2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Learning Crowd Behaviors in Navigation with Attention-based Spatial-Temporal Graphs

Yanying Zhou and Jochen Garcke

***Abstract*— Safe and efficient navigation in dynamic environments shared with humans remains an open and challenging task for mobile robots. Previous works have shown the efficacy of using reinforcement learning frameworks to train policies for efficient navigation. However, their performance deteriorates when crowd configurations change, i.e. become larger or more complex. Thus, it is crucial to fully understand the complex, dynamic, and sophisticated interactions of the crowd resulting in proactive and foresighted behaviors for robot navigation. In this paper, a novel deep graph learning architecture based on attention mechanisms is proposed, which leverages the spatial-temporal graph to enhance robot navigation. We employ spatial graphs to capture the current spatial interactions, and through the integration with RNN, the temporal graphs utilize past trajectory information to infer the future intentions of each agent. The spatial-temporal graph reasoning ability allows the robot to better understand and interpret the relationships between agents over time and space, thereby making more informed decisions. Compared to previous state-of-the-art methods, our method demonstrates superior robustness in terms of safety, efficiency, and generalization in various challenging scenarios.**

I. INTRODUCTION

As artificial intelligence advances, robots are becoming essential in modern human life. However, ensuring the safety and efficiency of social-aware robots navigating in crowds is a considerable challenge [1]–[3]. These robots must not only avoid collisions and reach their destinations promptly but also act in a socially compliant manner and maintain a pleasant respectful interaction with pedestrians. Therefore, we aim to develop a proactive collision avoidance scheme that is both adaptive and generalizable.

Social-aware robot navigation tasks have been extensively studied and achieved many successful implementations. Early rule-based methods [4]–[8] such as Optimal Reciprocal Collision Avoidance (ORCA) [5] and Social Forces (SFs) [6]–[8] use artificially set geometric or physical rules, where the next action of the robot is taken depending on one-step rules and the current state. This makes these methods short-sighted and highly dependent on the accuracy of the model. Further, trajectory-based methods [9]–[11] plan an appropriate path for the robot after predicting the future trajectories of other agents, which results in far-sighted decisions. However, both methods are prone to the freezing robot problem (FRP), which means that the planner determines that all feasible routes are unsafe and the robot gets stuck [12], [13].

Recently, the use of Deep Reinforcement Learning (DRL) [14]–[19] has enabled the development of learning-based methods for effective navigation strategies. These methods train a value network using state transitions of agents to estimate the value of each state. An optimal policy is then derived from the value network, guiding the robot to make decisions and choose the action that leads to the state with the highest value. They implicitly encode the interactions between agents, capturing the collective impact of the crowd by incorporating pairwise interactions through LSTMs [20] or the maximin operator. However, interactions vary significantly across different environments, rendering navigation policies trained in one environment unsuitable for another. This is due to uncertainties in the dynamic crowd, which affects the way and impact of interactions. Specifically, as the density of crowds increases, it leads to increased mutual influence and interference among individuals, which in turn increases the difficulty and complexity of human-robot interaction in such environments. Therefore, robots need to timely adjust their behavior strategies according to changes in crowd behavior, to better adapt to different scenarios and improve the quality of human-robot interaction.

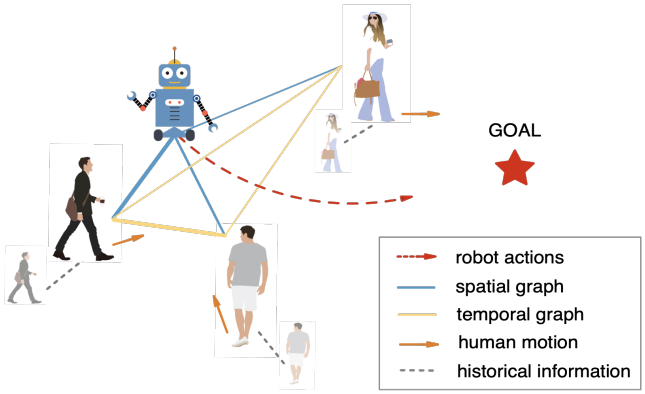


Fig. 1: Illustration of our work. Our model uses spatial-temporal graphs to capture complex crowd dynamics, which aggregates both spatial and temporal attention maps from each agent.

We propose an attention-based spatial-temporal graph learning framework for crowd navigation trained with DRL, which is called "ASTG". First, we introduce graph attention networks to fully model both spatial relations (such as spatial positioning) and temporal relations (such as trajectory intent inference) of the crowd navigation scenario separately. In addition, before modeling the temporal relation interactions, we incorporate an RNN [21] to encode each agent's historical trajectory information as input for the temporal graph, thus implicitly inferring the intentions of all humans. Then, we jointly aggregate the pairwise spatial-temporal interactions

The authors are with the Institute for Numerical Simulation, University of Bonn, Germany. Jochen Garcke is also from Fraunhofer SCAI, Sankt Augustin, Germany. Email: {zhou,garcke}@ins.uni-bonn.de

into a social attention mechanism to capture the relative importance of each human. Together, a comprehensive understanding of crowd behaviors enables efficient robot navigation policies, adaptable to various crowded scenarios.

