machines

MDPI

Article

Deep Reinforcement Learning Based on Social Spatial–Temporal Graph Convolution Network for Crowd Navigation

Yazhou Lu [1,2], Xiaogang Ruan [1,2] and Jing Huang [1,2,*]

[1] Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2] Beijing Key Laboratory of Computational Intelligence and Intelligence System, Beijing 100124, China
* Correspondence: huangjing@bjut.edu.cn

Abstract: In the crowd navigation, reinforcement learning based on graph neural network is a promising method, which effectively solves the poor navigation effect based on social interaction model and the freezing behavior of robot in extreme cases. However, since the information correlation of human trajectory has not been involved in the method, its performance still needs improvement. Therefore, we proposed a deep reinforcement learning model based on Social Spatial–Temporal Graph Convolution Network (SSTGCN) to handle the crowd navigation problem, in which the spatial–temporal information of human trajectory has been taken advantage to predict human behavior intentions and help robot plan path more efficiently. The model consists of graph learning module and robot forward planning module. In the graph learning module, the latent features of agents are taken advantage to reason about the relations among the agents, and SSTGCN is used to update feature matrix. In addition, value estimation module calculates state representation and state prediction module predicts the next state. The robot forward planning module makes use of k-step planning to estimate the quality of state and searches the best k steps planning. We tested our model in the Crowd-Nav platform, and the results show that our model has high navigation success rate and short navigation time. In addition, it has good robustness to crowd changes.

Keywords: SSTGCN; crowd navigation; graph learning; reinforcement learning; mobile robot

check for updates

Citation: Lu, Y.; Ruan, X.; Huang, J. Deep Reinforcement Learning Based on Social Spatial–Temporal Graph Convolution Network for Crowd Navigation. *Machines* **2022**, *10*, 703. https://doi.org/10.3390/machines10080703

Academic Editor: Fernando Gomez-Bravo

Received: 22 July 2022
Accepted: 13 August 2022
Published: 17 August 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the development of robot technology, the mobile robot plays an important role in many occasions (shopping malls, hospitals) [1]. During the movement of the robot, the actions of human around it should be considered to ensure that the robot can complete the navigation task efficiently and safely. However, the action intention of human is hard to predict, so the robot navigation in the crowd is a very challenging problem [2,3].

Some studies regard human as simple dynamic obstacles and only consider the next action. When the robot encounters human, it will adopt special obstacle avoidance rules to avoid collision, which makes the robot behavior look short-sighted and unnatural [4,5]. In order to solve the problem, a more complex action model is proposed [6]. This model infers the motion intention of the human and generates a group of predicted paths, and then uses the path planning algorithm to plan the path for the robot. However, the freezing behavior occurs when the robot and the human are about to meet, which is bad for crowd navigation [7]. The current solutions are divided into model-based and learning based. The model-based method needs to introduce a parameter adjustment model to explain the social interaction model, but it is not clear whether the complex human behavior follows these precise geometric behavior rules [8–10]. The purpose of the learning based method is to develop a strategy to imitate human behavior. Deep reinforcement learning is used to learn effective strategies to solve navigation problems [11–13]. These strategies either do not use human–robot interaction information, or simplify human behavior into simple dynamic

obstacles to simplify human action prediction, or generally only consider the current state of human action, and do not combine human behavior prediction for navigation. These conditions will lead to the robot navigation failure easily.

Since graph neural networks are good at describing the relationships among entities, they are introduced to solve the crowd navigation problem [14]. A variant of GNN is graph convolution neural network (GCN) [15]. Its adjacency matrix represents the relationship between nodes. Wang et al. [16] and Grover et al. [17] uses the learning attention to calculate new features in order to learn the relationship between nodes. To solve the problem of robot crowd navigation, we first need to model the human–robot interaction model, and then complete the next action planning of the robot according to the human trajectory information. Chen et al. [18] use graph learning to solve crowd navigation. However, it can not make full use information of trajectories. We propose a deep reinforcement learning model based on Social Spatial–temporal Graph Convolution Network (SSTGCN). SSTGCN models the relationship between agents, extracts the spatiotemporal information of trajectories, and calculates the interaction between agents. According to this interaction characteristic, we predict the trajectory of humans and use reinforcement learning training strategy to complete robot navigation. Figure 1 shows the interactive reasoning between robot and human, human and human, and completes the efficient and safe navigation of robot by predicting the motion trajectory. In the figure, the smaller the robot and human are, the farther the predicted steps are. The close relationship between humans is inferred according to the relationship and planned according to the human trajectory. The robot can safely reach the target point. Simulation experiments are designed to validate this strategy and we will conduct physical experiments to verify this strategy in the near future.

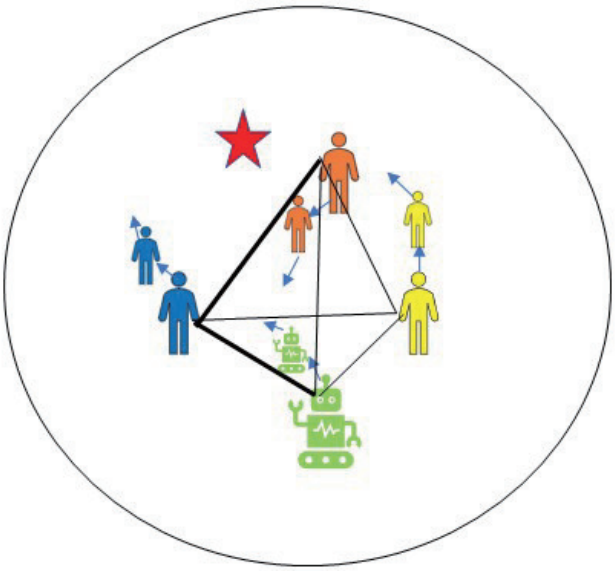

Figure 1. Robot and human interaction diagram.

