Spatial Temporal Graph Neural Networks for Decentralized Control of Robot Swarms (Demo Paper)

Siji Chen
Virginia Tech
Falls Church, VA, USA
sijic@vt.edu

Yanshen Sun
Virginia Tech
Falls Church, VA, USA
yansh93@vt.edu

Peihan Li
Drexel University
Philadelphia, PA, USA
pl525@drexel.edu

Lifeng Zhou
Drexel University
Philadelphia, PA, USA
lz457@drexel.edu

Chang-Tien Lu
Virginia Tech
Falls Church, VA, USA
ctlu@vt.edu

ABSTRACT

Recent research has explored the use of graph neural networks (GNNs) for decentralized control in swarm robotics. However, it has been observed that relying solely on local states is insufficient to imitate a centralized control policy. To address this limitation, previous studies proposed incorporating K-hop delayed states into the computation. While this approach shows promise, it can lead to a lack of consensus among distant flock members and the formation of small localized groups, ultimately resulting in task failure. Our approach is to include the delayed states to build a spatiotemporal GNN model (ST-GNN) by two levels of expansion: spatial expansion and temporal expansion. The spatial expansion utilizes K-hop delayed states to broaden the network while temporal expansion, can effectively predict the trend of swarm behavior, making it more robust against local noise. To validate the effectiveness of our approach, we conducted simulations in two distinct scenarios: free flocking and flocking with a leader. In both scenarios, the simulation results demonstrated that our decentralized ST-GNN approach successfully overcomes the limitations of local controllers. We performed a comprehensive analysis on the effectiveness of spatial expansions and temporal expansions independently. The results clearly demonstrate that both significantly improve overall performance. Furthermore, when combined, they achieve the best performance compared to global solution and delayed states solutions. The performance of ST-GNN underscores its potential as an effective and reliable approach for achieving cohesive flocking behavior while ensuring safety and maintaining desired swarm characteristics.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; • **Computer systems organization → Robotics**; • **Computing methodologies → Modeling and simulation**.

KEYWORDS

Spatial Temporal Graph Neural Network, Robot Swarms, Flocking, Imitation learning, Decentralize Control

ACM Reference Format:
Siji Chen, Yanshen Sun, Peihan Li, Lifeng Zhou, and Chang-Tien Lu. 2023. Spatial Temporal Graph Neural Networks for Decentralized Control of Robot Swarms (Demo Paper). In *The 31st ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '23), November 13–16, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3589132.3625630

1 INTRODUCTION

Flocking is a collective behavior observed in groups of agents, such as birds, fish, or artificial agents, where they move together in a coordinated manner. In a flock, each agent follows simple rules based on local information to achieve a common group objective [6]. Multi-agent systems based on flocking models exhibit self-organization and goal-directed behavior, making them suitable for various applications, including automated parallel delivery, sensor network design, and search and rescue operations. Flocking is typically modeled as a consensus or alignment problem, aiming to ensure that all agents in the group eventually agree on their states [5]. Classical methods such as those proposed by Tanner [8] and Olfati-Saber [5], define rules and constraints governing the position, speed, and acceleration of the agents. However, these methods heavily rely on parameter tuning and are limited to predefined scenarios. In contrast, learning-based methods spontaneously explore complex patterns and adapt their parameters through training, providing more flexibility and adaptability compared to classical approaches.

There are primarily two research directions in learning-based methods. One approach focuses on imitation learning, as demonstrated by Tolstaya et al. [9], Kortvelesy et al. [3], Zhou et al. [12], and Lee et al. [4]. The other approach involves multiagent deep reinforcement learning (MADRL) techniques, as explored in the works of Yan et al. [11] and Xiao et al. [10]. MADRL is particularly useful when labels are unavailable, but it presents its own set of challenges. In this work, we choose to utilize imitation learning due to the availability of an expert policy that has proven to be effective for our task [7–9].

