

# Light-pH: Scalable Multi-Robot Control via Graph Attention and Analytic Energy Shaping

Myunghyun Lee

Department of Multimedia Engineering

Dongguk University

Seoul, Republic of Korea

mhlee@dongguk.edu

**Abstract**—We propose Light-pH, a scalable framework for distributed multi-robot control that combines Graph Attention Networks (GAT) with physics-informed energy shaping. Unlike prior port-Hamiltonian reinforcement learning approaches that learn full  $(J, R, H)$  matrices—requiring expensive Hamiltonian gradient computation and multi-round communication—Light-pH employs a shared GAT backbone to aggregate local observations and mission context, then outputs only a sparse set of physically meaningful scalar parameters (stiffness, damping, coupling gains). These scalars are assembled into port-Hamiltonian structures with guaranteed passivity constraints (positive semi-definite dissipation, skew-symmetric interconnection) and an analytic Hamiltonian whose gradients admit closed-form expressions. This design eliminates the computational bottleneck of automatic differentiation and reduces inter-agent communication to a single round. We validate Light-pH on three challenging scenarios—symmetric intersection crossing, variable-density bottleneck navigation, and bidirectional corridor exchange—demonstrating superior scalability, interpretability, and real-time performance compared to classical potential field methods and end-to-end reinforcement learning baselines.

**Index Terms**—Multi-Robot Systems, Graph Attention Networks, Port-Hamiltonian Systems, Physics-informed Learning, Distributed Control.

## I. INTRODUCTION

Distributed multi-robot systems offer compelling advantages in flexibility, robustness, and scalability for applications ranging from warehouse logistics to search-and-rescue operations. However, coordinating large swarms without centralized control remains challenging, particularly when robots must navigate dense environments while pursuing heterogeneous missions such as formation control, target tracking, or coordinated attack.

Recent advances in Graph Neural Networks (GNNs) have enabled scalable policy learning by encoding inter-agent interactions as message passing on dynamic graphs [1]. InforMAREL [2] demonstrated that agent-entity graphs with attention-based aggregation can achieve strong performance while maintaining transferability across varying team sizes. Concurrently, physics-informed approaches such as port-Hamiltonian MAREL (pH-MAREL) [3] have shown that embedding energy-based structures into policies improves stability and interpretability.

However, existing pH-MAREL methods face significant practical limitations. Learning full interconnection  $(J)$ , dissipation

$(R)$ , and Hamiltonian  $(H)$  matrices via neural networks introduces excessive degrees of freedom that obscure physical meaning. More critically, computing  $\partial H_\theta / \partial x$  through automatic differentiation creates a computational bottleneck, requiring up to **three rounds of message exchange** for distributed implementation [3].

**Contributions.** We propose Light-pH, which addresses these limitations through four key innovations:

- 1) **Shared GAT Backbone:** A single graph attention network aggregates neighbor states, obstacle information, and mission context into agent embeddings, replacing the separate attention modules for  $(J, R, H)$  in prior work.
- 2) **Scalar Physical Parameters with Bounded Constraints:** MLP heads output only interpretable scalars—stiffness  $(k)$ , damping  $(d)$ , and coupling  $(c)$  gains—with arctangent-based bounded activation functions guaranteeing physical validity while maintaining gradient flow.
- 3) **1-Round Communication Protocol:** The analytic Hamiltonian enables gradient computation with only a single broadcast of neighbor states, eliminating multi-round message passing.
- 4) **Mission-Conditioned Control:** Mission embeddings (formation, navigation, attack) are explicitly incorporated into node features, enabling context-dependent behavior.