We present the following contributions: (1) We introduce a novel deep graph neural network ASTG, which fully models the spatial and temporal relations efficiently in crowd navigation. (2) We combine the above with a social-attention mechanism to encode the collective influence of neighbors. (3) We demonstrate the improved performance of our approach in comparison to previous methods, exhibiting strong robustness and generalization in both simple and complex scenarios with varying numbers of humans.

II. RELATED WORK

A. Social-aware Robot Navigation

Early works model agent interactions through *rule-based methods* [4]–[8] or *trajectory-based methods* [9]–[11]. Rule-based methods have a fast response, but are prone to encounter the freezing robot problem. This is because they take actions that only consider the current state without a long-term view ahead. Trajectory-based methods generate a path for the robot to follow after predicting other agents' future trajectories, which can be computationally intensive, especially when the environment has many dynamic agents.

Recently, combining deep learning and RL, *learning-based methods* [22]–[26] achieve good navigation performance in an environment. A self-attention module computes the relative importance in order to take actions, called SARL [22]. [23] propose a structural recurrent neural network to reason about spatial and temporal relationships for planning.

All of the above methods, however, may not generalize well to new or unseen environments and may require additional training data and fine-tuning for each new environment. Significantly, deep reinforcement learning models can suffer from overfitting to the training data, leading to poor generalization performance in new states. To tackle the problem, we introduce the graph attention network to adaptively weigh the importance of different nodes in the graph structure, providing a more informative and discriminative representation of the data. In addition, the attention mechanisms in Graph Attention Network (GAT) can also provide additional regularization, helping to prevent overfitting to the training data and improve generalization performance.

B. Graph-based Learning

Currently, graph neural networks (GNNs) are widely utilized in various applications, including social network analysis, to learn the relationships between entities [27]–[34]. Combining the robust representation capabilities of graphs with the powerful end-to-end learning of neural networks, GNNs represent components as nodes and their relationships as edges. Through local message passing on graphs, these networks efficiently encapsulate relations between nodes, proving effective in a wide array of structured tasks.

GNNs can be used to extract efficient representations in crowd navigation. Graph Convolutional Network (GCN) [28]–[30] and GAT [33]–[35] are typical types of GNNs. [28] present a relational graph model to learn the agents' interaction based on a two-layer GCN. G-GCNRL [29] learns attention weights and aggregates the crowd information with two GCNs. Unlike GCNs that use fixed weights, GAT uses learnable attention coefficients to weigh the contribution of each neighbor when aggregating information. In [34] a GAT is used to extract a nice graph representation. However, it only refers to spatial interaction without the temporal dynamics, which also are useful for planning paths in crowd navigation. [36] proposed a spatio-temporal interaction graph, relying on the predicted future trajectory. Our method applies GAT on the spatial graph and temporal graph separately to encode the environment relationships for dealing with uncertain dynamic environments. In addition, for better decision-making, we incorporate RNN into encoding temporal features to reason about the future intentions of each agent based on the current and past observations of each human. We show that combing spatial-temporal GAT with RNN improves performance in dynamic social navigation environments.

III. PROBLEM FORMULATION

A. Crowd Navigation Modeling

We consider the crowd robot navigation task, where a holonomic robot needs to move towards a goal position through a crowd of n humans. The task can be formulated as a sequential decision-making problem. Assume that all agents move in a two-dimensional space with Euclidean geometry. The current position $\mathbf{p} = [p_x, p_y]$, velocity $\mathbf{v} = [v_x, v_y]$ and radius r of each agent can be observed by the rest. The goal position $\mathbf{g} = [g_x, g_y]$ and the preferred velocity v_{pref} are the hidden parts that the robot is aware of only for itself and one cannot observe the hidden states of humans. Let $\mathbf{s}_t^r$ be the current state of the robot and $\mathbf{s}_t^h = [\mathbf{s}_t^1, \mathbf{s}_t^2, ..., \mathbf{s}_t^n]$ be the observable state of humans. Thus, the joint state for robot navigation is $\mathbf{s}_t^{jn} = [\mathbf{s}_t^r, \mathbf{s}_t^h]$.

We transform the coordinate system with robot-centric coordinates, where the robot is set as the center and the x-axis is the direction towards the goal. Thereby, the robot state and the i-th human state at time t are defined as:

$$\begin{aligned} s^r &= [d_g, v_x^r, v_y^r, v_{pref}, r], \\ s^i &= [p_x^i, p_y^i, v_x^i, v_y^i, r^i, d^i, r^i + r], \end{aligned} \tag{1}$$

where d_g is the L_2 distance between the robot and its goal. d^i is the L_2 distance between the robot and the i-th-human.