Our main contributions are threefold. Firstly, the Social Spatial–temporal Graph Convolution Network (SSTGCN)is proposed to model human–robot interaction. Compared with Graph Convolution Network (GCN), SSTGCN can make full use of the spatiotemporal information of trajectory to help robot make better navigation behavior. Secondly, we add a value prediction network to the model to make a certain depth prediction, which can predict the future human movement. Finally, we propose a new reinforcement learning based on graph method for crowd navigation tasks, which can improve the success rate of robot navigation and solve the freezing behavior of robots. Simulation experiments show that the model has significant advantages in learning effective strategies.

In this paper, the related work is presented in Section 2. The proposed model is explained in Section 3, while results are shown in Section 4. In addition, the discussion is shown in Section 5. Finally, the conclusion and future directions are discussed in Section 6.

2. Literature Review

This section describes the literature research on the relevant contents of the model. Firstly, it describes the related research of crowd navigation and indicates the navigation problems to be solved. Secondly, it describes the application of relational reasoning in crowd navigation. Finally, it introduces the research status of reinforcement learning based on the model.

2.1. Crowd Navigation

The solution of robot navigation in crowd is proposed. The social force model [19] has been applied to different environments and achieved good performance [8,20]. Reciprocal velocity observers (RVO) considering communication with other agents has been applied in multi-agent navigation scenarios [21]. ORCA makes multiple robots avoid collision when navigating in complex environments [22]. However, these model-based methods require parameter selection, which makes the modeling process very difficult. This can not well explain the real human behavior, and will make the robot produce unnatural navigation behavior. The navigation success rate is low, and the robot freezing behavior will occur in extreme cases.

The purpose of imitation learning is to teach robot learn how to complete crowd navigation directly from human behavior demonstration. Long et al. and Tai et al. [23,24] used imitation learning to learn navigation strategies from original laser data or depth input under supervision. Recently, Chen et al. and Chen et al. [11,25] proposed to learn the state value of the value network and use deep reinforcement learning for crowd navigation. Chen et al. [13] based on the existing work, used the attention pool module to learn the robot state representation by using the pairwise interaction feature. However, the model made a simplified assumption when the human action was not clear. A model about graph convolutional network (GCN) for reinforcement learning was proposed to integrate information about the environmental context of the robot[26]. Chen et al. [18] proposed to use the model-based deep reinforcement learning combined with graph learning method for crowd navigation. However, the temporal and spatial information of the trajectory is not integrated, and the inference of human behavior intention needs to be improved. Considering the spatiotemporal information of the trajectory, we propose the SSTGCN learning method, which captures the spatial and temporal correlation of the trajectory information to complete the human behavior intention inference, and combines with reinforcement learning to complete the k-step depth path planning to guide the robot to complete the navigation task.

2.2. Relational Reasoning

The purpose of relational reasoning is to determine the relationship between objects [27]. The graph structure well represents the relationship between objects. The graph convolution network extends the convolution operation to graph structure data, which is of great concern in graph representation learning. Most GNNS run in input systems with known graph input. However, many interaction relationships in robot crowd navigation tasks are unknown, such as human behavior [28] and crowd interaction [29]. It is very important for the robot to infer the relationship between travelers in the crowd navigation task, which helps to infer the human behavior intention (for example, the walking distance between friends is relatively close, the robot cannot pass through, and strangers will cross, which does not affect the robot's path planning). Kipf et al. [30] the proposed model can infer the relationship graph between the robot and the crowd at each time step, and learn the state representation of each agent based on the graph. A graph convolution network is proposed to capture the attention between robot and human, human and human to predict

human behavior [18]. In order to integrate the dynamic characteristics of the trajectory, Mohamed et al. [31] proposed the social spatiotemporal graph convolution network to predict the human trajectory. The model well integrates the spatiotemporal characteristics of the trajectory information and improves the human performance of the model, but the model is not used for crowd navigation. Inspired by the model, SSTGCN with a new structure is proposed to capture the spatiotemporal relationship of the robot and human's attention to predict the human behavior trajectory to help the robot with path planning.

2.3. Model Based Reinforcement Learning

Model based RL first learns the environment model from experience, and then uses the model to improve the value optimization. In this model, this method attempts to learn a model that predicts future observations based on behavior, which is used to simulate the real environment and plan future k-step actions. Finn et al. [32] learn the state transition model by predicting the future frame, so as to realize the goal oriented robot operation, Oh et al. [33] learn a dynamic model, and the abstract state of the model is trained to make selective prediction behavior for the future value. Chen et al. [18] proposed relationship graph prediction uses the original human state as the input, and predicts multiple interacting human trajectories. Based on it, we use SSTGCN to learn the spatial–temporal correlation of the original state feature information, and combine graph learning and model-based RL algorithm for robot crowd navigation task.

3. Methods

This model consists of SSTGCN, three MLP and k-step planning. In Figure 2, MLP1 is responsible for extending the dimension of state information obtained from the environment as the SSTGCN input. SSTGCN is responsible for updating the feature matrix I, where the graph convolution network layer is used to update the spatial features and its output is feature matrix II. The LSTM layer is used to capture the time-varying action characteristics.The output characteristics of robot nodes naturally aggregate the spatiotemporal information of human nodes. The model of trajectories by SSTGCN helps infer human action intentions compared with the GCN . Both robot node feature and human node feature are passed through MLP2 and MLP3, the action estimation value of the robot and the predicted action state of the human are obtained respectively. K-step planning is responsible for evaluating the quality of robot action estimates and searching for the best action sequence.

