sigma_1^2 \le w_1^2$. $\square$
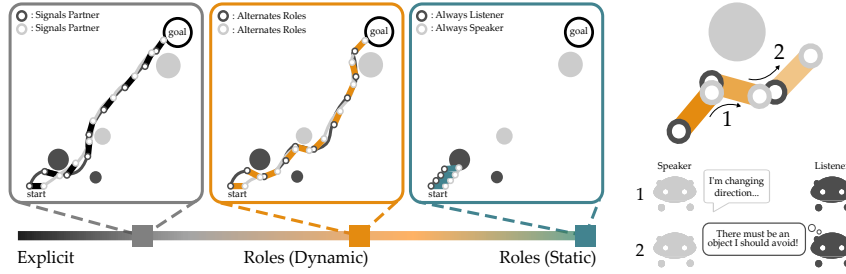
Figure 2: Simulated task. (Left) Two agents collaborate to carry a rigid object while avoiding obstacles. Each agent can only see the obstacles matching their color. In this example, the *explicit* and *dynamic roles* teams are able to negotiate the obstacles to reach the goal, while the *static roles* team fails. (Right) Example of implicit communication via actions: here the speaker sees an obstacle, and abruptly moves to the right (1). The listener updates its state estimate based on this unexpected motion, and also moves to avoid the unseen obstacle (2).

Summary. If two decentralized agents are collaborating with roles, they can match the behavior of a centralized team when their actions completely convey their observations. But even in these cases, alternating between speaker and listener roles is necessary; the team may become unstable if their roles never change. When choosing how best to speak and listen, greedily mimicking the centralized policy is effective, and robust to noisy, unbiased measurements of the partner's actions. In situations where the centralized policy is stochastic, however, a trade-off emerges: speakers should select more deterministic actions in order to better convey their observations to the listener.

5   Simulations and Experiments

Roles are sufficient when actions can completely convey the state an agent observes. But what about more general situations, where the observation space has a higher dimension than the action space? And how do our proposed roles function on actual robot teams? Here we explore these challenging cases in a simulated game (see Fig. 2) and a two-robot manipulation task (see Fig. 4). We compare fixed and dynamic role allocations to different amounts of explicit communication. Overall, we find that there is a *spectrum* across explicit and implicit communication, and that implicit communication via roles approaches the performance of equivalent explicit communication.

5.1   Simulated Table-Carrying Game with Implicit and Explicit Communication

We simulated a continuous state-action task in which two point robots carried a table across a plane (see Fig. 2). Each robot observed half of the obstacles within this plane, and so together the team needed to communicate to obtain an accurate state estimate. We compared using *explicit* and *implicit* communication. During explicit communication, agents sent and received messages that contained the exact location and geometry of the closest obstacle. By contrast, during implicit communication the agents leveraged roles to learn from their partner's actions. An example is shown in Fig. 2: here the speaker sees an obstacle—that the listener cannot observe—and changes its motion to implicitly indicate to the listener that there is an obstacle directly ahead.

Independent Variables. We varied the (a) communication strategy and (b) task complexity. There were three levels of communication strategy: *explicit*, *dynamic roles*, and *static roles*. Within *explicit* and *dynamic roles*, we also varied the number of timesteps T between communication; i.e., the *explicit* teams could only send messages about the closest obstacle every T timesteps, or, analogously, the speaker and listener roles alternated after T timesteps. In order to adjust the task complexity, we increased n, the total number of obstacles in the environment.

Dependent Measures. Within each simulation, two agents held a rigid table, and tried to carry that table to the goal without colliding with an obstacle. We measured the *success rate* (λ) of reaching the goal over 1000 randomly generated environments. As a baseline, we also tested fully centralized teams: these centralized teams reached the goal in all environments ($\lambda = 1$).