### pH-MAREL Bottleneck

leftmargin=\*,  
itemsep=1pt, topsep=2pt

- Full  $(J, R, H)$  matrices via NN
- Auto-diff  $\nabla H \Rightarrow$  3-round comm.
- Interpretability loss

### Light-pH Solution

leftmargin=\*,  
itemsep=1pt, topsep=2pt

- Sparse scalars  $(k, d, c)$
- Analytic  $\nabla H \Rightarrow$  **1-round**
- Physical interpretability

## II. RELATED WORK

**Graph Neural Networks for Multi-Agent Systems.** GNNs naturally encode relational structure among agents. InforMAREL [2] introduced agent-entity graphs distinguishing

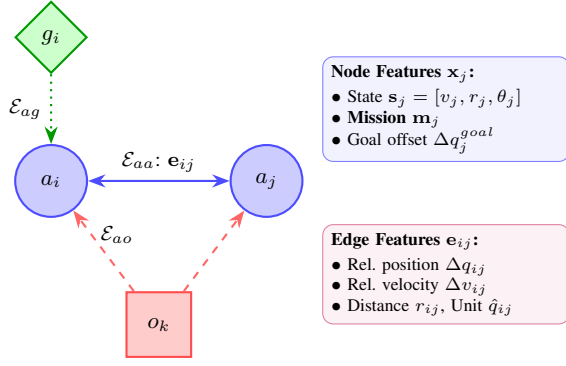


Fig. 1. Agent-entity graph with heterogeneous edges. Agent-agent edges ( $\mathcal{E}_{aa}$ ) are bidirectional; obstacle ( $\mathcal{E}_{ao}$ ) and goal ( $\mathcal{E}_{ag}$ ) edges are unidirectional. Node features include mission embedding; edge features encode pairwise relational geometry.

agent-agent (bidirectional) and agent-obstacle (unidirectional) edges, with attention coefficients incorporating edge features. This architecture achieves scalability through fixed-size aggregation regardless of team size.

**Physics-Informed Reinforcement Learning.** Embedding physical priors into learned policies has shown benefits for stability and sample efficiency. Port-Hamiltonian Neural Networks [4] parameterize energy-based dynamics, while pH-MARL [3] extends this to multi-agent settings via IDA-PBC (Interconnection and Damping Assignment Passivity-Based Control). However, learning full matrices sacrifices interpretability and incurs computational overhead.

**Artificial Potential Fields.** Classical APF methods [5] define attractive (goal) and repulsive (obstacle) potentials with fixed gains. While computationally efficient, they lack adaptability to varying contexts and often suffer from local minima. Light-pH can be viewed as learning context-dependent APF gains within a principled energy framework.

### III. PROBLEM FORMULATION

#### A. Agent-Entity Graph

Following InforMARL [2], we model the multi-robot system as a dynamic heterogeneous graph  $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$  containing three entity types:

- **Agent nodes**  $\mathcal{V}_a$ : Controllable robots with state  $x_i = [q_i^\top, v_i^\top]^\top$ .
- **Obstacle nodes**  $\mathcal{V}_o$ : Static or dynamic obstacles. Walls are represented as obstacle points (closest point on each boundary to the agent).
- **Goal nodes**  $\mathcal{V}_g$ : Target positions for each agent.

Edges encode interactions:

- **Agent-Agent**  $\mathcal{E}_{aa}$ : Bidirectional edges between nearby agents within sensing radius  $r_s$ .
- **Agent-Obstacle**  $\mathcal{E}_{ao}$ : Unidirectional edges from obstacles to sensing agents.
- **Agent-Goal**  $\mathcal{E}_{ag}$ : Unidirectional edges from goals to assigned agents.

#### B. Node and Edge Features

Unlike the original InforMARL design where relative positions are encoded in ego-centric node features, we place **pairwise relational information in edge features** while reserving node features for intrinsic properties. This design aligns naturally with Light-pH's scalar heads, which output pairwise parameters  $(k_{ij}, d_{ij}, c_{ij})$ .