In this section, the first part describes the deep reinforcement learning algorithm for crowd navigation. The second part uses SSTGCN to model the human–robot interaction model, infer the relationship between agents, and complete the robot's motion estimation value and human's predicted motion state. The third part describes the k-step planning algorithm to help the robot evaluate the motion quality and search for the best motion sequence. The fourth part describes the training methods used in the model

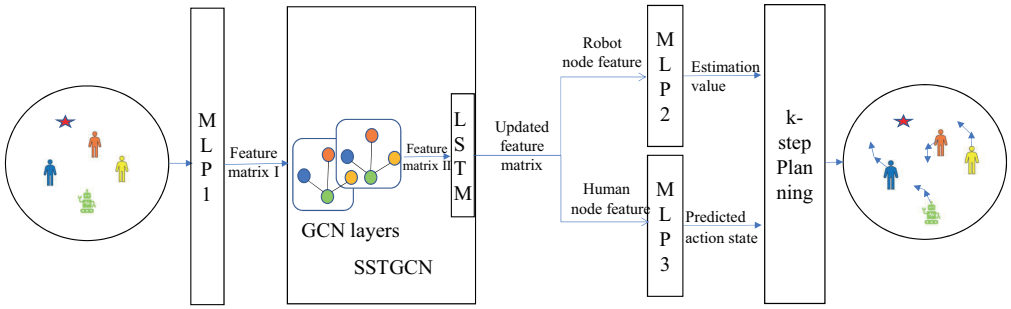

Figure 2. Structure of the model.

3.1. Collision Avoidance with Deep Reinforcement Learning

This part mainly solves the crowd navigation task, that is, the robot can navigate to the target point safely and efficiently in the environment of n humans. Reinforcement learning [34] is a sequential decision problem, which is also expressed as a sequential decision problem in this task [25,35]. We divide the state of the environment into two parts: one part is the information of the robot itself, which can be expressed by a fixed number of states; The other part is other information in the environment, which may contain an unknown number of objects. We can define a vector **s** about robots and other human $\mathbf{s}^{(i)}$,where i is number of human:

$$\mathbf{s} = \left[l_g, v_p.\delta, r\right] \tag{1}$$

$$\mathbf{s}^{(i)} = [p^{(i)}, v^{(i)}, r^{(i)}, l^{(i)}, r^{(i)} + r] \tag{2}$$

where changed the indent, pls confirm. $l_g = ||p_g - p||$ is the distance from the robot to the target point. $l^{(i)} = ||p - p^{(i)}||$ is the distance from robot to human No.i. v_p is the speed of the robot. δ is the heading angle. r is the radius of the detection range. $p^{(i)}$ is the coordinate of the human No.i. $v^{(i)}$ is the moving speed of human No.i. $r^{(i)}$ is the detection radius of the human No.i, and $r^{(i)} + r$ is the distance that the robot and the human No.i can detect each other. The action space of the robot is composed of speed and steering angle, when the speed is v_p, there are 6 courses evenly distributed between $\pm\frac{\pi}{6}$, when the speed is $0.5v_p$ and 0, the heading is selected as $[-\frac{\pi}{6}, +\frac{\pi}{6}]$. We set it this way to simulate the real motion state of the robot. We assume that the robot can always complete the selected action in one time step. One way to find the optimal strategy is to find the optimal value function:

$$V^*\left(s_0^j\right) = E\left[\sum_{t=0}^{T} \gamma^t R\left(s_j^t, \pi^*\left(s_j^t\right)\right)\right] \tag{3}$$

$$\pi^*\left(s_j^{t+1}\right) = \underset{a_t}{argmax}\, R\left(s^t, a_t\right) + \gamma^{\Delta v_p} \int_{s_j^{t+1}} P\left(s_j^t, s_j^{t+1} | a_t\right) V^*\left(s_j^{t+1}\right) ds_j^{t+1} \tag{4}$$

where blue $\gamma \in [0, 1)$ is a discount factor. There are many methods to train the Function (3) offline [34]. The current value function can be implemented as a strategy Function (4), where $R\left(s^t, a_t\right)$ is the reward received after taking action a_t at time t, $P\left(s_j^t, s_j^{t+1} | a_t\right)$ is the transition probability from time t to time $t + \Delta t$, and the moving speed v_p of the robot is the power exponent of the discount factor. The state transition probability model $P\left(s_j^t, s_j^{t+1} | a_t\right)$ is defined by the robot kinematics model. Because the behavior of human depends on their strategies and intentions, the state transition model of the system is unknown. In [13], it is assumed that the state transition function is known in the training process, and the state transition model is shown to be modeled in the testing process, which greatly reduces the complexity of the problem. The main reason is that the robot can search the next state space to solve the navigation problem. In this study, the state transition model is unknown, and the model-based method is used to predict human motion. We define the reward function formula to calculate the reward. When there is a collision or too close to other human, we will punish them.

$$R\left(s^t, a_t\right) = \begin{cases} 1, & if\ p = p_g, \\ -1 + \dfrac{d_{min}}{2}, & if\ 0 < d_{min} < 0.2, \\ -0.25, & if\ d_{min} <= 0, \\ 0, & otherwise. \end{cases} \tag{5}$$

where d_{min} is the closest distance between the robot and other human, p_g is the position of the target point. There are no uncooperative human in the environment. Decision makers only receive their observable states, and do not know their decisions or hidden intentions.

3.2. Graph-Based Crowd Representaion and Learning

The pairwise relationship between agents is very important for robot navigation and prediction of human future actions. We model the crowd as a graph, derive the relationship between humans through relational reasoning, and use the Social Spatial–temporal Graph Convolution Network (SSTGCN) to learn the state representation of robot and human. We mainly build two modules: one is the value estimation module $f_{value}()$, which estimates the value of the current state. One is the state prediction module $f_s()$, which predicts the state of the next time step.

