OPT2025: 17th Annual Workshop on Optimization for Machine Learning

From Emergence to Intention: A Statistical Inductive Bias for Tractable Optimization in Multi-Agent Coordination

Brennen A. Hill* BAHILL4@WISC.EDU
Madison, WI 53706

Mant Koh En Wei E0958776@U.NUS.EDU
Thangavel Jishnuanandh JISHNUANANDH@U.NUS.EDU
Singapore 119077

Abstract

Cooperative multi-agent reinforcement learning (MARL) presents a formidable non-convex optimization challenge, exacerbated by the non-stationarity introduced by concurrently learning agents. A common approach is to enable communication, allowing agents to coordinate. In this paper, we investigate how the structure of this communication impacts the tractability of the underlying optimization problem. We first analyze an end-to-end learned protocol, Learned Direct Communication (LDC), where messages emerge as part of the policy optimization. We then propose **Intention Communication**, an alternative that structures the optimization by integrating a learned statistical world model. Agents use this model to generate predictive trajectories via self-simulated rollouts and then communicate a compressed summary of this intent. This approach injects a strong inductive bias, reframing the optimization problem from discovering a communication protocol from scratch to learning to interpret explicit plans. We evaluate these methods on a cooperative task-allocation problem under partial observability. Our results demonstrate that while emergent communication is viable in simple settings, its performance degrades with scale. In contrast, by structuring the problem with a predictive statistical model, Intention Communication dramatically improves optimization stability and sample efficiency, achieving near-optimal performance in complex environments where other methods fail. This work underscores the efficacy of bridging statistical modeling with optimization to solve complex coordination problems.

Keywords: Non-convex Optimization, Multi-Agent Reinforcement Learning (MARL), Statistical World Models, Inductive Bias, Sample Efficiency, Emergent Communication, Optimization for Deep Learning, Scalability

1. Introduction: The Intersection of Optimization and Statistics in MARL

Multi-agent reinforcement learning (MARL) is a powerful paradigm for solving complex, decentralized problems [1, 12]. However, training MARL systems poses a significant optimization challenge. From the perspective of any single agent, the environment is *non-stationary*: the policies of other agents are constantly changing, which in turn alters the state transition and reward dynamics. This violates the Markov assumption underpinning most single-agent RL algorithms and can destabilize the optimization process, hindering convergence to an effective joint policy [4].

Communication is a natural mechanism to mitigate this non-stationarity. By exchanging information, agents can synchronize knowledge, align intentions, and make their behavior more predictable [7]. A key question at the intersection of optimization and statistics is how this communi-

* Corresponding author.

cation should be structured. One approach is to treat the communication protocol as another set of parameters to be learned end-to-end via optimization, a strategy often termed "emergent communication" [2, 8]. While appealingly general, this approach dramatically expands the search space, potentially leading to a high-variance learning signal that requires vast numbers of samples to converge.

An alternative is to inject statistical principles into the optimization process. If agents can build and share predictive models of their own future behavior, they can provide a more structured, low-variance signal to their teammates. This reframes the problem: instead of learning *what* to say from an arbitrary vocabulary, agents learn to interpret explicit, statistically-grounded plans. How can statistical properties and models of the data, in this case, an agent's predicted future trajectory, be leveraged to make the optimization problem more tractable?

In this paper, we explore this question in the context of a cooperative task allocation problem. We compare two distinct approaches:

1. **Learned Direct Communication (LDC):** A pure optimization approach where agents learn a low-bandwidth communication protocol from scratch, concurrently with their action policy.

2. **Intention Communication:** A hybrid approach where we introduce a strong inductive bias. Each agent uses a learned statistical world model, the **Imagined Trajectory Generation Module (ITGM)**, to generate an imagined trajectory, then communicates a compressed representation of this plan via a **Message Generation Network (MGN)**.

We demonstrate that while LDC can establish a simple protocol, it fails to scale as the statistical complexity of the environment increases. In contrast, Intention Communication provides a structured optimization landscape that enables agents to learn robust, scalable, and sample-efficient policies, even under significant computational constraints.

2. Experimental Framework

Environment and Task To analyze the efficacy of different communication strategies, we use a deterministic $N \times N$ grid world, implemented with PettingZoo [10]. The environment contains two agents and two goals at distinct, randomly initialized locations. The cooperative task is successful if each agent navigates to a unique goal. This setup creates a coordination challenge where agents must resolve the ambiguity of which goal to pursue. An episode terminates upon success or after 200 timesteps.