Recent research in this field adopts a graph-based approach to represent flocks and leverages Graph Neural Networks (GNNs) [1, 2] for modeling and analyzing flock dynamics. This approach shows

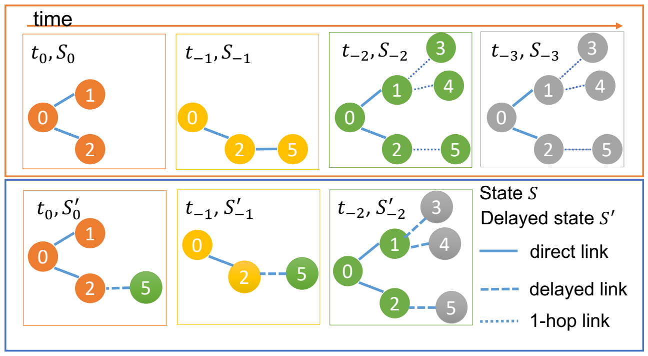

Figure 1: Comparison between the delayed state method and ST-GNN. The top row illustrates time-varying graphs, where t_i represents the timestamp and S_i denotes the corresponding state at t_i, depicted by different colors. The bottom row showcases the results of the 3-hop delayed state method at each t_i, with a focus on node 0 (V_0). At t_0, V_0 is directly connected to V_1 and V_2, but the link between V_0 and V_1 breaks at t_{-1}. Consequently, in delayed state S'_0, V_0 cannot access the states of V_3 and V_4 through V_1 due to the broken link. Similarly, at t_{-2}, we obtain S'_{-2} since V_0 is 1-hop away from V_3, V_4, through V_1. The delayed state method only utilize $\{S'_0\}$, while ST-GNN integrates the sequence of $\{S'_0, S'_{-1}, S'_{-2}\}$, providing a more comprehensive set of information.

promise in addressing complex flocking tasks by harnessing the power of graph-based representations and neural networks. In addition, studies such as Tolstaya et al. [9], Kortvelesy et al. [3], Zhou et al. [12], and Lee et al. [4] utilize a technique called "delayed state" to incorporate the information from the k-step-before states of a robot's k-hop neighbors, where $k = 1, 2, 3, \ldots$ [9]. This method enables the learning of spatially extended representations in the local network. However, this approach overlooks the influence of a robot's historical states and the historical states of its neighbors, thereby neglecting the temporal sequence of swarm movement.

We demonstrate the limitation of the delayed state approach in Figure 1. If we rely solely on the delayed state method, the target node V_0 can only use delayed states of S'_0 and is unable to access the previous states of V_3, V_4 as illustrated in S'_{-2}. With the ST-GNN approach, V_0 can leverage the history of delayed states S'_0, S'_{-1}, S'_{-2} to align its movement with V_3 and V_4, thereby incorporating both spatial and temporal information. By incorporating the temporal relations of robots, ST-GNN enables a more comprehensive understanding of the system dynamics, leading to improved coordination and performance. With this insight, we make the following primary contributions in this work:

Proposing a novel spatiotemporal imitation learning model for decentralized swarm control: ST-GNN (Spatio-Temporal Graph Neural Network) integrates both current and historical robot states, along with neighbor information, to enable effective action control. To the best of our knowledge, we are the first designed a ST-GNN model capable of achieving key objectives simultaneously, such as flocking, the leader following, with performance comparable to centralized solutions.

Performing extensive experiments on two tasks: In our evaluation, we conducted a comparative analysis of ST-GNN against delayed states GNN and other centralized solutions. We considered various metrics, including MAE, minimal distance of the swarm, velocity alignment, and distance to the leader. The results clearly demonstrate the advantages of ST-GNN in achieving successful outcomes in both flocking and leader following tasks.

Providing a flexible configurable environment to adapt to real-world scenarios: In real-world scenarios, each device has its own capacity limitations in terms of communication range, maximum velocity, and maximum acceleration. We have made these parameters configurable, allowing for adaptability and flexibility.