How to represent the interactive coding among all agents is the first problem of the human relations reasoning in the environment. We model robots and other humans as a graph $\mathcal{G} = (V, E)$ [36], where $V = v^i | i \in 1, ..., N+1$ is the node of the graph. There are $N+1$ nodes in total. In graph $\mathcal{G}$, v^i is the set of $v_t^i, \forall t \in 0, \ldots, T$. $E = e_{(i,j)} | \forall i, j \in 1, \ldots, N, N+1$ represents the edge between nodes, and the weight of the space edge $e_{(i,j)} \in E$ is expressed by normalizing the Euclidean distances of nodes i, j. The pairwise relationship between nodes is unknown. When reasoning about the node relationship, the initial value of the node is the state value of the agent $s = s_i^t|_{(i=1,\ldots,N+1)}$. In order to facilitate the representation of the state space, two multilayer perceptrons $f_{robot}()$ and $f_{human}()$ map the potential states belonging to robot and humans into a state space, where the matrix is the characteristic matrix F, the first row of the matrix is the potential state of robot, and the remaining rows are the potential states of human. Here, the point product similarity function in [16] is used to calculate the relationship matrix according to the characteristic matrix F. Similarity function is defined as point product similarity:

$$f(x_i, x_j) = \theta(x_i)^T \varphi(x_j) \tag{6}$$

where $\theta(x_i)^T = W_\theta x_i, \varphi(x_j) = W_\varphi x_j$.

Inspired by the spatiotemporal graph convolution network [28], a new Social Spatial–temporal Graph Convolution Network(SSTGCN) is designed as a graph learning module. SSTGCN adopts the implementation method similar to GCN [15], and convolutes in the time domain. SSTGCN takes features as input, uses the influence of spatial edge aggregation nodes, uses GCN to process spatial features, and LSTM to process time-varying motion features. We will analyze the rationality of the network from spatial correlation and temporal correlation.

The acquisition of complex spatial dependencies is a key problem in trajectory prediction. Traditional convolution neural network (CNN) can obtain local spatial features, but this is only limited to European space. Once irregular graph data are involved, CNN cannot accurately capture spatial dependence. We need to use graph neural network (GNN) which can deal with graph relations. Among them, graph convolution neural network (GCN) has received extensive attention. First, GCN acquisition is used to obtain spatial features from historical data. A two-layer GCN model is represented as follows:

$$f(A, X) = \sigma\left(\tilde{A} Relu\left(\tilde{A} X W_0\right) W_1\right) \tag{7}$$

where X represents the characteristic matrix, A represents the adjacency matrix, and $\tilde{A} = \tilde{D}^{\frac{-1}{2}} \tilde{A} \tilde{D}^{\frac{-1}{2}}$ indicates pretreatment step, $\tilde{A} = A + I$ is a matrix with self connected structure, I is the identity matrix, $\tilde{D}$is the order matrix, $\tilde{D} = \sum_j \tilde{A}_{i,j}$. Our interaction modeling is a second-order interaction. In the output of each layer, we use layer normalization [9], W_0 and W_1 represents the weight matrix of the first layer and the second layer respectively, $\sigma(\cdot)$ and Relu represent activation functions.

The acquisition of temporal correlation is another key problem in trajectory prediction. At present, recurrent neural network (RNN) is the most widely used neural network model for processing sequence data, but it has some shortcomings, such as gradient disappearance, gradient explosion and so on. LSTM [37] is a variant of recurrent neural network. It can solve the above problems, store relevant information in a hidden state, and forget the less important parts. In the SSTGCN model, LSTM is used as the output characteristic matrix

of the processing map convolution layer, which can selectively process the time-varying action characteristics.

Combining graph convolution network and LSTM to form SSTGCN model can obtain the spatiotemporal correlation of trajectory information. As shown in Figure 3, (a) shows that the feature matrix sequence with expanded dimensions generates a feature matrix with new feature information after the graph convolution layer and LSTM layer. (b) shows the specific internal structure of SSTGCN. On the one hand, SSTGCN can be used to obtain the spatial structure between robot and human, human and human, and obtain the spatial correlation, on the other hand, SSTGCN can be used to capture the changes of trajectory information of robot and humans, and obtain the temporal correlation. After SSTGCN propagation, we get the updated state matrix $P^t = F^2$, and $P^t[i,:]$ is the state representation of agent i coding its local interaction.

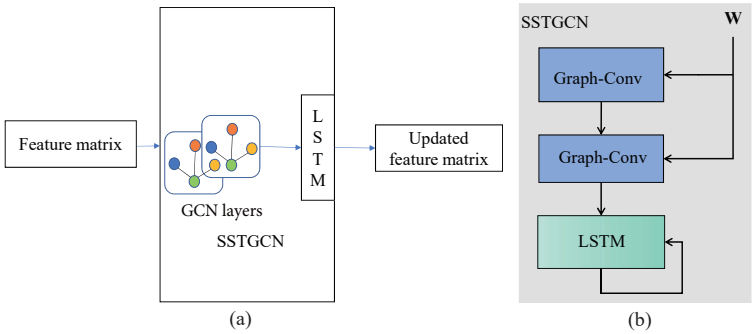

Figure 3. Framework of SSTGCN. (**a**,**b**) show the structure of SSTGCN, consisting of two-layer graph convolution and LSTM. (**a**) shows the flow in which the characteristic matrix is processed. (**b**) shows the internal calculation process of SSTGCN

Value estimation module $f_{value}()$ contains two models: use the above graph model to predict the robot's state representation $P^t[0,:]$, and the other is to predict the value of the obtained state representation $f_v\left(P^t[0,:]\right)$, where f_v uses a forward propagating perceptron MLP. We assume that state s^t passes through action a_t, and we can get $s^{(t+1)}$, which can be calculated. In order to avoid the collision between robot and human, the state prediction module is used to model the interaction between agents and predict the next state. Our state prediction module $f_s()$ also contains two modules: the relationship graph model predicts the relationship between agents and calculates the feature matrix, and then calculates the next state $s'_{i,t+1} = f_a\left(P^t[i,:]\right)$ of agents through the value function module for prediction, $s'_{i,t+1}$ indicates the prediction state of the ith agent at $t+1$ time, f_a perceptron MLP is used. We use Equation (5) to estimate the reward of the agent in the next state, expressed as $R'\left(s^t, a_t, s^{(t+1)}\right)$. By learning the agent state value and predicting the agent trajectory, the robot collision situation can be better solved.