Hypotheses:

H1. *Dynamically alternating roles leads to better performance than fixed role allocations, even in cases where actions can completely convey the location of the closest obstacle.*

H2. *In environments where actions cannot completely convey the closest obstacle, dynamic role allocations perform almost as well as explicitly communicating the closest obstacle.*

H3. *Implicit communication via roles is robust to noisy action observations.*

6

Table 1: Success rate λ when both robots knew the geometry of the obstacles *a priori*, but not their locations. Each agent implicitly communicated the positions of the obstacles that they could see through their actions. Importantly, the agents could completely communicate the position of the closest obstacle with their current action: hence, the success rate of *explicit* teams (not listed) is almost identical to that of *dynamic* teams.

	Roles (Dynamic)			Roles (Static)	
Obstacles	$T = 1$	$T = 4$	$T = 16$	Speaker-Listener	Speaker-Speaker
$n = 2$	0.995	0.914	0.827	0.872	0.757
$n = 4$	0.982	0.786	0.656	0.735	0.545
$n = 8$	0.924	0.586	0.434	0.539	0.305

We provide additional details about this simulated game in the Appendix (Sec. 7).

5.2 Rapidly Changing Roles Outperforms Static Roles

We first tested hypothesis **H1** in a setting where each agent knew the geometry of all the obstacles in the environment (i.e., all obstacles were circles with the same radius). Since the obstacle shape is known and fixed, agents could fully convey the closest obstacle's (x, y) location through their 2-DoF actions. Our results are shown in Table 1; we point out that *dynamic roles* and *explicit* are almost the same in this case, and so we focus on comparing *dynamic roles* to *static roles*. For each tested number of obstacles, the teams that rapidly alternated roles outperformed teams with fixed roles. Indeed, when there are only $n = 2$ obstacles, the mapping from action space to observation space was surjective, and the *dynamic roles* team converged towards $\lambda = 1$ as $T \to 0$ (Theorem. 1). But dynamically alternating roles was not always better: when the teammates changed roles too slowly ($T = 16$), their performance was actually worse than maintaining a constant speaker and listener.

5.3 Implicitly Communicating via Roles Competes with Explicit Communication

Next, we explored hypothesis **H2** in more complex settings where the agent's current action could not completely convey what they observed to their partner. Here the robots did not know the obstacle geometry *a priori*; instead, each obstacle radius was randomly sampled from a uniform distribution. Teams with *roles* had to try and implicitly communicate about both the obstacle location and shape— by contrast, *explicit* teams could send messages to their partner that completely conveyed the closest obstacle. Our findings are visualized in Fig. 3. We see a spectrum in performance across *explicit*, *dynamic roles*, and *static roles*. Although directly sending obstacle information is always better than learning from actions, implicitly communicating through *dynamic roles* is almost as successful as *explicitly* communicating with our partner at the same time interval T. This suggests that leveraging roles to learn from our partner's actions can be nearly as effective as direct communication, even in cases where the observations cannot be completely conveyed in a single action.

5.4 Roles with Noisy Action Observations Match Noisy Explicit Communication

So far we have conducted simulations in settings where the agents can perfectly measure the communication of their partner. Now we take a step back, and consider hypothesis **H3** when the communication channel *itself* is noisy. This encompasses scenarios where the partners cannot perfectly measure each other's actions, or, analogously, when the explicit messages are corrupted. Our results with zero mean Gaussian noise are listed in Table 2. As expected, including noise decreased perfor-
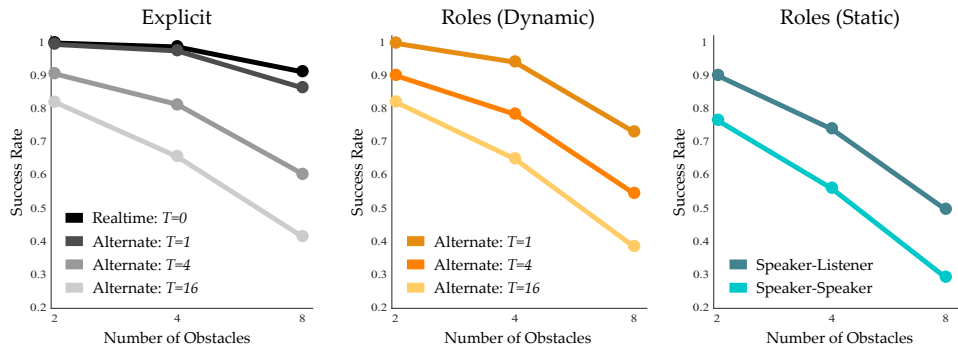

