CatlNet: Learning Communication and Coordination Policies from CaTL+ Specifications

Wenliang Liu[1], Kevin Leahy[2], Zachary Serlin[2], Calin Belta[1]
[1]Boston University, MA, USA, [2]MIT Lincoln Laboratory, Lexington, MA, USA

Introduction

- Capability Temporal Logic plus (CaTL+) [1] is an expressive language to specify complex mission requirements for heterogeneous multi-agent systems (HMAS).
- Previous work [1] applied trajectory optimization to synthesize controllers from CaTL+ specifications, which is **computationally expensive**, **open-loop**, and **centralized**.
- We present a learning-based framework to simultaneously learn the **communication** and **distributed control** policies for a HMAS under CaTL+ specifications.
- Both policies are implemented using a novel neural network model called **CatlNet**. A **repair** algorithm is also introduced to improve training efficiency and performance.

System Model

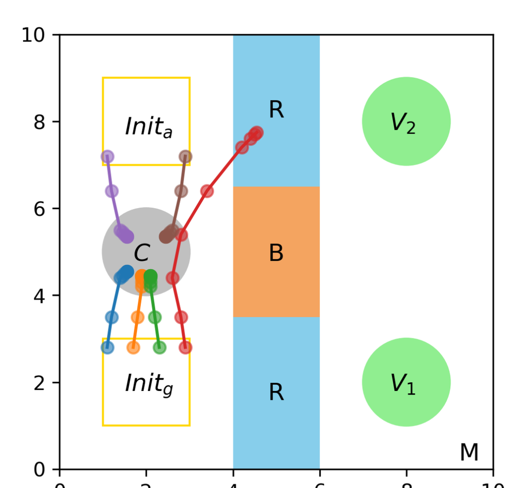
Consider a team of agents $\mathcal{J}$. Each agent $j \in \mathcal{J}$ has dynamics

$$x_j(t+1) = x_j(t) + u_j(t), \quad t = 0, 1, \ldots, H-1. \qquad (1)$$

Each agent j has a random initial state $x_j(0)$ with distribution P_j and a set of capabilities Cap_j.

Ind. trajectory: $\mathbf{x}_j = x_j(0) \ldots x_j(H)$.
Team trajectory: $\mathbf{X} = \{(\mathbf{x}_j, Cap_j)\}_{j \in \mathcal{J}}$.

Example:
- 4 UGVs: {"*Delivery*", "*Ground*"}.
- 2 UAVs: {"*Delivery*", "*Inspection*"}.



CaTL+

CaTL+, interpreted over **team trajectories**, can express rich, complex requirements. For example:

- 6 agents with capability "*Delivery*" eventually reach C;
$$\Phi_1 = \langle \mathbf{F}_{[0,8]} x \in C,\ \text{"Delivery"},\ 6 \rangle$$
- 3 agents with "*Delivery*" eventually reach V_1 and V_2;
$$\Phi_2 = \langle \mathbf{F}_{[0,25]} x \in V_1,\ \text{"Delivery"},\ 3 \rangle \wedge \langle \mathbf{F}_{[0,25]} x \in V_2,\ \text{"Delivery"},\ 3 \rangle$$
- UGVs cannot go over the bridge until 2 UAVs inspect it;
$$\Phi_3 = \neg \langle x \in B,\ \text{"Ground"},\ 1 \rangle \mathbf{U}_{[0,5]} \langle x \in B,\ \text{"Inspection"},\ 2 \rangle$$
- UGVs should always avoid entering the river R:
$$\Phi_4 = \mathbf{G}_{[0,25]} \langle \neg(x \in R),\ \text{"Ground"},\ 4 \rangle$$
- No more than 1 UGV can be on the bridge B:
$$\Phi_5 = \mathbf{G}_{[0,25]} \neg \langle x \in B,\ \text{"Ground"},\ 2 \rangle$$
- All agents should always stay in region M:
$$\Phi_6 = \mathbf{G}_{[0,25]} \langle x \in M,\ \text{"Delivery"},\ 6 \rangle$$