Conducting comprehensive analysis on ST-GNN expansion settings: We performed a thorough analysis to evaluate the effectiveness of spatial expansions, temporal expansions, and their combination in ST-GNN to gain insights into these expansion settings' impact on the model's performance.

2 METHOD

In this section, we will cover problem formulation, the expert policy, i.e., a centralized solution, and our proposed decentralized model using spatiotemporal graph neural networks.

Consider a scenario where we have a collection of N robots situated in a two-dimensional plane. We assume that all robots are identical, possessing the same capabilities, such as a maximum acceleration $U_{\max}$, a maximum velocity $V_{\max}$, and a communication range R_c. Each robot i can be uniquely identified by its position $\mathbf{p}_i$ and its velocity $\mathbf{v}_i$. The objective of our model is to estimate the next acceleration command $\mathbf{u}_i$ for each robot, based on the current state by itself and its neighbors, represented by the position and velocity $(\mathbf{p}, \mathbf{v})$. In our analysis, we consider each robot i as a vertex denoted by V_i in the network $\mathcal{G}$. An edge (i, j) exists between two robots if the distance r_{ij} between them is less than R_c. This connectivity can be represented using an adjacency matrix $A \in \mathbb{R}^{N \times N}$, where an element A_{ij} is non-zero if and only if the edge (i, j) is present in the network $\mathcal{G}$.

For the expert policy, we adopted Tanner's solution [8], which incorporates two crucial components: collision avoidance and velocity alignment among the swarm. Tanner's expert policy generates the acceleration command u by combining the collision avoidance potential (as defined in Equation 1) with the velocity agreement within the swarm (as the first term defined in Equation 2). The resulting control u_i as defined in Equation 2, is a centralized solution that takes into account the velocity differences among all robots and the local collision potential.

$$U_{i,j} = \frac{1}{r_{i,j}^2} + log||r_{i,j}||^2, ||r_{i,j}|| \leq \rho. \tag{1}$$

$$u_i = -\sum_{j=1}^{N}(v_i - v_j) - \sum_{j=1}^{N}(\nabla_{r_{i,j}} U(i,j)). \tag{2}$$

To design our decentralized model using only local information, we implement two level expansions. First we incorporate spatial expansion by merging in K-hop delayed state X_k, where X_k represents the delayed state at a specific time step in the past (e.g., $t = -k$). We utilize the adjacency matrix A_k, which captures the connectivity at time step $t = -k$ to merge in the delayed state (as defined in

Equation 3). Next we pass all the delayed states X_k through the first Long Short-Term Memory (LSTM) layer. The LSTM network allows us to capture the delayed states of K-hop neighbors, expanding the spatial representation of the system. The spatially expanded embeddings are then stored sequentially and passed through the second LSTM layer for temporal expansion. The temporal expansion plays a crucial role in predicting the movement of the swarm, overcoming the limitations imposed by relying solely on local information. This process is illustrated in Figure 2.

$$X_K = (\prod_{i=0}^{K} A_i) X_k, k = 1, 2, ..., K. \tag{3}$$

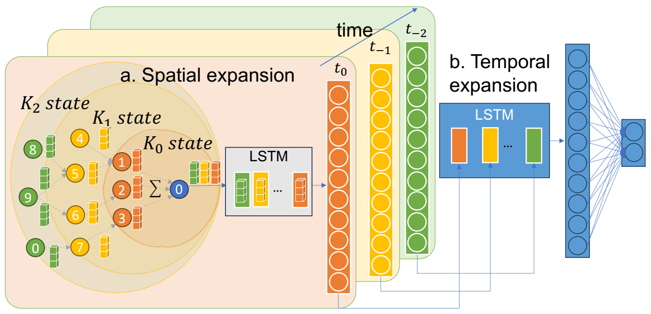

Figure 2: ST-GNN model spatial temporal expansion.