Figure 3: Success rate λ when the robots did not know the obstacle geometry *a priori*. Not only was the observation space higher dimensional than the action space, but agents could not even convey the closest obstacle's position and radius in a single action. There is a spectrum in performance across both communication type and frequency. For realtime explicit communication, the agents fully convey the closest obstacle at every timestep.

Table 2: Success rate λ when the communication was noisy. We simulated zero-mean Gaussian noise, and chose the variance so that the coefficient of variation was 0.1. Both explicit communication (noisy messages) and implicit communication (noisy action observations) had the same noise ratio. Dynamic roles performed almost on par with realtime explicit communication, where both agents exchanged messages at every timestep.

	Explicit	Roles (Dynamic)			Roles (Static)	
Obstacles	$T = 0$	$T = 1$	$T = 4$	$T = 16$	Speaker-Listener	Speaker-Speaker
$n = 2$	0.925	0.884	0.818	0.787	0.829	0.757
$n = 4$	0.833	0.747	0.669	0.606	0.655	0.545
$n = 8$	0.553	0.512	0.416	0.357	0.398	0.305

mance across both *explicit* and *dynamic roles*. But teams with *dynamic roles* that rapidly alternated were still almost on par with *explicit* teams that communicated in realtime, demonstrating that roles were as robust to noisy actions as explicit teams were to noisy messages. Similar to before, fixed speaker-listener teams were more successful than teams that slowly alternated roles.

5.5 Manipulation Experiments with Two Robot Arms

We implemented our *dynamic role* and *static role* strategies on two decentralized robot arms (Panda, Franka Emika). Our experimental setup is shown in Fig. 4: the robots were controlled using separate computers, and were tasked with placing a rod on the table top without any explicit communication. In order to complete this task, the robots had to negotiate two obstacles—but, like in our simulations, each robot could only see one of these obstacles. Because of their differing knowledge about the system state, the robots originally had opposite plans about how to move the rod to the table: one robot wanted to move the rod forwards (with respect to the camera), while the other robot planned to move the rod backwards. We expect intelligent robots to recognize that there is a *reason* why their partner disagrees with them, and *learn* from their partner's actions in realtime to update their estimate of the system state. We controlled the robots using static speaker-speaker and speaker-listener roles, as well as dynamic roles that alternated every 0.5 s. During the experiments, only the *dynamic roles* team inferred the obstacles that their partners observed, and aligned their actions to successfully place the rod on the table (see Fig. 4 and ai.stanford.edu/blog/learning-from-partners/).

6 Discussion

Summary. Decentralized robot teams should learn about what their partner observes based on their partner's actions, but this is not possible when both agents attempt to simultaneously exploit their observations and communicate with their partner. We have therefore introduced separate speaker and listener roles: our analysis shows that teammates which dynamically alternate roles theoretically and experimentally approach the performance of teammates that can explicitly communicate.