3.3. Robot Forward Planning and Learning

The Monte Carlo tree search method is used for complex search problems. The go game [38] introduces a value network to directly estimate the state of go in order to more conveniently approach the state value of leaf nodes in the tree search process. In this part, we use the value prediction network [33] to simulate future planning, but instead of using the Monte Carlo tree search method for planning, we use a k-step depth planning method to perform prediction at a certain depth using the value prediction network. Given a state s and an action a, the definition of Q value calculation based on k-step depth planning is as follows:

$$Q^k(s,a) = R'\left(s^t, a_t, s^{(t+1)}\right) + \gamma V^k\left(s^{(t+1)}\right) \tag{8}$$

$$V^k(s^t) = \begin{cases} V(s^t), & if\ k = 1, \\ \frac{1}{d}V^1(s^t) + \frac{k-1}{k} max_{a_t} Q^{k-1}(s,a), & k > 1. \end{cases} \quad (9)$$

where $Q^k(s,a)$ represents the Q value of k-step depth prediction, and $V^k\left(s^{(t+1)}\right)$ represents the reward value of the predicted next status value, γ is the discount factor for future rewards, $max_{a_t} Q^{k-1}(s,a)$ represents the average of $k-1$ estimated values. Our planning algorithm includes depth prediction and forward propagation. Depth prediction is a k-step prediction state based on the current state. As shown in Figure 4, in the forward propagation step, the current state estimates $V(s^t)$ and the weighted average value of $max_{a_t} Q^{k-1}(s,a)$ are calculated to calculate the estimated value $V^k(s^t)$ of k-step depth prediction. The value prediction network can be trained according to any existing value based reinforcement learning algorithm, combined with supervised learning for reward and discount. Here we use the training method of literature [33] and adopt the training method based on the above planning method ϵ-greedy policy generates tracks

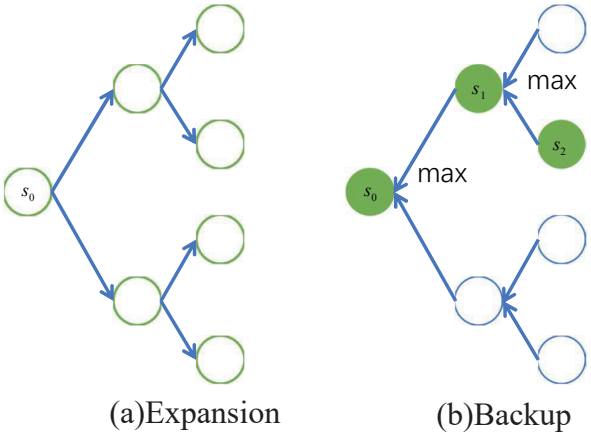

Figure 4. K-step Planning.

3.4. Joint Value Estimation and State Human Learning

Algorithm 1 shows the training code for learning based on value estimation and state prediction. In the line 13 of Algorithm 1, it is shown that the state matrix is transferred to SSTGCN as an input, the relationship matrix is calculated and added as a weight matrix to the two GCN convolution layer for calculation, and LSTM is responsible for feature extraction of the updated feature matrix. After several tests and referring to other research work, the appropriate matrix dimension is selected. All required parameters are set in the configuration file for training. In the training, firstly, the network is supervised and trained to imitate the demonstration of OCRA strategy. The latter will drive the robot through the crowd to reach the destination, initialize the model from the experience collected, and then use reinforcement learning to train the strategy. Due to the sparse rewards in the navigation process, imitation learning is very important in initializing the model, otherwise reinforcement learning strategy will not converge. We use RL to train $f_{value}()$ and using supervised learning training $f_s()$. Using SSTGCN, we can integrate time-varying action information and spatial information into trajectory prediction, and learn more accurate value functions than these strategies [13,18].

Algorithm 1. This table shows the training process of the algorithm, which includes Imitation learning and Reinforcement learning

Deep learning for $f_{value}()$ **and** $f_s()$
/* Imitation learning */
Input: collected state-value pair(s_{nav}, f_{value})
for epoch $\leftarrow$ 1 to num of epochs do
  $\tilde{f}_{value} = F(s_{nav}, w)$
  e=MSE$(f_{value}, \tilde{f}_{value})$
  w=backprop(e, w)
end
/* Reinforcement learning */
for episode $\leftarrow$ 1 to num of episodes do
while not reach goal, collide or timeout do
  $a_t \leftarrow argmax_{(a_t \in A)} \tilde{R}\left(s^t, a_t, s^{(t+1)}\right) + \gamma^{\left(\Delta t v_p\right)} V^k\left(s^t\right)$ where $s^{(t+1)} = f_s\left(s^t, a_t\right)$
  Get r_t and $s^{(t+1)}$ after execute a_t
  Store transition $\left(s^t, a_t, r_t, s^{(t+1)}\right)$ in B
  Sample random minibatch of transition $\left(s^t, a_t, r_t, s^{(t+1)}\right)$ in B
  Transfer the state matrix to SSTGCN for value estimation
  $y_i = r_i + \gamma^{\left(\Delta t v_p\right)} V^k\left(s_{(i+1)}\right)$
  Minimizing $||f_{value}(s_i - y_i)||$ for updating f_{value}
  Minimizing $||f_s(s_i, a_i)||$ for updating f_s
  Update target value network $\tilde{f}_{value} \leftarrow f_{value}$
end for
return f_{value}, f_s

4. Experiments

In this section, we carry out relevant experimental verification on our model. In Section 4.1, we describe the experimental design scheme and parameter settings. We analyze the experimental data in detail in Section 4.2. In Section 4.3, we make a quantitative evaluation of the model and show that freezing behavior will not occur.