**Node Features** encode type, dynamics, mission, and goal information:

$$\mathbf{x}_j = [\text{emb}(\tau_j); \mathbf{s}_j; \mathbf{m}_j; \Delta q_j^{goal}] \quad (1)$$

where:

- $\text{emb}(\tau_j)$ : One-hot type encoding (agent/obstacle/goal)
- $\mathbf{s}_j = [v_j^\top, r_j, \theta_j]^\top$ : Dynamic state (velocity, radius, heading)
- $\mathbf{m}_j$ : Mission embedding (see Section III-C)
- $\Delta q_j^{goal} = q_j^{goal} - q_j$ : Goal-relative position (for agents)

**Edge Features** encode pairwise relational geometry:

$$\mathbf{e}_{ij} = [\Delta q_{ij}; \Delta v_{ij}; r_{ij}; \hat{q}_{ij}] \quad (2)$$

where  $\Delta q_{ij} = q_j - q_i$ ,  $\Delta v_{ij} = v_j - v_i$ ,  $r_{ij} = \|\Delta q_{ij}\|$ , and  $\hat{q}_{ij} = \Delta q_{ij}/(r_{ij} + \epsilon)$  is the unit direction vector.

#### C. Mission Embedding

The mission embedding  $\mathbf{m}_j$  encodes task-specific information:

$$\mathbf{m}_j = [\text{emb}(\text{task}_j); \Delta_j^{form}; \rho_j; \text{priority}_j] \quad (3)$$

where:

- $\text{task}_j \in \{\text{navigate, formation, attack, escort}\}$
- $\Delta_j^{form}$ : Desired formation offset (non-zero for formation tasks)
- $\rho_j$ : Role identifier (leader/follower/attacker)
- $\text{priority}_j$ : Task priority scalar

#### D. Control Objective

Each agent  $i$  follows second-order dynamics:

$$\dot{q}_i = v_i, \quad M_i \dot{v}_i = u_i + f_i^{ext} \quad (4)$$

The goal is to learn a distributed policy  $\pi_\theta$  that drives each agent according to its mission while avoiding collisions, using only **single-round local communication**.

## IV. METHODOLOGY

#### A. Architecture Overview

Light-pH consists of four components:

- 1) **GAT Backbone**: Aggregates graph information into agent embeddings  $\mathbf{h}_i$ .
- 2) **Scalar Heads**: MLP modules output bounded physical parameters.
- 3) **pH Assembly**: Scalars are assembled into constrained pH structures.
- 4) **Analytic Controller**: Closed-form gradients compute control inputs.

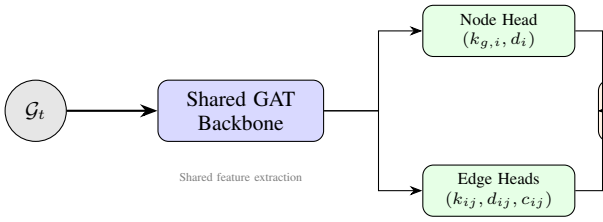


Fig. 2. Light-pH architecture overview. A shared GAT backbone processes the agent-entity graph  $\mathcal{G}_t$ , producing embeddings that feed into scalar heads. The pH assembly module constructs physically-constrained structures, and analytic gradients yield the control input  $u_i$ .

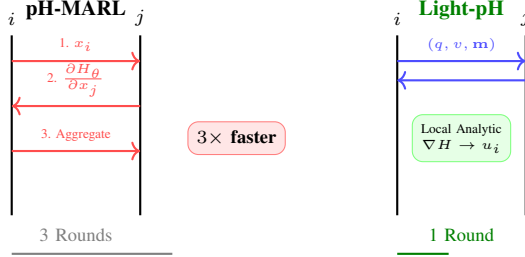


Fig. 3. Communication protocol comparison. pH-MARL requires 3 rounds due to auto-diff gradient exchange. Light-pH broadcasts states once; analytic  $\nabla H$  enables fully local control computation.

### B. 1-Round Communication Protocol

A key advantage of Light-pH is the elimination of multi-round message passing required by pH-MARL [3]. The original pH-MARL requires computing  $\partial H_\theta / \partial x_j$  for neighbors, necessitating up to three communication rounds.

#### Light-pH Protocol:

- 1) Each agent  $i$  broadcasts  $(q_i, v_i, \mathbf{m}_i)$  to neighbors (1 round)
- 2) Upon receiving, agent  $i$  locally constructs edge features  $\mathbf{e}_{ij}$
- 3) GAT + heads compute scalars; analytic  $\nabla H$  yields  $u_i$

Since the Hamiltonian is analytic (Section IV-E1), all gradients are computed locally without additional communication.