3 SIMULATION

In this section, we discuss the simulation settings and present the results of our study. Our model aims to generate the estimated acceleration command $\hat{u}$. To train our model, we utilized the L2 loss function along with the Adam optimizer. The architecture of our model includes a single SAGEConv layer with a parameter size of 16, which is used to aggregate neighborhood states. The aggregated states are then used to calculate delayed states using Equation 3. Subsequently, all K-delayed states are passed through the first LSTM layer with a hidden layer size of 16 to achieve spatial expansion. The output is then fed into the second LSTM layer with a hidden layer size of 16 to complete the temporal expansion. Finally, the last output from the second LSTM is extracted and passed through a two-layer MLP to generate $\hat{u}$.

We train models in the free flocking environment with $N = 30$. The swarm size is chosen to ensure spatial expansion could encompass a significant number of robots. The parameters are set as follows: $R_c = 7$, sample period $T_s = 0.01$, $V_{max} = 10$, $U_{max} = 50$, random initial positions uniformly distributed in $[0, R_c * \sqrt{N}]$, and random initial velocities uniformly distributed in $[-10, 10]$. To ensure a valid initial configuration, each robot must have at least two neighbors, and the minimum distance between any two robots must be greater than 0.5.

3.1 Compared Algorithms

We conducted a comprehensive comparison of our proposed method, ST-GNN, across various spatial (K) and temporal (L) expansion sizes, ranging from 1 to 5. Additionally, we combined spatial and temporal expansions with (K, L) set as $(2, 2)$, $(3, 3)$, and $(5, 5)$. To provide a comprehensive evaluation, we compared ST-GNN with Delayed Aggregation Graph Neural Networks (D-GNN) [9]. We also explored a global variant of ST-GNN (G-STGNN), where R_c was set to infinity. In addition to Mean Absolute Error (**MAE**), we use three other metrics to evaluate the performance. Minimal Distance of swarm (**D$_{\mathbf{min}}$**) which is the mean of minimal distance between any two robots during the episode. A large value indicates a failure in swarm formation while a small value indicates higher collision risk. **Close to expert policy or larger value indicates better performance.** Velocity alignment (**V**) which is the mean of velocity variance of the swarm during the episode. In the leader following scenario, the variance is computed based on the leader. **The lower value indicates better performance.** Distance to Leader (τ) is used in the leader following scenario, and measures the mean distance from any robot to the leader. **The lower value indicates better performance.**

3.2 Experiment Results

We evaluate model's performance in new, unseen test environments, where the swarm's next states are determined by the model's predictions $\hat{u}$.

Model	MAE	V	D_{min}
Expert	–	0.17±0.00	4.75±0.13
Global GNN	2.46±0.17	0.16±0.00	4.62±0.12
D-GNN(K=1)	12.79±2.65	2.46±2.48	4.60±0.18
D-GNN(K=2)	12.04±0.85	0.59±0.32	3.09±0.12
D-GNN(K=3)	4.71±0.57	0.26±0.28	3.88±0.12
D-GNN(K=5)	**4.54±0.45**	**0.18±0.02**	**3.97±0.11**
ST-GNN(K=1,L=1)	5.59±0.34	0.20±0.02	4.86±0.08
ST-GNN(K=2,L=1)	2.54±0.31	0.17±0.02	**4.76±0.09**
ST-GNN(K=3,L=1)	**2.29±0.23**	**0.17±0.01**	4.69±0.14
ST-GNN(K=5,L=1)	2.58±0.27	0.19±0.02	4.77±0.12
ST-GNN(K=1,L=2)	5.22±0.49	**0.16±0.03**	**4.80±0.06**
ST-GNN(K=1,L=3)	5.05±0.37	0.18±0.02	4.68±0.09
ST-GNN(K=1,L=5)	**4.25±0.43**	0.18±0.03	4.78±0.06
ST-GNN(K=2,L=2)	2.26±0.35	0.20±0.02	4.75±0.15
ST-GNN(K=3,L=3)	2.07±0.18	**0.17±0.01**	4.71±0.12
ST-GNN(K=5,L=5)	**1.94±0.17**	0.18±0.02	**4.77±0.12**

Table 1: Multi-robot flocking results with $N = 30$. ST-GNN ($K = 5$, $L = 5$) achieves the best performance.