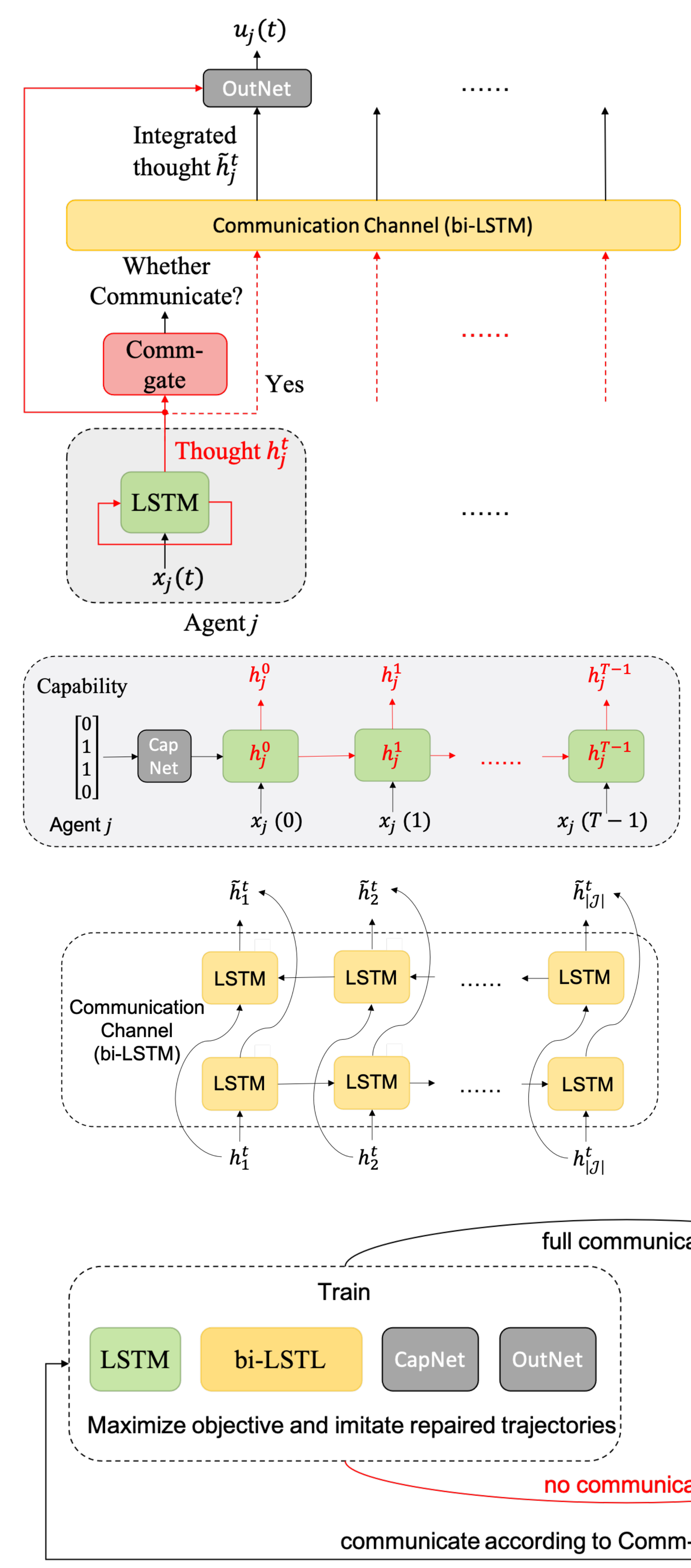
The **robustness** of CaTL+ measures how strongly a specification Φ is satisfied by $\mathbf{X}$.

Problem Formulation

- each agent can only observe its own state $x_j(t)$ at time t;
- all agents have access to a communication channel and can broadcast and receive a vector to and from the channel at each time t.

Problem: Given a HMAS and a CaTL+ specification Φ, we have **two objectives**:
- find the control policy and the communication vectors that maximize the CaTL+ robustness and minimize a cost.
- minimize the number of times that agents access the communication channel. (**When and what to communicate.**)

CatlNet Architecture and Training



Architecture:
- Agents share the same CatlNet parameters → good scalability.
- Communication and CapNet make agents behave differently.