Limitations and Future Work. This work is limited to Dec-MDPs, where the agents collectively observe the full system state. But we recognize that often there are parts of the state that neither agent can fully observe; accordingly, our future work will focus on extending roles to Dec-POMDPs.
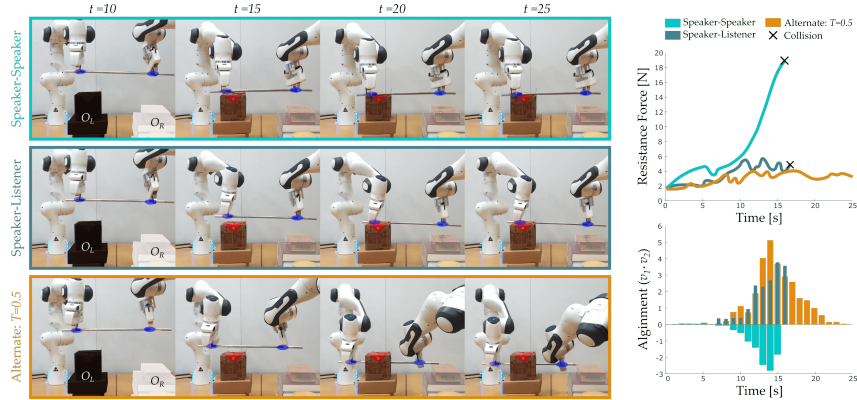

Figure 4: Two decentralized robot arms tasked with placing a rod on the table while implicitly communicating via roles: the left robot sees obstacle O_L and the right robot sees O_R. (Left) The behavior of *static role* and *dynamic role* teams after t s. Both static role teams collided with obstacle O_L because they failed to mutually communicate their observations. The team that alternated roles successfully reasoned over each other's actions, conveyed the obstacle positions, and aligned their actions. (Right) We plot the norm of the force the two agents exerted on one another, as well as the alignment between the two agent's end-effector movements.

References

[1] A. Prorok, A. Bahr, and A. Martinoli. Low-cost collaborative localization for large-scale multi-robot systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4236–4241, 2012.

[2] N. Ayanian and V. Kumar. Decentralized feedback controllers for multiagent teams in environments with obstacles. *IEEE Transactions on Robotics*, 26(5):878–887, 2010.

[3] P. Song and V. Kumar. A potential field based approach to multi-robot manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1217–1222, 2002.

[4] K. Sreenath and V. Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems (RSS)*, 2013.

[5] S. L. Bowman, C. Nowzari, and G. J. Pappas. Coordination of multi-agent systems via asynchronous cloud communication. In *IEEE Conference on Decision and Control (CDC)*, pages 2215–2220. IEEE, 2016.

[6] P. Culbertson and M. Schwager. Decentralized adaptive control for collaborative manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 278–285, 2018.

[7] N. Nigam, S. Lall, P. Hovareshti, K. Ezra, L. Mockus, D. Tolani, and S. Sloan. Sufficient statistics for optimal decentralized control in system of systems. In *2018 First International Conference on Artificial Intelligence for Industries (AI4I)*, pages 92–95. IEEE, 2018.

[8] L. Lessard and S. Lall. Optimal control of two-player systems with output feedback. *IEEE Transactions on Automatic Control*, 60(8):2129–2144, 2015.

[9] C. Amato, G. Chowdhary, A. Geramifard, N. K. Üre, and M. J. Kochenderfer. Decentralized control of partially observable Markov decision processes. In *IEEE Conference on Decision and Control (CDC)*, pages 2398–2405, 2013.

[10] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[11] A. Nayyar, A. Mahajan, and D. Teneketzis. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658, 2013.

[12] C. Amato, G. Konidaris, G. Cruz, C. A. Maynor, J. P. How, and L. P. Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1241–1248, 2015.

[13] F. S. Melo and M. Veloso. Decentralized MDPs with sparse interactions. *Artificial Intelligence*, 175(11):1757–1789, 2011.

[14] J. Swigart and S. Lall. Optimal controller synthesis for a decentralized two-player system with partial output feedback. In *American Control Conference (ACC)*, pages 317–323, 2011.

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

[16] 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 (NeurIPS)*, pages 6379–6390, 2017.

[17] J. K. Gupta, M. Egorov, and M. Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 66–83, 2017.

[18] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. In *AAAI Conference on Artificial Intelligence*, 2018.