B. Reinforcement Learning for Crowd Navigation

In this work, a Deep V-learning approach [29] is employed to evaluate the state values and choose the best action. The robot starts from an initial state $\mathbf{s}_0^{jn}$ at the beginning of each episode. Then, based on the learned policy $\pi(\mathbf{a}_t|\mathbf{s}_t^{jn})$, after taking an action $\mathbf{a}_t \in \mathcal{A}$ at each timestep t, the robot receives a reward r_t and transitions to the successor state $\mathbf{s}_{t+1}^{jn}$. For humans, based on their policies, they take actions and transition to their successor states. For each episode, the process continues until the robot reaches the destination, collides with other humans, or the navigation time t exceeds

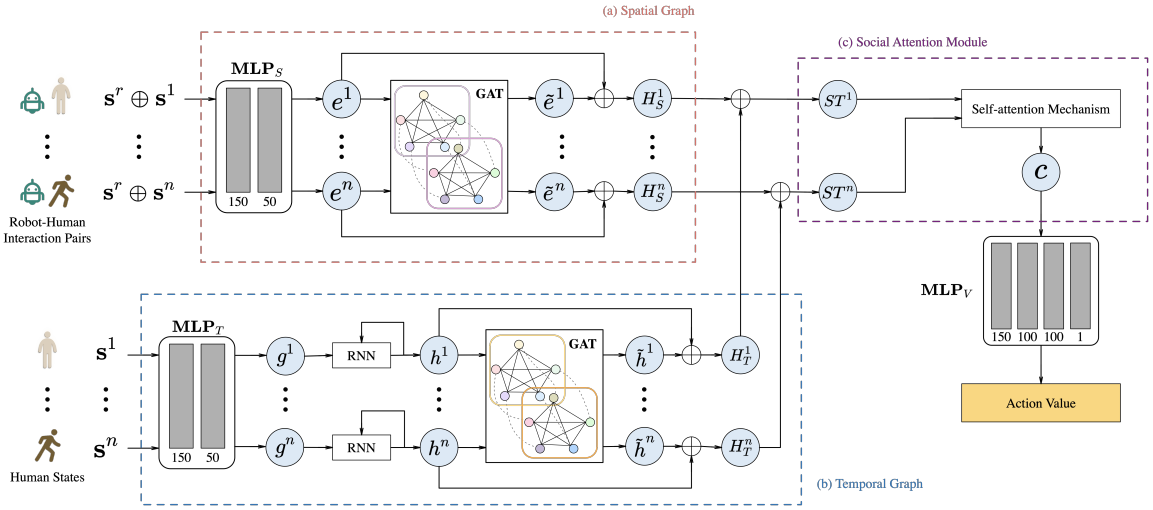


Fig. 2: Network architecture from Section IV. (a) The spatial graph utilizes the GAT to encode direct and indirect spatial interactions between agents. (b) The temporal graph incorporates an RNN to reason about the temporal interactions based on historical information. (c) The social attention module jointly aggregates the pairwise spatial-temporal interactions to capture the crowd representation in the crowd feature, which is then used to estimate the action values.

the episode length T. The objective is to maximize the cumulative reward for each state and thereby to find the optimal policy $\pi^*(\mathbf{s}_t^{jn})$:

$$\pi^*(\mathbf{s}_t^{jn}) = \underset{\mathbf{a}_t}{\operatorname{argmax}}\, R(\mathbf{s}_t^{jn}, \mathbf{a}_t) + \gamma^{\Delta t \cdot v_{pref}} \int_{\mathbf{s}_{t+\Delta t}^{jn}} P(\mathbf{s}_t^{jn}, \mathbf{a}_t, \mathbf{s}_{t+\Delta t}^{jn}) V^*(\mathbf{s}_{t+\Delta t}^{jn})\, d\mathbf{s}_{t+\Delta t}^{jn},$$

$$V^*(s_t^{jn}) = \sum_{t'=t}^{T} \gamma^{t' \cdot v_{pref}} R(\mathbf{s}_{t'}^{jn}, \pi^*(\mathbf{s}_{t'}^{jn})), \tag{2}$$

where Δt is the time step and $r_t = R(\mathbf{s}_t^{jn}, \mathbf{a}_t)$ denotes the received reward at time t. $P(\mathbf{s}_t^{jn}, \mathbf{a}_t, \mathbf{s}_{t+\Delta t}^{jn})$ is the transition probability between t and $t + \Delta t$. $\gamma \in (0, 1)$ is a discount factor that is normalized by the preferred velocity v_{pref}. We utilize the reward function formulated as in [22], which provides rewards for task accomplishment while simultaneously imposing penalties for collisions or uncomfortable distances.

IV. METHODOLOGY

We propose a novel framework for social-aware robot navigation, see Fig. 2. It leverages the superior flexibility of graph structures with varying nodes, as they can adapt to graphs and input data of varying sizes through information exchange between nodes. This is in contrast to using a feed-forward neural network to reason about crowd interactions, which requires input data of fixed size

A. Spatial Graph Representation

The spatial-temporal graph network calculates the spatial attention and temporal attention separately from the environment state matrix, which is used to represent the edges of the spatio-temporal graph. In the spatial graph, we input rough pairwise spatial interactions to calculate agents' spatial dependencies. For i-th human, we initially aggregate its state at the current moment with that of the robot to compute rough spatial interactions $(\mathbf{s}^r \oplus \mathbf{s}^i)$. Then, we calculate spatial attention maps through graph attention network layers, representing the importance of adjacent interactions, and encode spatial features such as distance and relative direction between agents in pairwise interactions. For instance, interaction at the i-th node is influenced by its connection with other nodes, with a more substantial influence reflected by a higher attention weight. Simultaneously, this node aggregates spatial information from adjacent interactions through the influence of spatial edges, representing the collective impact on the node from the crowd dynamics. This impact not only considers explicitly modeled pairwise human-robot interactions but also implicitly captures indirect human-human interactions and crowd-robot interactions through edge weights.