Multi-robot flocking scenario. In the first experiment, each model was tested with 10 random initializations and continued for 1000 steps. The results are summarized in Table 1. In spatial expansion, as we increased K while keeping $L = 1$, ST-GNN's performance improved as K increased up to $K = 3$, as indicated by lower MAE and velocity variance values. However, further increasing K to 5 decreased performance, highlighting the limitations of the spatial-only approach. In temporal expansion, increasing L while keeping $K = 1$ resulted in improved performance, with the best results achieved when $L = 5$, as indicated by MAE. When expanding both K and L, performance further improved. The best overall performance was observed when $K = 5$ and $L = 5$, surpassing all other GNN models. These results demonstrate the effectiveness of our approach in enhancing decentralized flocking models as a valuable complement to centralized solutions.

Leader following scenario. In the second experiment, we tested the new scenario using the same models trained in the classical flocking scenario. In this experiment, we increased the swarm size from $N = 30$ to $N = 50$ to demonstrate the models' ability to transfer to different swarm size without any changes. Following random initialization, we selected two robots as leaders and set their velocities to be the same, randomly assigned from the range of $[-5, 5]$. The velocities of the remaining robots were reset to 0. We halved the V_{max} of the leaders to allow the rest of the robots to catch up, since their initial velocity was 0. The two leaders were not controlled by $\hat{u}$ and continued moving with the same velocity throughout the experiment. The results can be found in Table 2. For spatial expansion, ST-GNN exhibited improved performance as K increased until $K = 3$, resulting in lower MAE, velocity variance, and closer proximity to the leaders. Similarly, in temporal expansion, ST-GNN demonstrated optimal performance when $L = 5$. Moreover, the best overall performance was achieved when $K = 5$ and $L = 5$. These results highlight ST-GNN's ability to transfer and adapt to different scenarios while maintaining superior performance.

Model	MAE	V	τ
Expert	–	0.24±0.01	14.63±1.95
D-GNN(K=1)	15.67±1.57	2.66±1.66	24.57±5.11
D-GNN(K=2)	15.12±1.09	1.42±0.92	23.85±7.77
D-GNN(K=3)	10.47±1.23	0.55±0.26	21.21±6.72
D-GNN(K=5)	**9.67±0.91**	**0.41±0.14**	**20.65±7.03**
ST-GNN(K=1,L=1)	14.89±0.67	0.52±0.09	17.60±2.34
ST-GNN(K=2,L=1)	11.14±1.01	0.36±0.04	16.52±2.87
ST-GNN(K=3,L=1)	**9.56±1.16**	**0.30±0.03**	**16.12±2.93**
ST-GNN(K=5,L=1)	9.87±1.46	0.32±0.03	16.31±3.21
ST-GNN(K=1,L=2)	13.23±0.39	0.47±0.05	17.25±2.57
ST-GNN(K=1,L=3)	14.30±1.95	0.49±0.07	18.89±4.98
ST-GNN(K=1,L=5)	**13.14±0.92**	**0.44±0.08**	**17.25±2.24**
ST-GNN(K=2,L=2)	11.69±0.74	0.39±0.05	16.59±2.67
ST-GNN(K=3,L=3)	9.39±1.27	0.30±0.02	16.09±2.78
ST-GNN(K=5,L=5)	**8.17±1.46**	**0.28±0.03**	**15.69±2.52**

Table 2: Leader following results with $N = 50$ and 2 leaders. ST-GNN($K = 5$, $L = 5$) achieves the best performance.