4.1. Simulation Environment and Experimental Parameter Setting

We use Crowd-Nav to simulate the environment [13]. The hidden units of $f_{robot}()$, $f_{human}()$, $f_{value}()$, $f_s()$ have dimensions $(64, 32)$, $(64, 32)$, $(150, 100, 100)$, $(64, 32)$. The learning rate is 0.001. The discount factor γ is set to be 0.95.The simulation environment is a circular environment with a radius of 4 m. In the environment, humans use ORCA strategy with parameters from Gaussian distribution to perform diversified behaviors. The x and y coordinates of humans are subject to random disturbance. The maximum speed of the robot is 1 m/s, and the coordinates of target points have been set. The robot is invisible to human beings in order to simulate the natural behavior of human beings. The navigation strategies in the experiment include the navigation strategy ORCA based on social model, the deep reinforcement learning strategies LSTM and SARL based on non graph neural network, the deep reinforcement learning strategy MP-RGL-Multistep based on graph neural network, and the model SSTGCN-V-Learning (this strategy) of this study. The experiment is divided into three groups of experiments. The first group of experiment environment is 3 humans and 1 robot, 5 humans and 1 robot, 10 humans and 1 robot. The depth of the depth planning tree is 1, and the navigation strategy is SSTGCN-V-Learning. The second group of experiments included 3 humans and 1 robot, 5 humans and 1 robot, 10 humans and 1 robot. The depth of the depth planning tree is 1. The navigation strategies were ORCA, LSTM, SARL, MP-RGL-Multistep and SSTGCN-V-Learning. The third group of experimental environments are 3 humans and 1 robot, 5 humans and 1 robot, 10 humans and 1 robot. The depth of the depth planning tree is 1 and 2, and the navigation strategy is

SSTGCN-V-Learning. The purpose of the first group of experiments is to test the crowd navigation function of the navigation strategy studied this time. The second group of experiments is to test the navigation performance of the model, and also use this group of experiments to analyze the robustness of two graph learning navigation strategies to number of human changes. The third group of experiments is to test the influence of predicted depth on navigation performance. Finally, we analyze whether freezing behavior will occur when robots and humans are about to meet.

In order to ensure fairness, all experiments are implemented on the same platform. Only the models are different. All models are evaluated with 200 random test cases. The indicators of the evaluation model are success rate, collision rate, navigation time and average reward. The navigation success rate represents the success rate in 200 tests, the collision rate represents the collision rate in 200 tests, the navigation time is the average time in 200 tests, and the total reward is the sum of the rewards in 200 tests. Set the training rounds of LSTM, SARL, MP-RGL-Multistep and SSTGCN-V-Learning to 4000 episodes and batch size to 5. All codes are based on Python. Hardware platform parameters are shown in Table 1.

Table 1. Hardware and Software Platform Parameters.

Hardware and Software	Parameter
ARM	32GB
CPU	Intel Xeon Silver 4210
GPU	NVIDIA GeForce RTX 3080 with 10 GB
Operating System	Ubuntu 18.04

4.2. Navigation Obstacle Avoidance Performance Test

This module contains four groups of comparative experiments, which are the benchmark experiment, performance comparison experiment, depth planning tree depth influence comparison experiment and robustness experiment to the change of human number. Table 2 shows the success rate, collision rate, navigation time and average reward for different strategies and different human numbers. The basic performance of the strategy is tested in the environment where the depth and width of the k-step planning tree are 1 and the number of humans is 3, 5 and 10. Figure 5 shows the navigation examples in different scenarios. (a) shows the scene when the number of humans is 3, (b) shows the scene when the number of humans is 5, and (c) shows the scene when the number of humans is 10. From the data in Table 2, it can be seen that the robot navigation success rate is at a high level.

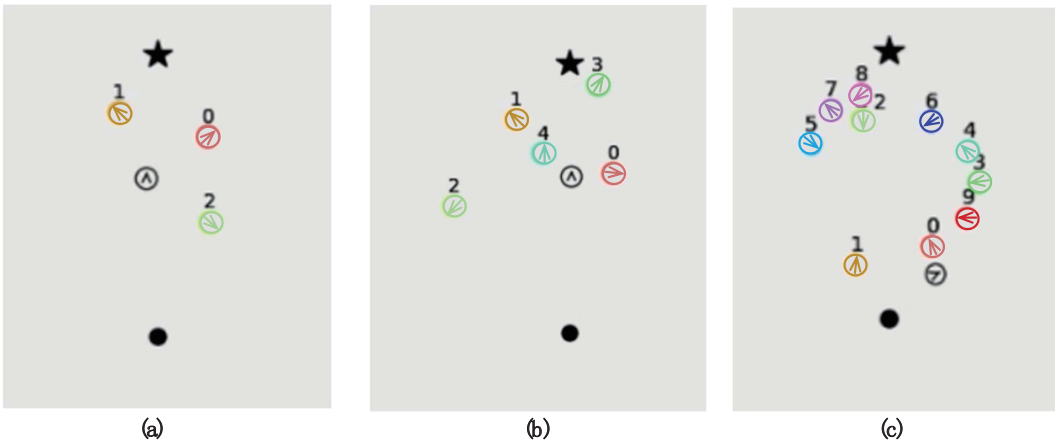

Figure 5. The black circle represents mobile robot. Colored circles represent human. The black star represents the target point. (**a**) shows robot navigation scene when the number of humans is 3, (**b**) shows robot navigation scene when the number of humans is 5, and (**c**) shows robot navigation scene when the number of humans is 10.