Observations and State Space In the fully observable setting, an agent's observation $o_t^{(i)}$ is the 4-tuple of goal coordinates. In the more challenging partially observable setting, an agent only observes goals within a fixed 'vision_range'. To provide agents with short-term memory, the input to their policy network is a stack of the last four observations. To prevent agents from learning trivial static policies (e.g., agent 1 always takes the first goal), the order of goals in one agent's observation vector is randomly shuffled each episode. Agents can choose one of five actions: stay, up, down, left, or right.

Shared Reward and Optimization We employ a shared, symmetric reward signal to foster cooperation: $+1.0$ for success (agents on different goals), -0.1 for collision (agents on the same goal), and a small per-step penalty of -0.01 to encourage efficiency. We use an on-policy, Advantage

Actor-Critic (A2C) optimization algorithm [6, 9]. The policy and value functions are updated at the end of each episode. To balance exploration and exploitation, we use a linear learning rate decay schedule, starting at an initial lr_0 and decreasing to a minimum of 1×10^{-5} over the course of training. All experiments were conducted on Google Colab, imposing computational constraints that favor sample-efficient methods.

3. Learned Direct Communication (LDC): An End-to-End Optimization Approach

Our first approach, LDC, treats communication as an emergent phenomenon learned through end-to-end optimization. Each agent's policy network simultaneously outputs a distribution over actions and a distribution over messages. The message from agent j at time t is sampled and becomes part of agent i's input at time $t + 1$. The communication protocol is thus implicitly defined, without any explicit reward or structure to shape its meaning.

For our experiments, we use a minimal binary message space $\{0, 1\}$. Our objective is to see if the optimizer can discover a meaningful protocol to encode information, such as intended goals.

Analysis in a Fully Observable Setting In a 6×6 fully observable grid, agents successfully learned to coordinate. To verify that the learned messages were statistically meaningful, we analyzed the conditional probability of an agent's action given the message it received. We observed a clear correlation; for instance, Agent 1 was substantially more likely to move 'Up' when receiving a message of '0' (13.54%) than '1' (4.53%). This indicates that agents learn an implicit model of each other's message-conditioned policies. An ablation study confirmed this: disabling communication by fixing messages to a constant value of 0 resulted in a slight drop in success rate (89.4% to 88.6%) and a corresponding increase in steps to completion. The optimizer successfully found a local minimum where messages convey useful, albeit noisy, information.

Performance under Partial Observability We then moved to a partially observable 6×6 grid with 'vision_range = 3'. Here, communication is more critical, as agents may need to share information about goals outside their teammate's view. LDC still showed a benefit, with the success rate dropping from 31.89% to 30.26% when communication was ablated. However, the overall performance is significantly lower, suggesting that as the statistical complexity of the task increases, this simple, unstructured optimization approach begins to struggle. The emergent protocol, while better than nothing, provides a high-variance signal that is insufficient for robust coordination under uncertainty.

4. Intention Communication: Integrating a Statistical Forward Model

To address the scaling limitations of LDC, we designed **Intention Communication**, an architecture that injects a strong inductive bias into the learning process. Instead of asking the optimizer to discover a protocol from scratch, we structure the problem around a core statistical concept: prediction. Agents first use a learned world model to form a plan, then communicate that plan. This transforms the communication channel from a low-bitrate, arbitrary signal to a structured, high-information message grounded in a forward-looking simulation.

The architecture consists of two key modules integrated into the agent's decision-making loop: the Imagined Trajectory Generation Module (ITGM) and the Message Generation Network (MGN).

4.1. Imagined Trajectory Generation Module (ITGM)

The ITGM is a learned, latent-space world model that functions as a statistical forecaster [3]. It combines a learned latent transition model with the agent's own policy to generate a short-term forecast.

1. State Encoding. The agent's current local observation $o_t^{(i)}$ (a stack of frames) and the message received from its teammate, $m_{t-1}^{(j)}$, are concatenated and encoded into an initial latent state vector $z_t^{(i)}$ via a linear encoder:

$$z_t^{(i)} = \text{ReLU}(\mathbf{W}_{enc}[o_t^{(i)} \oplus \text{embed}(m_{t-1}^{(j)})] + \mathbf{b}_{enc})$$

This vector $z_t^{(i)}$ serves as a compressed representation of the agent's current belief state.