We present a qualitative result of the leader following scenario in Figure 3. The ST-GNN model, trained with parameters $K = 5$ and $L = 5$ in the classical flocking scenario with $N = 30$, is tested in the new leader following scenario with a larger swarm size of $N = 100$. This visualization provides a detailed depiction of how the ST-GNN model effectively achieves velocity alignment, collision avoidance, and leader following, resulting in the formation of a cohesive swarm.

4 CONCLUSION

We demonstrate the effectiveness of ST-GNN as a decentralized solution for swarm control. By leveraging ST-GNN, we can overcome the limitations associated with relying solely on local information.

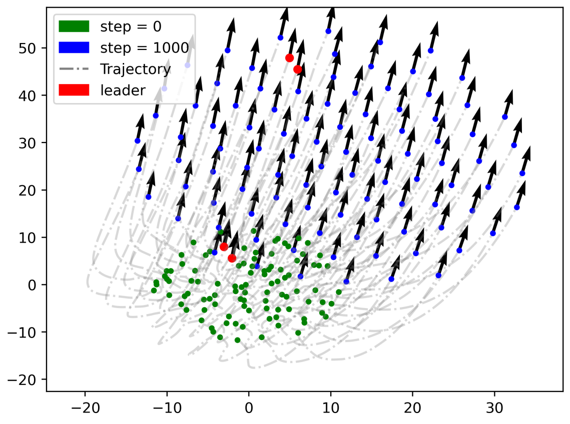

Figure 3: Leader following simulation results using ST-GNN(K=5,L=5) with $N = 100$ and 2 leaders.

This is accomplished by integrating prediction capabilities into our model, enabling it to capture and respond to global swarm dynamics. Our ST-GNN based learning model, specifically with parameters K=5 and L=5, has consistently outperformed spatial-only models, showcasing its ability to leverage both spatial and temporal information for improved performance.

REFERENCES

[1] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[2] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[3] Ryan Kortvelesy and Amanda Prorok. 2021. ModGNN: Expert policy approximation in multi-agent systems with a modular graph neural network architecture. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 9161–9167.

[4] Elijah S Lee, Lifeng Zhou, Alejandro Ribeiro, and Vijay Kumar. 2023. Graph neural networks for decentralized multi-agent perimeter defense. *Frontiers in Control Engineering* 4 (2023), 1104745.

[5] Reza Olfati-Saber. 2006. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control* 51, 3 (2006), 401–420.

[6] Craig W Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. 25–34.

[7] Gokul Swamy, Sanjiban Choudhury, Drew Bagnell, and Steven Wu. 2022. Sequence Model Imitation Learning with Unobserved Contexts. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=3nbKUphLBg5

[8] Herbert G Tanner, Ali Jadbabaie, and George J Pappas. 2003. Stable flocking of mobile agents part I: dynamic topology. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, Vol. 2. IEEE, 2016–2021.

[9] E. Tolstaya, F. Gama, J. Paulos, G. Pappas, V. Kumar, and A. Ribeiro. 2019. Learning Decentralized Controllers for Robot Swarms with Graph Neural Networks. In *Conference Robot Learning 2019*. Int. Found. Robotics Res., Osaka, Japan.

[10] Jian Xiao, Zhuoran Wang, Jinhui He, and Guohui Yuan. 2023. A Graph Neural Network Based Deep Reinforcement Learning Algorithm for Multi-agent Leader-follower Flocking. *Information Sciences* (2023), 119074.

[11] Chao Yan, Chang Wang, Xiaojia Xiang, Kin Huat Low, Xiangke Wang, Xin Xu, and Lincheng Shen. 2023. Collision-Avoiding Flocking With Multiple Fixed-Wing UAVs in Obstacle-Cluttered Environments: A Task-Specific Curriculum-Based MADRL Approach. *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[12] Lifeng Zhou, Vishnu D Sharma, Qingbiao Li, Amanda Prorok, Alejandro Ribeiro, Pratap Tokekar, and Vijay Kumar. 2022. Graph neural networks for decentralized multi-robot target tracking. In *2022 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 195–202.