[19] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252, 2013.

[20] L. B. Johnson, H.-L. Choi, and J. P. How. The role of information assumptions in decentralized task allocation: A tutorial. *IEEE Control Systems Magazine*, 36(4):45–58, 2016.

[21] S. Sukhbaatar, R. Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2244–2252, 2016.

[22] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2137–2145, 2016.

[23] J. Paulos, S. W. Chen, D. Shishika, and V. Kumar. Decentralization of multiagent policies by learning what to communicate. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[24] Z. Yan, N. Jouandeau, and A. A. Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10(12):399, 2013.

[25] Z. Wang and M. Schwager. Kinematic multi-robot manipulation with no communication using force feedback. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 427–432. IEEE, 2016.

[26] E. Pagello, A. D'Angelo, F. Montesello, F. Garelli, and C. Ferrari. Cooperative behaviors in multi-robot systems through implicit communication. *Robotics and Autonomous Systems*, 29 (1):65–77, 1999.

[27] H. Zhang, J. Chen, H. Fang, and L. Dou. A role-based pomdps approach for decentralized implicit cooperation of multiple agents. In *2017 13th IEEE International Conference on Control & Automation (ICCA)*, pages 496–501. IEEE, 2017.

[28] P. Nalepka, R. W. Kallen, A. Chemero, E. Saltzman, and M. J. Richardson. Herd those sheep: Emergent multiagent coordination and behavioral-mode switching. *Psychological Science*, 28 (5):630–650, 2017.

[29] R. X. Hawkins, N. D. Goodman, and R. L. Goldstone. The emergence of social norms and conventions. *Trends in Cognitive Sciences*, 2018.

[30] K. B. Reed and M. A. Peshkin. Physical collaboration of human-human and human-robot teams. *IEEE Transactions on Haptics*, 1(2):108–120, 2008.

[31] A. Mörtl, M. Lawitzky, A. Kucukyilmaz, M. Sezgin, C. Basdogan, and S. Hirche. The role of roles: Physical cooperation between humans and robots. *The International Journal of Robotics Research*, 31(13):1656–1674, 2012.

[32] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O'Malley. A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction. *Applied Mechanics Reviews*, 70(1):010804, 2018.

[33] D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan. Cooperative inverse reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS*, pages 3909–3917, 2016.

[34] R. Dobbe, D. Fridovich-Keil, and C. Tomlin. Fully decentralized policies for multi-agent systems: An information theoretic approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2941–2950, 2017.

[35] J. Barraquand, B. Langlois, and J.-C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(2):224–241, 1992.

7 Appendix

7.1 Derivations for Eqn. (1)

$$\pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t) = \sum_{s_2^{0:t}} P(a_1^t, s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1}, a_2^t)$$

$$= \frac{\sum_{s_2^{0:t}} P(a_1^t, a_2^t, s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1})}{P(a_2^t \mid s_1^{0:t}, a^{0:t-1})}$$

$$\propto \sum_{s_2^{0:t}} P(a_1^t, a_2^t, s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1}) \tag{8}$$

$$= \sum_{s_2^{0:t}} P(a_2^t \mid a_1^t, s_2^{0:t}, s_1^{0:t}, a^{0:t-1}) \cdot P(a_1^t \mid s_2^{0:t}, s_1^{0:t}, a^{0:t-1})$$

$$\cdot P(s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1})$$

$$= \sum_{s_2^{0:t}} \pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \cdot \pi_1^*(a_1^t \mid s_1^t, s_2^t) \cdot P(s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1})$$

Thus, we reach the policy for agent 1:

$$\pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t) \propto \sum_{s_2^{0:t}} \pi_1^*(a_1^t \mid s_1^t, s_2^t) \cdot \pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \cdot P(s_2^{0:t} \mid s_1^{0:t}, a^{0:t-1}) \tag{9}$$