As shown in Fig. 2, in the spatial graph, we use the states of the robot and the i-th human at time t to derive a node feature via a multilayer perceptron (MLP$_S$):

$$e_t^i = f_{spatial}(\mathbf{s}_t^r, \mathbf{s}_t^i; W_e), \tag{3}$$

where W_e are the network weights and $f_{spatial}(\cdot)$ is an MLP with a ReLU activation function. The rectified linear unit (ReLU) activation function used in the framework aims to capture non-linear features in the feedforward network.

We use a graph attention layer to model spatial interactions in a crowd. The input is denoted by $E = [e_t^1, ..., e_t^n]$ and $e^i = e_t^i$ for short. The normalized spatial coefficients α_{ij}^S in the attention mechanism can be computed by:

$$\alpha_S^{ij} = \frac{\exp(\text{LeakyRelu}(a[\mathbf{W}e^i || \mathbf{W}e^j]))}{\sum_{k \in \mathcal{N}^i} \exp(\text{LeakyRelu}(a[\mathbf{W}e^i || \mathbf{W}e^k]))}, \tag{4}$$

where α_S^{ij} represents the importance of node j to node i. $\mathbf{W}(\cdot)$ is the weight matrix and $||$ is the concatenation operation, while $\mathcal{N}^i$ indicates the neighborhood of node i in the graph. After passing through a fully connected network

(FCN) $a(\cdot)$, a LeakyReLU activation follows. Finally, a softmax function is used to normalize the spatial attention coefficients α_{ij}^{S}. Thus, the output node features of the graph attention layer are:

$$\tilde{e}^{\,i} = \sigma(\sum_{j \in \mathcal{N}^i} \alpha_S^{ij}\ \mathbf{W} e^{j}), \quad \tilde{E} = [\tilde{e}^{\,1}, ..., \tilde{e}^{\,n}], \tag{5}$$

In this work, through the graph attention layer, the i-th interaction feature is transformed into a higher-level spatial feature that models indirect robot-human and human-human interactions in a crowd. Additionally, residual connections are employed in the spatial and temporal graph networks to accelerate convergence and stabilize the framework [37]. Thus, the final spatial features $H_{spatial}$ is described by:

$$H_S^i = e^i + \tilde{e}^{\,i}, \quad H_{spatial} = [H_S^1, ..., H_S^n]. \tag{6}$$

B. Temporal Graph Representation

Due to the high motion-dependency of the temporal dimension, we introduce an RNN for each agent to capture the dynamics of its trajectory. Then, in the temporal graph, we treat the motion feature of each human as a node of the graph, and the edges signify the relationships between humans' motions, whose weights can indicate the importance level. Since motion is continuous, we can predict future actions based on the current motion information. To achieve this, we utilize graph attention network layers to compute a temporal attention map, estimating the mutual influence of neighboring agents' motion behaviors and aggregating these influences to form temporal features. These temporal features encompass both the current trajectory information of humans and predictions of future trajectories, allowing the robot to better understand human behavior and intentions.

First, we embed i-th human's state through an MLP_T into a fixed length vector g_t^i:

$$g_t^i = f_{temporal}(\mathbf{s}_t^i; W_g), \tag{7}$$

where W_g are the embedding weights and $f_{temporal}(\cdot)$ is an MLP with ReLU activation function. Then, we process g_i^t with an RNN cell:

$$h_t^i = \text{RNN}(h_{t-1}^i, g_t^i), \tag{8}$$

where h_t^i is the hidden state at time t, which changes over time, reflecting the evolution of i-th agent's state.

The hidden states h_t^i are then fed into the graph attention layer to model temporal interactions. GAT learns the attention distribution between nodes adaptively, enabling it to weigh the information from different nodes based on their relationship. This implies that GAT facilitates the propagation of temporal interaction information between nodes. The input is $H = [h_t^1, ..., h_t^n]$ and $h^i = h_t^i$ for short. The normalized temporal coefficients α_{ij}^T are described by:

$$\alpha_T^{ij} = \frac{\exp(\text{LeakyRelu}(\alpha[\mathbf{W}h^i || \mathbf{W}h^j]))}{\sum_{k \in \mathcal{N}^i} \exp(\text{LeakyRelu}(\alpha[\mathbf{W}h^i || \mathbf{W}h^k]))}. \tag{9}$$

Combining RNN and GAT allows for a more accurate capture of the evolving states and interactive changes of agents in a social environment over time. While RNN captures the temporal variations in agents' states, GAT, through its attention mechanism, considers interactions with other nodes, leading to a finer-grained temporal dynamic modeling. By comprehensively considering the evolution of each agent's state alongside their social relationships, the robot can better predict future human behaviors and intentions, and even assist in analyzing collective behavioral patterns within a group. Hence, the final temporal graph feature with a one-layer GAT can be described as follows:

$$\tilde{h}^i = \sigma(\sum_{j \in \mathcal{N}^i} \alpha_T^{ij}\ \mathbf{W}h^j), \quad \tilde{H} = [\tilde{h}^1, ..., \tilde{h}^n], \tag{10}$$

After deploying a residual connection structure, the final temporal features $H^{temporal}$ is:

$$H_T^i = h^i + \tilde{h}^i, \quad H_{temporal} = [H_T^1, ..., H_T^n]. \tag{11}$$

C. Social Attention Mechanism

To capture the uncertainty of crowd movements, we build a social attention module that fuses the spatial and temporal features of each agent and captures dependencies among agents. Inspired by [22], we build a self-attention network to assign attention weights w^i for each agent's aggregated spatial-temporal pairwise feature $ST^{\,i} = [H_S^i, H_T^i]$ and encode the collective impact of a crowd.