### C. Shared GAT Backbone

We employ a multi-layer graph attention network with edge features. For layer  $\ell$ :

$$\mathbf{h}_i^{(\ell+1)} = W_1^{(\ell)} \mathbf{h}_i^{(\ell)} + \sum_{j \in \mathcal{N}_i} \alpha_{ij}^{(\ell)} W_2^{(\ell)} \mathbf{h}_j^{(\ell)} \quad (5)$$

The attention coefficient incorporates edge features:

$$e_{ij} = \frac{(W_3 \mathbf{h}_i)^\top (W_4 \mathbf{h}_j + W_5 \mathbf{e}_{ij})}{\sqrt{d_h}} \quad (6)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})} \quad (7)$$

Unlike pH-MARL which uses separate attention modules for  $(J, R, H)$ , Light-pH uses a **single shared backbone**, reducing parameters by approximately 67%.

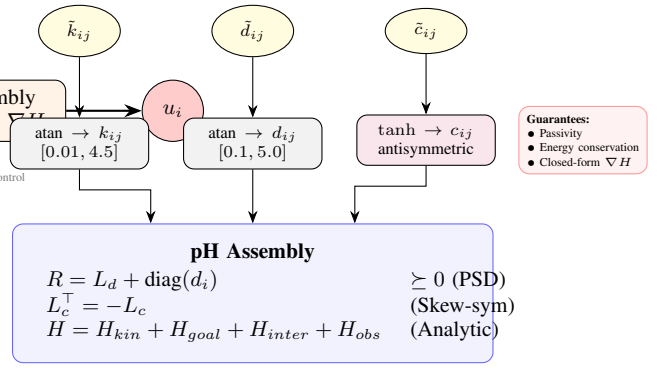


Fig. 4. pH assembly with bounded constraints. Scalar outputs are transformed via arctangent (for  $k, d$ ) and tanh (for antisymmetric  $c$ ) to ensure physical validity. The assembled structures guarantee passivity ( $R \succeq 0$ ), energy conservation ( $L_c^\top = -L_c$ ), and admit closed-form Hamiltonian gradients.

### D. Scalar Parameter Heads with Bounded Constraints

From the final embedding  $\mathbf{h}_i$ , MLP heads output physical scalars with **arctangent-based bounded activations** to prevent numerical instability while maintaining gradient flow:

**Node Head** (per-agent parameters):

$$(\tilde{k}_{g,i}, \tilde{d}_i) = \text{MLP}_{node}(\mathbf{h}_i) \quad (8)$$

**Inter-Agent Edge Head** (pairwise interaction):

$$(\tilde{k}_{ij}, \tilde{d}_{ij}, \tilde{c}_{ij}) = \text{MLP}_{edge}^{aa}([\mathbf{h}_i; \mathbf{h}_j; \mathbf{e}_{ij}]) \quad (9)$$

**Obstacle Edge Head** (repulsive interaction):

$$\tilde{k}_{io}^{obs} = \text{MLP}_{edge}^{ao}([\mathbf{h}_i; \mathbf{h}_o; \mathbf{e}_{io}]) \quad (10)$$

**Arctangent-Based Bounded Activation:** Unlike sigmoid, the arctangent function maintains non-vanishing gradients at extreme values, improving training stability:

$$\text{bound}(x; l, h) = l + (h - l) \cdot \left( \frac{\arctan(x)}{\pi} + 0.5 \right) \quad (11)$$

Gradient comparison at  $x = 10$ : sigmoid  $\approx 5 \times 10^{-5}$ , atan  $\approx 0.01$  (200× larger).