We can use the same steps to derive the symmetric policy for agent 2:

$$\pi_2(a_2^t \mid s_2^{0:t}, a^{0:t-1}, a_1^t) \propto \sum_{s_1^{0:t}} \pi_2^*(a_2^t \mid s_1^t, s_2^t) \cdot \pi_1(a_1^t \mid s_1^{0:t}, a^{0:t-1}, a_2^t) \cdot P(s_1^{0:t} \mid s_2^{0:t}, a^{0:t-1}) \tag{10}$$

7.2 Extension of Theorem. 1 to N-Agent Teams

Moving beyond two-agent teams, robots can also alternate between speaker and listener roles in N-agent scenarios to obtain optimal centralized performance. Let one agent be the listener, and let the other $N-1$ agents speak to this listener. As in the two-agent case, we define a_i^* as the optimal centralized action for agent i, and we define $\bar{a}_i$ as the naive action that purely depends on agent i's local state. Under our surjective mapping assumption, we have $g_i(\bar{a}_i) = s_i$. We choose the speaker i's action to be $\bar{a}_i$ and the listener j's action to be $a_j^* + (N-1)(a_j^* - \bar{a}_j)$. Agents alternate the listener role with interval dt. Taking the limit as $dt \to 0$, the overall team action a becomes:

$$a^t = \lim_{dt \to 0} \frac{\sum_{k=0}^{N-1} a^{t+kdt}}{N}$$

$$= \lim_{dt \to 0} \frac{1}{N} \left(\begin{bmatrix} a_1^{*t} + (N-1)(a_1^{*t} - \bar{a}_1^t) \\ \bar{a}_2^t \\ \vdots \\ \bar{a}_N^t \end{bmatrix} + \begin{bmatrix} \bar{a}_1^{t+dt} \\ a_2^{*t+dt} + (N-1)(a_2^{*t+dt} - \bar{a}_2^{t+dt}) \\ \vdots \\ \bar{a}_N^{t+dt} \end{bmatrix} \right.$$

$$\left. + \ldots + \begin{bmatrix} \bar{a}_1^{t+(N-1)dt} \\ \bar{a}_2^{t+(N-1)dt} \\ \vdots \\ a_N^{*t+(N-1)dt} + (N-1)(a_N^{*t+(N-1)dt} - \bar{a}_N^{t+(N-1)dt}) \end{bmatrix} \right)$$

$$= a^{*t}$$

And so, following the simple yet effective role rotation mechanism, the decentralized team converges to the optimal action of the centralized team as role alternation frequency increases.

7.3 Derivations for Eqn. (7)

For the stochastic centralized policy, we have:

$$\pi_1^*(a_1 \mid s_1, s_2) = \mathcal{N}(K_{11}s_1 + K_{12}s_2, w_1^2)$$
$$\pi_2^*(a_2 \mid s_1, s_2) = \mathcal{N}(K_{21}s_1 + K_{22}s_2, w_2^2) \tag{11}$$

The states s_1 and s_2 are constant and independent, and sampled from Gaussian priors: $s_1 \sim \mathcal{N}(\mu_1, \sigma_{s_1}^2)$ and $s_2 \sim \mathcal{N}(\mu_2, \sigma_{s_2}^2)$. Without loss of generality, let agent 1 be speaker and agent 2 is listener. The speaker and listener policies are:

$$\pi_1(a_1 \mid s_1) = \mathcal{N}(K_{11}s_1 + K_{12}\mu_2, \sigma_1^2) \tag{12}$$

$$\pi_2(a_2 \mid s_2, a_1) = \mathcal{N}(K_{21}K_{11}^{-1}(a_1 - K_{12}\mu_2) + K_{22}s_2, \sigma_2^2) \tag{13}$$