First, $ST^{\,i}$ is transformed into an attention score w^i:

$$ST^{\,m} = \frac{1}{n}\sum_{k=1}^{n} ST^{\,k}, \quad w^i = f_\alpha(ST^{\,i}, ST^{\,m}; W_\alpha), \tag{12}$$

where W_α is the weight and f_α is nonlinear transformation. We then obtain the final crowd representation c by the product sum of each spatial-temporal pairwise feature $ST^{\,i}$ and attention weights w_i:

$$c = \sum_{i=1}^{n} softmax(w^i) ST^{\,i} \tag{13}$$

D. Graph-Based Planning

The crowd representation c is the input to MLP_V to estimate the state value V for planning (where W_v denotes the weights):

$$V = f_v(\mathbf{s}^r, c; W_v). \tag{14}$$

V. EXPERIMENTS

A. Simulation Environment

Our 2D simulation environment for crowd robot navigation follows [22]. In particular, we use circle-crossing scenarios with a radius of 4m in our simulations and all humans are controlled by the ORCA policy. We keep the invisible setting for the robot to prevent humans from overreacting to the robot and thus avoid learning an aggressive navigation policy that moves too directly to the goal position and gets too close to humans. Dynamic humans are randomly positioned on the circle and will move to opposite positions on the same circle, while static humans are randomly located in the same circle and do not move. We assume holonomic kinematics for the

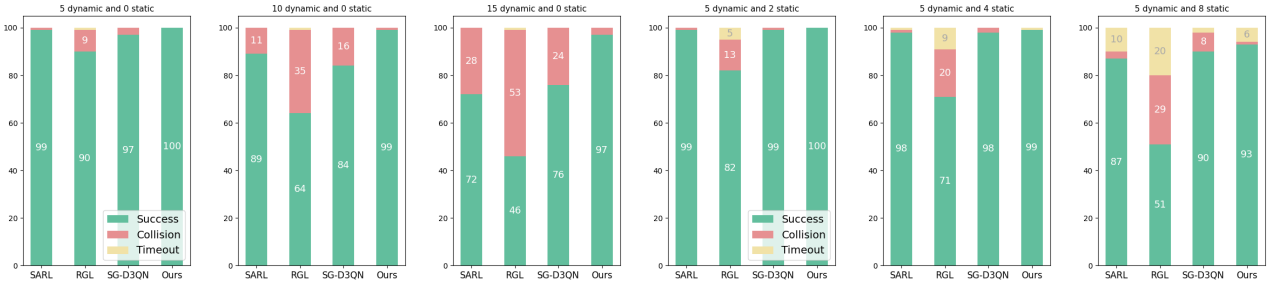
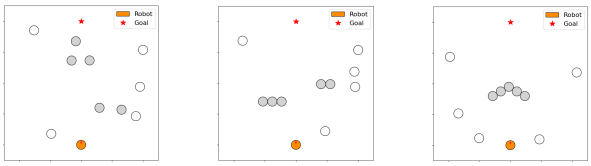


Fig. 3: Quantitative evaluation on scenarios with different numbers of humans.

Methods	different numbers of dynamic humans									5 dynamic humans and different numbers of static humans								
	Disc.(%)			$t_{succ.nav}$			$t_{weighted.nav}$			Disc.(%)			$t_{succ.nav}$			$t_{weighted.nav}$		
	5	10	15	5	10	15	5	10	15	2	4	8	2	4	8	2	4	8
SARL [22]	0.02	0.11	0.22	10.58	11.64	**12.33**	10.91	14.18	17.80	0.05	0.09	0.15	10.51	**10.64**	**11.16**	11.10	11.71	13.52
RGL [28]	0.21	0.42	0.53	10.23	11.43	12.51	12.57	17.65	21.60	0.34	0.43	0.57	10.59	10.97	12.06	13.88	15.53	18.41
SG-DQN [34]	0.05	0.08	0.09	**10.03**	**11.24**	13.41	**10.63**	13.71	16.56	0.04	0.06	**0.09**	**10.07**	11.24	12.07	**10.42**	11.79	13.62
Ours	**0.01**	**0.04**	**0.08**	11.40	12.65	13.28	11.45	**12.97**	**14.13**	**0.03**	**0.06**	0.10	11.09	11.22	11.97	11.25	**11.56**	**12.83**

TABLE I: Evaluation performance comparison in the simple scenarios. "Disc. (%)" is the average frequency of duration that the robot invades the humans' comfort area. "$t_{succ.nav}$" is the average navigation time of success cases. "$t_{weighted.nav}$" (Eq.(18)) is a novel weighted navigation time metric considering the impact of the discomfort steps and collision cases.