2. Latent Rollout (The Plan). The ITGM generates the plan by unrolling a trajectory for a fixed horizon H. Crucially, this rollout actively uses the agent's own policy head (π_{act}) to sample imagined actions. This policy-conditioned simulation ensures the plan is consistent with the agent's actual behavior. For each step k in the horizon ($k = 0, \ldots, H-1$), a planned action $\tilde{a}$ is sampled, and a transition model (f_{trans}) predicts the next latent state:

$$\tilde{a}_{t+k} \sim \pi_{act}(\cdot|z_{t+k}^{(i)})$$

$$z_{t+k+1}^{(i)} = \text{ReLU}(\mathbf{W}_{trans}[z_{t+k}^{(i)} \oplus \text{embed}(\tilde{a}_{t+k})] + \mathbf{b}_{trans})$$

This iterative process generates a sequence of predicted future latent states: $\tau^{(i)} = \langle z_{t+1}^{(i)}, \ldots, z_{t+H}^{(i)} \rangle$.

4.2. Message Generation Network (MGN)

The MGN's role is to perform information compression: it distills the high-dimensional imagined trajectory $\tau^{(i)}$ into a compact message vector $m_t^{(i)}$ that summarizes the agent's intention.

1. Attention Mechanism. We use a multi-head self-attention mechanism [11] to identify the most statistically salient components of the predicted trajectory (e.g., the final destination or a potential conflict point). The sequence $\tau^{(i)}$ serves as the query, key, and value inputs:

$$\mathbf{Q} = \tau^{(i)}\mathbf{W}_Q, \quad \mathbf{K} = \tau^{(i)}\mathbf{W}_K, \quad \mathbf{V} = \tau^{(i)}\mathbf{W}_V$$

The attention output is aggregated via mean pooling to produce a single summary vector v_{pooled}, which is then processed by hidden layers to produce h_2.

2. Multi-Component Discrete Message. To allow for expressive communication, h_2 is fed into N_{comp} separate linear heads. Each head $k \in [1, N_{comp}]$ outputs N_{sym} logits, defining a categorical distribution $\pi_{msg}^{(k)}$.

$$\pi_{msg}^{(k)} = \text{Softmax}(\mathbf{W}_{head}^{(k)} h_2 + b_{head}^{(k)})$$

The final message $m_t^{(i)}$ consists of discrete symbols sampled from each of these N_{comp} distributions. This structured, multi-part message is far more expressive than the single bit used in LDC.

4.3. End-to-End Optimization

The modules are integrated into a plan-then-act sequence. At each timestep, an agent (1) generates its imagined trajectory $\tau^{(i)}$ using the ITGM, (2) compresses it into an intention message $m_t^{(i)}$ using the MGN, and (3) broadcasts this message. Simultaneously, the agent uses the initial latent state $z_t^{(i)}$ (conditioned on the *incoming* message from the teammate) to select its real action $a_t^{(i)}$.

The entire architecture is trained end-to-end using the A2C loss function. Gradients from the actor loss flow back through the MGN and ITGM, updating the message generation, the transition model, and the encoder simultaneously. This unified objective forces the agent to learn what constitutes a good plan (ITGM), how to communicate it effectively (MGN), and how to act upon plans.

5. Results: The Scaling Advantages of Structured Communication

We compared Intention Communication against LDC and a non-communicating baseline in partially observable environments of increasing size, with ‘vision_range = 2‘. Success requires agents to locate and claim distinct goals within 200 steps.

Table 1: Success Rates in Partially Observable Environments of Increasing Size. By structuring the communication channel with a predictive model, Intention Communication enables the optimizer to find near-perfect policies where unstructured methods fail.

Environment	Model	Success Rate
10×10	Baseline (No Communication)	0.0%
10×10	Learned Direct Communication (LDC)	30.8%
10×10	**Intention Communication**	**99.9%**
15×15	Baseline (No Communication)	0.0%
15×15	Learned Direct Communication (LDC)	12.2%
15×15	**Intention Communication**	**96.5%**

The results in Table 1 are stark. The baseline model completely fails, highlighting the necessity of communication. LDC, the pure optimization approach, learns a partially effective strategy in the 10×10 grid but its performance degrades severely as the state space expands (15×15). The optimization struggles to find a meaningful protocol in the face of increased statistical complexity.

In contrast, Intention Communication demonstrates remarkable robustness and scalability. By grounding communication in statistical prediction, it provides a stable learning signal that allows the optimizer to converge to a near-optimal policy in both environments. This high performance, achieved under the computational constraints of Google Colab, highlights the superior sample efficiency of our structured approach. The clear implication is that the engineered inductive bias fundamentally simplifies the optimization landscape, making the coordination problem tractable.

6. Conclusion

This work provides a direct comparison between treating multi-agent communication as a pure optimization problem versus a structured one guided by statistical modeling. Our findings indicate that for non-trivial coordination tasks, relying on an optimizer to discover a communication protocol end-to-end (as in LDC) is a brittle and unscalable strategy. The learning signal is often too high-variance for the optimizer to succeed as problem complexity grows.