The performance comparison experiment was carried out in the environment where the depth of the k-step planning tree were 1 and the number of human was 3, 5 and 10. The violation of the preventive assessment of ORCA was violated in the invisible environment. Since local collision free operation could not be guaranteed, the navigation success rate of this method was low. The reward obtained by ORCA navigation strategy also shows the poor navigation performance of this strategy when other intelligent body pose information is not known. In the Figure 6, (a) shows the navigation success rate when the number of human is 3, (b) the navigation success rate when the number of human is 5, (c) shows the navigation success rate when the number of human is 10. In the Table 2, human number is the number of human in the crowd, success rate is the proportion of robots reaching the target point without collision, collision rate is the collision between robots and humans, navigation time is the average time for robots to navigate to the target point, and total reward is the reward value for robots to navigate to the target point. It can be seen from the data shown in Figure 6 and Table 2 that the best model is SSTGCN-V-Learning model, ORCA is the worst performing model. Compared with MP-RGL-Multistep, SSTGCN has more advantages in navigation effect than GCN. The main reason is that LSTM uses hidden states to capture these time-varying motion characteristics by using the updated feature matrix of graph convolution network to improve the accuracy of decision-making. However, the model with the best performance does not appear without collision. The main reason is that humans cannot see the robot during walking. Robot navigation uses the predicted trajectory of humans, not the real trajectory of the next time, which makes collision inevitable.

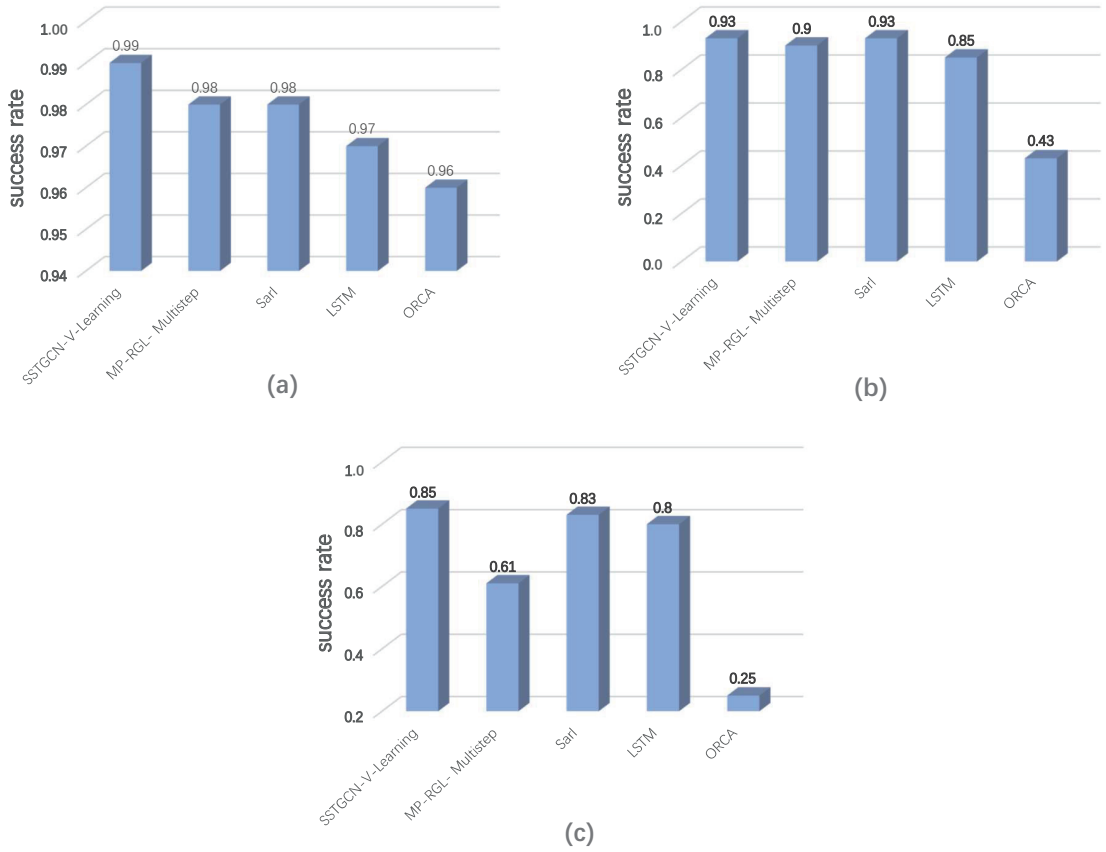

Figure 6. Navigation success rate of related strategies in different scenarios. (**a**) shows the navigation success rate when the number of human is 3. (**b**) shows the navigation success rate when the number of human is 5. (**c**) shows the navigation success rate when the number of human is 10.

Table 2. Performance comprison among the crowd navigation algorithms.