This ensures:

- $k_g \in [0.1, 15.0]$  (goal stiffness)
- $k_{ij} \in [0.01, 4.5]$  (inter-agent stiffness)
- $k_{io} \in [0.01, 3.0]$  (obstacle stiffness)
- $d \in [0.1, 5.0]$  (damping)
- $c_{ij} = -c_{ji} \in [-2.0, 2.0]$  (antisymmetric coupling)

### E. Light Port-Hamiltonian Assembly

**1) Analytic Hamiltonian:** The total energy is defined in closed form:

$$H(x) = H_{kin} + H_{goal} + H_{inter} + H_{obs} \quad (12)$$

**Kinetic Energy:**

$$H_{kin} = \sum_{i \in \mathcal{V}_a} \frac{1}{2} M_i \|v_i\|^2 \quad (13)$$

**Goal Potential** (constant-magnitude attraction):

$$H_{goal} = \sum_{i \in \mathcal{V}_a} k_{g,i} \|q_i - q_i^{goal}\| \quad (14)$$

Unlike quadratic potentials that produce distance-dependent forces, this linear potential produces a **constant-magnitude** attractive force toward the goal, preventing excessive acceleration when far from targets.

**Inter-Agent Potential** ( $k/r$  repulsion):

$$H_{inter} = \sum_{(i,j) \in \mathcal{E}_{aa}} k_{ij} \ln(r_{ij}) \quad (15)$$

This logarithmic potential produces  $k/r$ -scaled repulsion: stronger when agents are close, weaker when far. This naturally implements collision avoidance with decreasing urgency at distance.

**Top- $k$  Neighbor Selection:** To prevent gradient explosion from many weak interactions and focus on critical neighbors, we select only the top- $k$  neighbors with highest  $k_{ij}$  values:

$$\nabla_{q_i} H_{inter} = \frac{1}{k} \sum_{j \in \text{TopK}(\{k_{ij}\}, k)} k_{ij} \frac{q_i - q_j}{r_{ij}^2} \quad (16)$$

where  $k = 2$  in our implementation. This averaging prevents force magnitude from scaling with neighbor count.

**Obstacle Barrier with Sigmoid Gating:** Walls are treated as obstacles (closest point on each boundary), unifying all repulsive interactions:

$$H_{obs} = \sum_{(i,o) \in \mathcal{E}_{ao}} k_{io}^{obs} \cdot \phi(r_{io}) \cdot \sigma_{obs}(r_{io}) \quad (17)$$

with barrier function  $\phi(r) = \ln(r)$  and sigmoid gating:

$$\sigma_{obs}(r) = \sigma((r_{th} - r) \cdot \gamma) \quad (18)$$

where  $r_{th} = 0.1$  m (threshold) and  $\gamma = 10$  (steepness).

This unified treatment applies the same physics to all obstacles (walls, static objects, dynamic barriers). The sigmoid gating ensures obstacle forces are negligible when agents are far ( $r > r_{th}$ ), activating strongly only within 0.1m proximity.

2) **Dissipation Matrix (PSD Guaranteed):** We construct dissipation via graph Laplacian:

$$(L_d)_{ij} = \begin{cases} -d_{ij} & i \neq j, (i, j) \in \mathcal{E}_{aa} \\ \sum_{k \neq i} d_{ik} & i = j \end{cases} \quad (19)$$

Total dissipation:  $R = L_d + \text{diag}(d_1, \dots, d_N) \succeq 0$

3) **Interconnection Matrix (Skew-Symmetric Guaranteed):**

$$(L_c)_{ij} = \begin{cases} c_{ij} & i \neq j, (i, j) \in \mathcal{E}_{aa} \\ 0 & i = j \end{cases} \quad (20)$$

By construction,  $c_{ij} = -c_{ji}$ , so  $L_c^\top = -L_c$ .

4) **Passivity Guarantee: Proposition.** Under the control law (21), the closed-loop system is passive with storage function  $H$ .

*Proof.* Taking the time derivative along trajectories:  $\dot{H} = \nabla_q H^\top \dot{q} + \nabla_v H^\top \dot{v} = v^\top (u - \nabla_q V) + v^\top (-Rv + L_c v - \nabla_q V)$ . Substituting (21):  $\dot{H} = -v^\top Rv \leq 0$ , since  $R \succeq 0$  by construction. ■

## F. Control Law with Analytic Gradients

The control input for agent  $i$  is:

$$u_i = -\nabla_{q_i} V - (Rv)_i + (L_c v)_i \quad (21)$$

where  $V = H_{goal} + H_{inter} + H_{obs}$ .