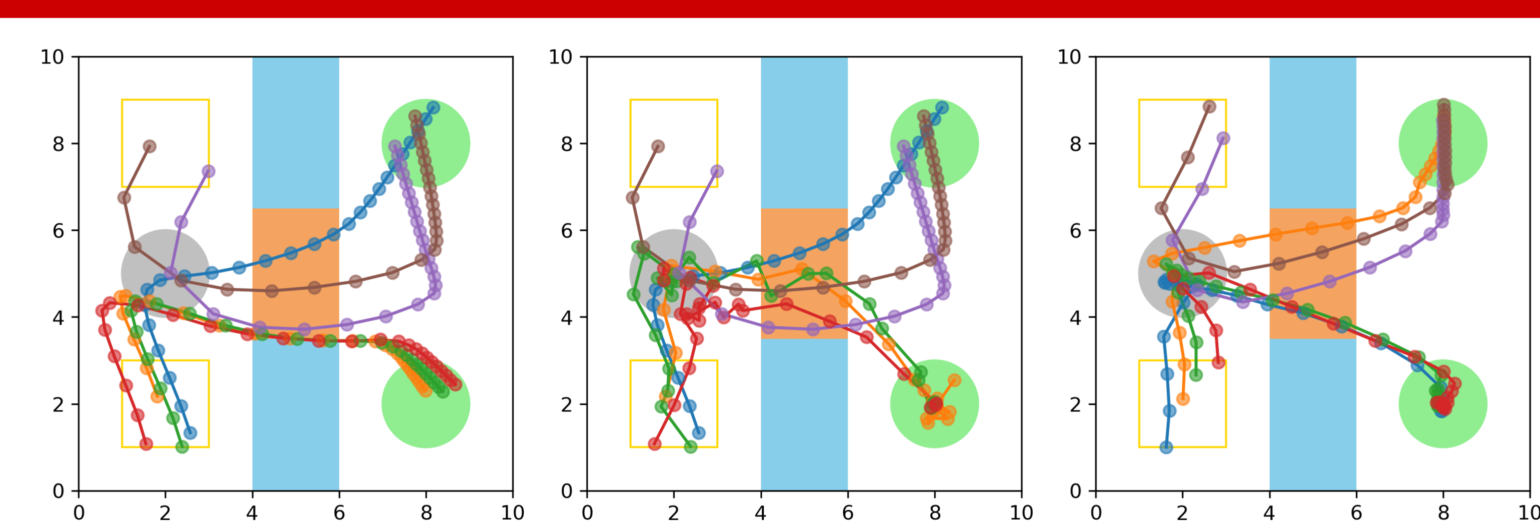
Training:
- Centralized training and decentralized execution.
- Guided by a repair algorithm.

Repair Algorithm

- Direct learning might stuck at local optima, while learning from demonstrations might fail given unseen conditions.
- Demonstrations adapted from the learner's own behavior (i.e., repair) is easier to learn.

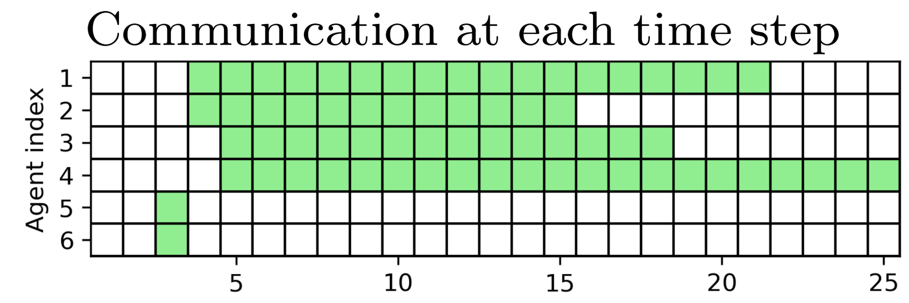
Repair trajectories given by CatlNet to satisfy CaTL+ specification. **Retrain** CatlNet to simultaneously maximize objective and imitate repaired trajectories.

Simulation Results



Before repair   After repair   Final CatlNet

Communication at each time step



- Repaired 213 trajectories.
- Tested over 10000 random initial states, 100% success rate.

References

[1] Liu, Wenliang, et al. "Robust Multi-Agent Coordination from CaTL+ Specifications." IEEE American Control Conference (ACC) 2023.