(a) DS (distributed) (b) RO (row3and2) (c) CO (concave5)

Fig. 4: Simple and complex scenarios. All are with 5 dynamic humans and 5 static humans in different combinations: (a) Completely dispersed, (b) divided into two rows with 3 and 2 static humans separately, (c) in a concave shape.

robot that can move in any direction. The robot's velocity is discretized into 5 exponential speeds within the range of $(0, V_{pref}]$, accompanied 16 headings evenly spaced between $[0, 2\pi)$. To fully analyze the effectiveness of the proposed model, we evaluate all models for 1000 random test cases in both simple and complex scenarios.

B. Implementation Details

Baselines for comparison: We compare the performance of our model with three state-of-the-art methods: SARL [22], RGL [28] and SG-DQN [34]. SARL is a baseline for Deep V-learning. In addition, RGL and SG-DQN are used as baselines for graph-based learning methods.

Training: We train all methods in an environment that consists of 5 dynamic and 2 scattered static humans using data from 10k episodes. For a fair comparison, the network architectures of all baselines are as in the original papers.

Evaluation: We evaluate the generalization capability of the policy of each method, learned in the above training scenario, in both simple and complex scenarios with 1k episodes. To vary the start and end positions of agents over the episodes we use different random seeds, where each (random) episode configuration is evaluated for all methods. As evaluation metrics, we use the percentage of success, collision, and timeout cases, respectively, as well as navigation time-related metrics.

Simple scenarios are set up with different numbers of distributed dynamic and/or static humans (like Fig. 4(a)), such as scenarios with only $n = 5, 10, 15$ dynamic humans, and scenarios with 5 dynamic humans or $m = 2, 4, 8$ scattered static humans. In addition, based on 5 dynamic humans, the complex scenario has 5 static humans with different group combinations (like Fig. 4(b),(c)), whose positions are randomly set in the circle.

C. Quantitative Evaluation

To show the effectiveness of our method, we compare our method with SARL [22], RGL [28] and SG-DQN [34]. Fig.3, 5, Tab.I, II show the results of all methods under different environments, where ASTG is the best one with the highest success rate and the lowest comfort rates in all scenarios.

In Fig. 3 and 5, we observe that the SARL method [22] demonstrates restricted performance and social compliance due to its inability to fully capture intricate social interactions. Moreover, despite utilizing GNN for modeling human-robot interactions, RGL [28] has not demonstrated strong generalization performance. This limitation stems from the GCN's simple convolution operations applied on features dictated by fixed rule-based weights, which weaken adaptability to varying environments, leading to a sharp increase in collision rates as the obstacles' amount grows. In contrast, GAT can assign different weights based on the importance of these features, thereby better capturing critical environmental characteristics. This advantage is visible in SG-DQN's [34] competitive stance alongside our ASTG. However, SG-DQN [34] lacks the ability to capture time-varying motion features, thus reducing predictive accuracy. In comparison, our ASTG exhibits significantly slower increases in collision rates and navigation times in pure dynamic environments and

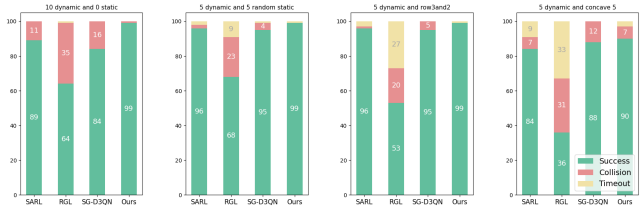


Fig. 5: Quantitative evaluation under complex scenarios.

Group statics	Disc.(%)			$t_{succ.nav}$			$t_{weighted.nav}$		
	DS	RO	CO	DS	RO	CO	DS	RO	CO
SARL [22]	0.11	0.10	0.05	**10.83**	**11.08**	11.54	12.18	12.31	13.21
RGL [28]	0.44	0.56	0.59	11.25	11.37	11.27	16.22	16.50	18.58
SG-DQN [34]	0.07	0.07	0.06	10.96	11.25	11.15	11.93	12.20	13.06
Ours	**0.06**	**0.06**	**0.03**	11.35	11.34	**11.05**	**11.75**	**11.76**	**12.21**

TABLE II: Evaluation performance comparison in the scenarios with 5 dynamic humans and different static groups (DS (distributed), RO (row3and2), and CO (concave)).

environments containing both dynamic and static humans, demonstrating excellent generalization performance.

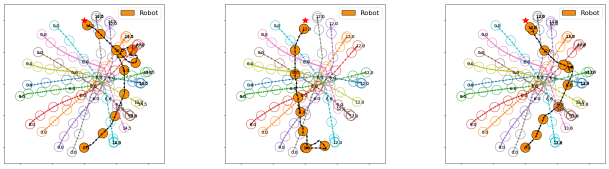
From Tab.I, II, it shows that the baselines enjoy a better performance in $t_{succ.nav}$, which is defined as the average navigation time of the successful cases in seconds. However, they have either a higher discomfort frequency or higher collision rate, which means, to achieve a shorter navigation time, the robot takes a lot of aggressive actions, badly affecting human comfort or even having collisions. To better measure and analyse the behavior, we propose a weighted navigation time metric that penalizes aggressive policies:

$$t_{weighted.nav} = \frac{t_{dnav} + N_{coll} * t_{limit}}{N_{succ} + N_{coll}}, \tag{15}$$

where N_{succ} and N_{coll} are the number of success and collision cases, respectively. t_{limit} is the maximum navigation time, which set as 25s in our simulations. t_{dnav} is extended from the original navigation time t_{nav} to penalize the impact of discomfort. Specifically, if a step is causing discomfort, we add half of a time step to $t_{succ.nav}$, which equals $t_{dnav} = N_{succ} * t_{succ.nav} + N_{disc} * 0.5 * \Delta t$ and N_{disc} is the number of discomfort steps in success cases. Thus, when considering collision cases and the discomfort frequency, our method shows a shorter weighted navigation time $t_{weighted.nav}$.