By incorporating a learned statistical forward model and an information compression mechanism, our Intention Communication architecture imposes a strong, beneficial inductive bias. This structure does not solve the problem directly, but rather reshapes the optimization landscape, making it dramatically easier for an actor-critic algorithm to discover a robust and scalable cooperative policy. Our results show that integrating principles from statistics, such as predictive modeling and information-theoretic compression, is a powerful method for designing more effective and sample-efficient optimization algorithms for complex machine learning challenges.

Acknowledgments and Disclosure of Funding

We thank Denon Chong Cheng Zong and Jeff Lee for their contributions to research on MARL communication. We thank Dr. Matthew Berland for their mentorship and feedback.

Contributions

Brennen Hill: Project concept, project leadership, environment development, baseline model development, Learned Direct Communication development, Intention Communication development, scaling, optimization, MARL communication research, agent training.
Mant Koh En Wei: Intention Communication development, baseline model development, optimization, scaling, agent training.
Thangavel Jishnuanandh: Baseline model development, MARL communication research, optimization, scaling, agent training.

References

[1] Jesse Anglen. The impact of multi-agent reinforcement learning (marl). *Rapid Innovation*, 2024.

[2] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 29:2137–2145, 2016. doi: 10.5555/3157096.3157290. URL https://papers.nips.cc/paper/6570-learning-to-communicate-with-deep-multi-agent-reinforcement-learning.pdf.

[3] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, 2020.

[4] Su Kefan, Zhou Siyan, Jiang Jiechuan, Gan Chuang, Wang Xiangjun, and Lu Zongqing. Multi-agent alternate q-learning. *IFAAMAS*, 2024.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[6] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[7] Yang Ming, Zhao Kaiyan, Wang Yiming, Dong Renzhi, Du Yali, Liu Furui, Zhou Mingliang, and Hou Leong. Team-wise effective communication in multi-agent reinforcement learning. *Springer US*, 2024.

[8] Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. *Advances in Neural Information Processing Systems*, 2016. URL https://papers.nips.cc/paper/6398-learning-multiagent-communication-with-backpropagation.pdf.

[9] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

[10] J Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[12] Caroline R Vincent. Multi-agent reinforcement learning for autonomous robotics. *DSpace@MIT*, 2024.

Appendix A. Appendix: Supplementary Experiments and Hyperparameters

To further probe the challenges of the end-to-end optimization approach in LDC, we conducted a series of supplementary experiments. These explorations reinforce our conclusion that without beneficial structural priors, the optimization problem of emergent communication is exceedingly difficult, especially under computational constraints.

Varying the Message Space Capacity We hypothesized that LDC's limited performance might stem from its low-bandwidth (1-bit) message. We experimented with increasing the message capacity, either by allowing a wider range of integer values (e.g., 0-4, 0-99) or by sending multiple values per timestep. These attempts invariably failed. The only configurations that achieved any meaningful learning (success rate ¿ 1%) were those with a single message in a binary space. We

conjecture that increasing the message dimensionality adds significant noise and complexity to the joint action-observation space, which destabilizes the actor-critic optimization process and prevents convergence.

Reward Shaping and Architecture We attempted to guide the optimization by adding an explicit reward for sending meaningful messages. This was based on measuring the influence of a received message on the recipient's value function estimate. However, this introduced a difficult hyperparameter tuning problem; high coefficients destabilized learning, while low coefficients were ignored. Similarly, we forced the need for communication by restricting vision such that agents could only see each other's goals. Even in this setting, LDC failed to converge, highlighting the brittleness of the emergent paradigm.

Hyperparameters We provide the hyperparameters used in our experiments in Table 2 to facilitate reproducibility.

Table 2: Key Training and Architectural Hyperparameters

Parameter	Value
Training Hyperparameters	
Initial LR (lr_0)	1×10^{-3}
Final LR (lr_f)	1×10^{-4}
Discount Factor (γ)	0.99
Optimizer	Adam [5]
Critic Loss Coefficient (α_c)	0.5
Max Episode Length (T_{max})	200
Architectural Hyperparameters (all)	
Frame Stack (F)	4
Hidden Dimension (D_{hidden})	64
Hidden Layers (N_{hidden})	2
Architectural Hyperparameters (Intention Communication)	
ITGM Rollout Horizon (H)	3
Attention Heads (N_{heads})	1
Message Components (N_{comp})	8
Symbol Vocabulary Size (N_{sym})	8