Now we can solve for the parameters $\theta = \{\sigma_1, \sigma_2\}$ that minimize the KL divergence between actual and desired policies:

$$
\min_\theta KL = \min_\theta \left[\log\frac{w_1}{\sigma_1} + \frac{\sigma_1^2 + (K_{11}s_1 + K_{12}s_2 - K_{11}s_1 - K_{12}\mu_2)^2}{2w_1^2} + \log\frac{w_2}{\sigma_2} + \right.
$$
$$
\left. \frac{\sigma_2^2 + (K_{21}s_1 + K_{22}s_2 - K_{21}K_{11}^{-1}(a_1 - K_{12}\mu_2) - K_{22}s_2)^2}{2w_2^2} \right]
$$
$$
= \min_\theta \left[\log\frac{w_1 w_2}{\sigma_1 \sigma_2} + \frac{\sigma_1^2}{2w_1^2} + \frac{K_{12}^2(s_2 - \mu_2)^2}{2w_1^2} + \frac{\sigma_2^2 + K_{21}^2(s_1 - K_{11}^{-1}(a_1 - K_{12}\mu_2))^2}{2w_2^2} \right] \tag{14}
$$

Taking the expectation over Eqn. (14), we reach:

$$\min_\theta \mathbb{E}[KL] = \min_\theta \left[\frac{K_{12}^2 \sigma_{s_2}^2}{2w_1^2} + \log\frac{w_1 w_2}{\sigma_1 \sigma_2} + \frac{\sigma_1^2}{2w_1^2} + \frac{\sigma_2^2 + K_{21}^2 \sigma_1^2 / K_{11}^2}{2w_2^2} \right] \tag{15}$$

Thus, the optimal choices of σ_1 and σ_2 that minimize the KL divergence are:

$$\sigma_1 = \frac{K_{11}w_1 w_2}{\sqrt{K_{11}^2 w_2^2 + K_{21}^2 w_1^2}}, \quad \sigma_2 = w_2 \tag{16}$$

7.4 Simulation Environment

In our simulations the agents move with velocity v_i. Formally, Eqn. (17) gives the system dynamics:

$$v_c^{trans} = \frac{1}{2}(v_1 + v_2), \quad \omega = \frac{1}{r}u_i^r \times (v_i - v_c^{trans}) \tag{17}$$

The table's translation velocity v_c^{trans} is the average of two agents' input velocities. The angular velocity is computed by first finding the difference in velocities orthogonal to the table and then dividing by r, the length from table center to the agents.

7.5 Planning

We adopt the well-known potential field model [35] for planning. Agents at location q plan their path under the influence of an artificial potential field $U(q)$, which is constructed to reflect the environment. There are two types of potential field sources. We denote the set of attractors as $k \in \mathcal{A}$ and the set of repulsive obstacles as $j \in \mathcal{R}$. The overall potential field is the sum of all attractive and repulsive potential field sources as in Eqn. (18). Agents move along the gradient of this potential field Eqn. (19). w_v is a predefined hyperparameter controlling scaling.

$$U(q) = \sum_{k \in \mathcal{A}} U_{att}^k(q) + \sum_{j \in \mathcal{R}} U_{rep}^j(q) \tag{18}$$

$$v_i(q_i) = -w_v \nabla U(q_i) = -w_v \left(\sum_{k \in \mathcal{A}_i} \nabla U_{att}^k(q_i) + \sum_{j \in \mathcal{R}_i} \nabla U_{rep}^j(q_i) \right) \tag{19}$$

The gradients for the attractive and repulsive fields are shown in Eqn. (20) and Eqn. (21). w_{att} and w_{rep} are hyperparameters controlling the relative scale of the attractive and repulsive fields. q_k is the location for attractor k. q_j is the location for repulsive source j. $\rho_j(q)$ is the minimum distance of agent at location q to obstacle j. ρ_0 is a hyperparameter controlling the effective range of the repulsive potential field. The repulsive potential field is 0 outside of the range ρ_0.