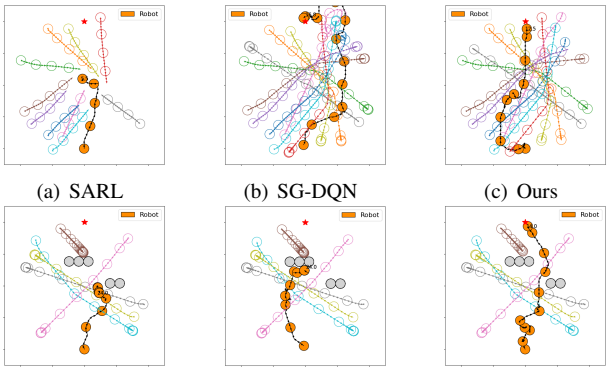
D. Ablation Study

To assess the individual contributions of the different components in our model, we conduct an ablation study where we independently train and test the spatial and temporal branches with the same conditions as before. Fig. 6 shows that the spatial graph and ASTG learn to navigate a safer side away, while the spatial graph has a longer navigation time. Furthermore, the temporal graph and ASTG are good at reasoning about the dynamics of the crowd, while ASTG demonstrates better social norm compliance. These findings suggest that incorporating the spatial and temporal graph can improve the performance and better adapt to dense scenarios with dynamic varying numbers of humans. More quantitative details of our work are available in [38].



(a) Only spatial graph (b) Only temporal graph (c) Full ASTG

Fig. 6: Simulation trajectories on the same case.



(a) SARL (b) SG-DQN (c) Ours

(d) SARL (e) RGL (f) Ours

Fig. 7: Trajectory comparisons under simple and complex scenarios.

E. Qualitative Evaluation

To qualitatively evaluate our method, we show in Fig.7 trajectories for different methods in a fixed exemplary, but representative, simple and complex scenario. In the simple scenario with 10 dynamic humans, Fig. 7(top), SARL collides with humans, while SG-DQN detours largely to achieve the goal. In contrast, our method slightly detours at first and then directly reaches the goal, resulting in a shorter path. When the scenarios are extended from simple to complex, Fig.7(bottom) shows SARL and RGL have timeouts in the static group scenario. As shown in Fig.7(f), our method avoids these timeout failures by moving slowly at first and then going through the crowd. Spatial-temporal graph reasoning helps to understand relationships between agents over time and space and use that to make more informed decisions. The examples illustrate how our method adapts to dense and complex situations in uncertain environments.

VI. CONCLUSIONS

We proposed a novel attention-based spatial-temporal graph learning model for crowd robot navigation. By formulating spatial and temporal graphs with a GAT, we implicitly compute both direct and indirect spatial interactions and temporal crowd features so that our model can reason for decision-making under changing scenarios. Integrating a RNN, our model incorporates past trajectories into the temporal graph to also reason about the future intentions of each agent. We intend to further analyse and extend the GAT-module, for example using multiple graph layers, which can help to extract deeper information from the crowd. Finally, for transitioning from simulation to real-world application, a major challenge is ensuring the algorithm's accurate adaptation to the complexities and variations of the real world.

REFERENCES

[1] C. Mavrogiannis, F. Baldini, A. Wang, D. Zhao, P. Trautman, A. Steinfeld, and J. Oh, "Core challenges of social robot navigation: A survey," arXiv preprint arXiv:2103.05668, 2021.

[2] R. Möller, A. Furnari, S. Battiato, A. Härmä, and G. M. Farinella, "A survey on human-aware robot navigation," Robotics and Autonomous Systems, vol. 145, p. 103837, 2021.

[3] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," Autonomous Robots, vol. 46, no. 5, pp. 569–597, 2022.

[4] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in 2008 IEEE ICRA, 2008, pp. 1928–1935.

[5] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in Robotics Research: The 14th International Symposium ISRR. Springer, 2011, pp. 3–19.

[6] G. Ferrer, A. Garrell, and A. Sanfeliu, "Social-aware robot navigation in urban environments," in 2013 European Conference on Mobile Robots, 2013, pp. 331–336.

[7] G. Ferrer, A. G. Zulueta, F. H. Cotarelo, and A. Sanfeliu, "Robot social-aware navigation framework to accompany people walking side-by-side," Autonomous robots, vol. 41, no. 4, pp. 775–793, 2017.

[8] X.-T. Truong and T. D. Ngo, "Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model," IEEE Transactions on Automation Science and Engineering, vol. 14, no. 4, pp. 1743–1760, 2017.

[9] K. Li, M. Shan, K. Narula, S. Worrall, and E. Nebot, "Socially aware crowd navigation with multimodal pedestrian trajectory prediction for autonomous vehicles," in 2020 IEEE ITSC, 2020, pp. 1–8.

[10] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," Autonomous Robots, vol. 35, no. 1, pp. 51–76, 2013.