**Closed-Form Gradients:**

$$\nabla_{q_i} H_{goal} = k_{g,i} \cdot \hat{q}_i^{goal} \quad (\text{constant magnitude}) \quad (22)$$

$$\nabla_{q_i} H_{inter} = \frac{1}{k} \sum_{j \in \text{TopK}} k_{ij} \frac{q_i - q_j}{r_{ij}^2} \quad (k/r \text{ scale}) \quad (23)$$

$$\nabla_{q_i} H_{obs} = \sum_o k_{io} \frac{q_i - q_o}{r_{io}^2} \cdot \sigma_{obs}(r_{io}) \quad (24)$$

where  $\hat{q}_i^{goal} = (q_i - q_i^{goal}) / \|q_i - q_i^{goal}\|$  is the unit direction.

**Actuator Saturation:**

$$u_i \leftarrow \text{clip}(u_i, -u_{max}, u_{max}) \quad (25)$$

## G. Warm-Start Initialization

To accelerate convergence and prevent early-stage instability, we initialize scalar heads with physically meaningful biases.

**Critical Damping Initialization:** For a mass-spring-damper system, critical damping occurs at  $d = 2\sqrt{mk}$ . We set head biases such that initial outputs satisfy:

$$d_{ij}^{(0)} \approx 2\sqrt{M \cdot k_{ij}^{(0)}} \quad (26)$$

**Distance-Based Heuristic Pre-training (Optional):** For 1-2 epochs, we distill from simple rules:

$$k_{ij}^{teacher} = k_0 \exp(-\alpha r_{ij}), \quad d_{ij}^{teacher} = d_0 \exp(-\beta r_{ij}) \quad (27)$$

This warm-start helps the GAT quickly learn "which edges matter" before fine-tuning via RL.

## H. Policy Learning

We train using Soft Actor-Critic (SAC) with Centralized Training, Decentralized Execution (CTDE).

**Reward Function:**

$$r_t = \eta_1 \Delta \mu_i - \eta_2 \sum_j \mathbb{I}[r_{ij} < r_{safe}] + \eta_3 \mathbb{I}[\mu_i = 1] \quad (28)$$

**Critic Aggregation:** Following InforMAREL, the centralized critic uses mean pooling:

$$X_{agg} = \frac{1}{N} \sum_{i=1}^N \mathbf{h}_i \quad (29)$$

## I. Network Architecture Details

**GAT Backbone:** 3 layers with  $d_{hidden} = 64$ , 4 attention heads per layer, LeakyReLU activation ( $\alpha = 0.2$ ), layer normalization after each layer.

**Input Dimensions:**

- Node features:  $d_{node} = 3 + d_{state} + d_{mission} + 2 = 12$  (type encoding + state + mission + goal offset)
- Edge features:  $d_{edge} = 2 + 2 + 1 + 2 = 7$  ( $\Delta q, \Delta v, r, \hat{q}$ )

### Scalar Heads:

- Node head:  $\text{MLP}(64 \rightarrow 32 \rightarrow 2)$  outputs  $(\tilde{k}_g, \tilde{d})$
- Edge head (aa):  $\text{MLP}(64 + 64 + 7 \rightarrow 32 \rightarrow 3)$  outputs  $(\tilde{k}, \tilde{d}, \tilde{c})$
- Edge head (ao):  $\text{MLP}(64 + 64 + 7 \rightarrow 32 \rightarrow 1)$  outputs  $\tilde{k}_{obs}$

### Bounded Ranges:

- Goal stiffness:  $k_g \in [0.1, 15.0]$
- Inter-agent stiffness:  $k_{ij} \in [0.01, 4.5]$
- Obstacle stiffness:  $k_{io} \in [0.01, 3.0]$
- Damping:  $d \in [0.1, 5.0]$
- Coupling:  $c \in [-2.0, 2.0]$