$$\nabla U_{att}^k(q) = w_{att} * \frac{q - q_k}{\|q - q_k\|} \tag{20}$$

$$\nabla U_{rep}^j(q) = \begin{cases} w_{rep} * \left(\frac{1}{\rho_j(q)} - \frac{1}{\rho_0} \right) \left(\frac{1}{\rho_j(q)} \right) \frac{q - q_j}{\|q - q_j\|} & \text{if } \rho_j(q) \leq \rho_0 \\ 0 & \text{if } \rho_j(q) > \rho_0 \end{cases} \tag{21}$$

Table 3: Average game length in failure cases when both robots knew the geometry of the obstacles *a priori*.

| Obstacles | Roles (Dynamic) | | | Roles (Static) | |
	$T=1$	$T=4$	$T=16$	Speaker-Listener	Speaker-Speaker
n = 2	**164.00**	152.93	146.53	145.96	143.13
n = 4	**181.11**	154.98	149.99	148.04	146.75
n = 8	**167.61**	160.47	156.89	158.13	154.65

7.6 Inference

As discussed in Sec. 3 and Sec. 4, the listener performs inference over the speaker i's action $v_i(q_i)$. Since the goal attractor is known and shared in our collaborative tasks, the listener is trying to infer the speaker's repulsive obstacles $j \in \mathcal{R}_i$. In our implementation, the listener approximates speaker i's repulsive potential field with one inferred obstacle $\bar{j}_i$ at location $\bar{q}_i$. More formally, the listener solves:

$$v_i(q_i) = -w_v \Big(\sum_{k \in \mathcal{A}_i} \nabla U_{att}^k(q_i) + \nabla U_{rep}^{\bar{j}_i}(\bar{q}_i) \Big) \tag{22}$$

Combined with Eqn. (20) and Eqn. (21), the numerical solution for obstacle $\bar{j}_i$'s location $q_{\bar{j}_i}$ is obtained by bisection iteration.

7.6.1 Supplementary Results: Rapidly Changing Roles Outperforms Static Roles

In addition to the success rate λ from Table 1, we also measured the average game length l. The average game length *during only failure cases* is shown in Table 3. Changing roles rapidly fails in games that require more time steps. This result indicates that the cases where *Roles (Dynamic)* fail are the most difficult, requiring longer and more complicated trajectories to avoid the obstacles.

7.6.2 Supplementary Results: Roles with Noisy Action Observations Match Noisy Messages

In Sec. 5.4, we explored noisy actions where the coefficient of variance was 0.1 (see Table 2). Here we also provide the results from different coefficients of variation: $\frac{\sigma}{\mu} \in \{0.001, 0.01, 0.1\}$. Our full results are summarized in Table 4. As noise level increases, the success rate gradually drops and the gap between dynamic roles and explicit communication becomes greater. This is because the error gets amplified by the nonlinearity in the action inference process. Nevertheless—across all cases— the dynamic role strategy dealt with noise and performed on par with explicit communication.

Table 4: Success rate λ over 1000 games when communication is noisy. $\frac{\sigma}{\mu}$ is coefficient of variation that controls the noise level for both action observation and explicit communication. Here S-L represents the static Speaker-Listener team and S-S represents the static Speaker-Speaker team.

| Obstacles | noise level $\frac{\sigma}{\mu}$ | Explicit | Roles (Dynamic) | | | Roles (Static) | |
		$T=0$	$T=1$	$T=4$	$T=16$	S-L	S-S
	0.001	0.997	0.989	0.89	0.818	0.889	
n=2	0.01	0.996	0.980	0.891	0.814	0.889	0.758
	0.1	0.925	0.884	0.818	0.787	0.829	
	0.001	0.984	0.943	0.778	0.653	0.731	
n=4	0.01	0.981	0.927	0.768	0.651	0.731	0.558
	0.1	0.833	0.747	0.669	0.606	0.655	
	0.001	0.904	0.738	0.531	0.388	0.495	
n=8	0.01	0.900	0.716	0.504	0.388	0.491	0.296
	0.1	0.553	0.512	0.416	0.357	0.398	