[11] Y. Chen, F. Zhao, and Y. Lou, "Interactive model predictive control for robot navigation in dense crowds," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 4, pp. 2289–2301, 2021.

[12] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in IEEE/RSJ IROS, 2010, pp. 797–803.

[13] A. J. Sathyamoorthy, U. Patel, T. Guan, and D. Manocha, "Frozone: Freezing-free, pedestrian-friendly navigation in human crowds," IEEE Robotics and Automation Letters, vol. 5, no. 3, pp. 4352–4359, 2020.

[14] Y. Li, "Deep reinforcement learning: An overview," arXiv preprint arXiv:1701.07274, 2017.

[15] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," Tsinghua Science and Technology, vol. 26, no. 5, pp. 674–691, 2021.

[16] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: A survey," in 2020 IEEE Symp. Series on Comput. Intelligence (SSCI), 2020, pp. 737–744.

[17] R. Wang, W. Wang, and B.-C. Min, "Feedback-efficient active preference learning for socially aware robot navigation," in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2022, pp. 11 336–11 343.

[18] S. S. Samsani, H. Mutahira, and M. S. Muhammad, "Memory-based crowd-aware robot navigation using deep reinforcement learning," Complex & Intelligent Systems, vol. 9, no. 2, pp. 2147–2158, 2023.

[19] H. Zeng, R. Hu, X. Huang, and Z. Peng, "Robot navigation in crowd based on dual social attention deep reinforcement learning," Mathematical Problems in Engineering, vol. 2021, pp. 1–11, 2021.

[20] A. Graves and A. Graves, "Long short-term memory," Supervised sequence labelling with recurrent neural networks, pp. 37–45, 2012.

[21] L. R. Medsker and L. Jain, "Recurrent neural networks," Design and Applications, vol. 5, pp. 64–67, 2001.

[22] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in 2019 IEEE ICRA, 2019, pp. 6015–6022.

[23] S. Liu, P. Chang, W. Liang, N. Chakraborty, and K. Driggs-Campbell, "Decentralized structural-rnn for robot crowd navigation with deep reinforcement learning," in 2021 IEEE ICRA, 2021, pp. 3517–3524.

[24] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in 2017 IEEE/RSJ IROS, 2017, pp. 1343–1350.

[25] W. Wang, R. Wang, L. Mao, and B.-C. Min, "Navistar: Socially aware robot navigation with hybrid spatio-temporal graph transformer and preference learning," arXiv preprint arXiv:2304.05979, 2023.

[26] S. Wang, R. Gao, R. Han, S. Chen, C. Li, and Q. Hao, "Adaptive environment modeling based reinforcement learning for collision avoidance in complex scenes," in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2022, pp. 9011–9018.

[27] H. He, H. Fu, Q. Wang, S. Zhou, and W. Liu, "Spatio-temporal transformer-based reinforcement learning for robot crowd navigation," arXiv preprint arXiv:2305.16612, 2023.

[28] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, "Relational graph learning for crowd navigation," in IROS, 2020, pp. 10 007–10 013.

[29] Y. Chen, C. Liu, B. E. Shi, and M. Liu, "Robot navigation in crowds by graph convolutional networks with attention learned from human gaze," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 2754–2761, 2020.

[30] Y. Chen, C. Liu, X. Mei, B. E. Shi, and M. Liu, "HGCN-GJS: Hierarchical graph convolutional network with groupwise joint sampling for trajectory prediction," in IEEE/RSJ IROS, 2022, pp. 13 400–13 405.

[31] Z. Zhou, Z. Zeng, L. Lang, W. Yao, H. Lu, Z. Zheng, and Z. Zhou, "Navigating robots in dynamic environment with deep reinforcement learning," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 25 201–25 211, 2022.

[32] S. Liu, P. Chang, Z. Huang, N. Chakraborty, W. Liang, J. Geng, and K. Driggs-Campbell, "Socially aware robot crowd navigation with interaction graphs and human trajectory prediction," arXiv preprint arXiv:2203.01821, 2022.

[33] T. Zhang, T. Qiu, Z. Pu, Z. Liu, and J. Yi, "Robot navigation among external autonomous agents through deep reinforcement learning using graph attention network," IFAC-PapersOnLine, vol. 53, no. 2, pp. 9465–9470, 2020.

[34] Z. Zhou, P. Zhu, Z. Zeng, J. Xiao, H. Lu, and Z. Zhou, "Robot navigation in a crowd by integrating deep reinforcement learning and online planning," Applied Intelligence, pp. 1–17, 2022.

[35] T. Zhang, Z. Liu, Z. Pu, J. Yi, Y. Liang, and D. Zhang, "Robot subgoal-guided navigation in dynamic crowded environments with hierarchical deep reinforcement learning," International Journal of Control, Automation and Systems, pp. 1–13, 2023.

[36] S. Liu, P. Chang, Z. Huang, N. Chakraborty, K. Hong, W. Liang, D. L. McPherson, J. Geng, and K. Driggs-Campbell, "Intention aware robot crowd navigation with attention-based interaction graph," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 12 015–12 021.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[38] Y. Zhou, "Social-aware robot navigation based on deep reinforcement learning," Ph.D. dissertation, Institute for Numerical Simulation, University of Bonn, 2024, submitted.