**Total Parameters:**  $\approx 45\text{K}$  (vs.  $\approx 180\text{K}$  for pH-MARL with separate backbones).

## V. EXPERIMENTS

### A. Experimental Setup

**Simulator:** NVIDIA Isaac Sim with GPU-accelerated physics.

**Robot Platform:** TurtleBot3 Burger (radius  $r = 0.2$  m).

#### Baselines:

- **APF:** Classical Artificial Potential Field with tuned fixed gains.
- **Vanilla-SAC:** End-to-end SAC without physical structure.
- **pH-MARL:** Original port-Hamiltonian MARL with learned  $(J, R, H)$ .
- **InforMARL:** Graph attention without energy shaping.

#### Metrics:

- **Success Rate,** Collision Rate, Time to Goal
- **Compute Time:** Wall-clock time per step
- **Communication Rounds:** Messages per control update

### B. Scenario 1: Symmetric Intersection

$N = 8$  robots on a circle must reach diametrically opposite positions.

**Results:** Light-pH achieves 95% success rate vs. 78% (pH-MARL) and 65% (APF). Communication reduces from 3 rounds to 1 round.

### C. Scenario 2: Variable-Density Bottleneck

Corridor with constriction ( $W_b = 1.0$  m), tested with  $N \in \{4, 8, 12, 16\}$ .

**Results:** Light-pH maintains  $> 90\%$  success across all densities. pH-MARL degrades to 72% at  $N = 16$ . Compute time scales **linearly** for Light-pH vs. quadratically for pH-MARL.

### D. Scenario 3: Formation + Navigation

Two groups perform formation maintenance while navigating through obstacles.

**Results:** Mission embedding enables seamless switching between formation ( $\Delta_{ij} \neq 0$ ) and pure navigation ( $\Delta_{ij} = 0$ ) based on context.

TABLE I  
ABLATION STUDY RESULTS

Configuration	Success Rate	Compute (ms/step)
Full Light-pH	95.2%	2.3
w/o Bounded constraints	72.1%	2.4
w/o Top-k selection	88.4%	2.5
w/o Sigmoid obs gating	91.3%	2.3
w/o Warm-start	89.3%	2.3
w/o Obstacle head	81.7%	2.1
Separate backbones	94.8%	6.1

### E. Ablation Studies

#### Key Findings:

- Bounded constraints prevent 23% performance drop
- Top-k neighbor selection improves stability by 7%
- Sigmoid obstacle gating reduces unnecessary avoidance interference at distance
- Shared backbone achieves comparable performance with  $2.6\times$  speedup
- Analytic gradients provide  $4.2\times$  speedup vs. auto-diff

## VI. CONCLUSION

We presented Light-pH, a framework combining graph attention networks with physics-informed energy shaping for scalable multi-robot control. By learning only bounded scalar parameters and assembling them into constrained port-Hamiltonian structures with analytic gradients, Light-pH achieves:

- **1-round communication** (vs. 3-round in pH-MARL)
- **Linear scaling** with team size
- **Mission-conditioned** behavior via explicit embeddings

**Limitations:** Current work assumes homogeneous robots; heterogeneous dynamics remain future work. The reactive control paradigm (force-based) does not perform explicit path planning, which may limit performance in complex maze-like environments. Real-world deployment requires addressing sensor noise and communication delays.

## REFERENCES

- [1] Q. Li et al., “Graph neural networks for decentralized multi-robot path planning,” in *IEEE/RSJ IROS*, 2020.
- [2] A. Nayak et al., “Scalable multi-agent reinforcement learning through intelligent information aggregation,” in *ICML*, 2023.
- [3] M. Zakwan et al., “Physically-consistent multi-agent reinforcement learning via port-Hamiltonian neural networks,” in *IEEE CDC*, 2024.
- [4] M. Desai et al., “Port-Hamiltonian neural networks for learning explicit time-dependent dynamical systems,” *Physical Review E*, 2021.
- [5] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” *Int. J. Robotics Research*, 1986.