Method	Human Number	Success Rate	Collision Rate	Navigation Time (s)	Total Reward
SSTGCN-V-Learning	3	0.99	0.01	10.12	0.3305
	5	0.93	0.03	11.49	0.3017
	10	0.85	0.13	14.01	0.1150
MP-RGL- Multistep	3	0.98	0.02	10.67	0.3104
	5	0.90	0.05	11.05	0.2920
	10	0.61	0.07	16.44	0.0754
SARL	3	0.98	0.01	10.51	0.3148
	5	0.93	0.05	12.48	0.2203
	10	0.83	0.10	14.38	0.1348
LSTM	3	0.97	0.02	10.91	0.3065
	5	0.85	0.14	10.98	0.1895
	10	0.80	0.19	13.38	0.1044
ORCA	3	0.96	0.03	10.87	0.3127
	5	0.43	0.57	10.93	0.0615
	10	0.25	0.30	17.66	−0.1499

The depth test of k-step planning tree (i.e., the test of predicted steps) is carried out in the environment with different human numbers. The prediction of k-step number will affect the decision-making of the robot and thus the navigation performance. In the Figure 7, H represents the number of human in the environment, and D represents the depth of the depth planning tree, that is, the number of prediction steps. In the Table 3, Depth refers to the number of prediction steps, success rate refers to the navigation success rate, navigation time refers to the average navigation time, and training time refers to the time required for the training strategy. This parameter is related to the equipment used. According to Figure 7, it can be seen that the navigation success rate and navigation time increase slightly with the increase of the predicted step number. The main reason is that a k-step depth planning method is usefull in this model, and the value prediction network is used to perform the prediction of a certain depth. However, according to what we observed in the experiment, it is not that the more prediction steps, the better. With the increase of prediction steps, the training time will also increase significantly. This is because the planning algorithm includes depth prediction and forward propagation. If a larger depth is selected, these two calculation processes will consume more time. Therefore, selecting a suitable depth will help to improve performance and save training time.

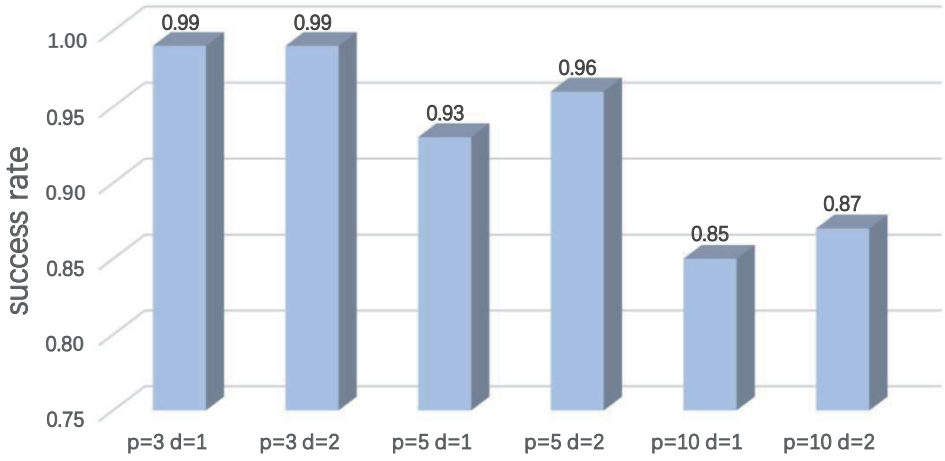

Figure 7. Navigation success rate in different environments.

Table 3. Influence of Depth on navigation performance of SSTGCN-V-Learning.

Human Number	Depth	Success Rate	Navigation Time (s)	Training Time (h)
3	1	0.99	10.12	19
	2	0.99	9.79	30
5	1	0.93	11.49	22.8
	2	0.96	11.28	37
10	1	0.85	14.01	29.25
	2	0.87	13.86	49

The robustness comparison experiment environment is mainly to compare MP-RGL-Multistep and SSTGCN-V-Learning when the number of human is 3, 5 and 10. It is not difficult to find from Table 2 that the navigation success rate of these two navigation strategies decreases to varying degrees with the increase of the number of human, but the performance of MP-RGL-Multistep strategy decreases significantly, and the success rate declines more rapidly in the same environment as SSTGCN-V-Learning strategy, MP-RGL-Multistep strategy is sensitive to changes in the number of human. According to the experimental analysis, the MP-RGL-Multistep strategy can not well predict the behavior trend between human when there are many human, resulting in the inability to guide the robot to quickly move to the target point. More often, the robot generates other actions according to the predicted complex human behavior, resulting in timeout. The SSTGCN used in this strategy not only handles spatial features, but also handles long-term dependency, It can be seen from the total reward in Table 2 that this strategy is more accurate than MP-RGL-Multistep in predicting the performance of human behavior.

4.3. Qualitative Evaluation

We also studied the freezing behavior of robots when approaching human. Figures 8 and 9 show the behavior of robot when they are near humans. Figure 8 shows the performance of the navigation strategy when robot approaches human in the scene with 3 and 10 humans. In the Figure 8, black star indicates the target point to be reached. (a) and (b) is the performance of the robot when facing the freezing problem at one step depth in the scene with 3 humans, showing that when it was about to meet human No.2, the robot made a 90° left turn to try to bypass human No.2. (c) and (d) is the performance of the robot when facing the freezing problem in a one-step depth in the scene where the number of humans is 10. It shows that when the robot is about to meet human No.9, the robot makes an evasion action in advance according to the predicted action of human No.9, rotating 135° to the left to avoid human No.9.

In the Figure 9, (a) and (b) shows that when the robot is about to meet human No.3, it turns 90° to the right to try to bypass human No.3. The robot will try to bypass human No.3 in front of human No.3, which will make the robot step into the comfortable range of human walking and get a punishment because it does not conform to the social code of conduct. In (c) and (d), the robot and human No.3 are about to meet. But due to two-step planning, it is predicted that a higher reward value can be obtained by passing through the position with the opposite movement direction of No.3 (that is, the position just passed by No.3). Although they all face the situation of meeting human, the robot does not appear freezing behavior and has reached the target point. From the side, it shows that the robot can use this strategy to avoid freezing behavior when navigating, and the two-step planning has better performance than the one-step planning.

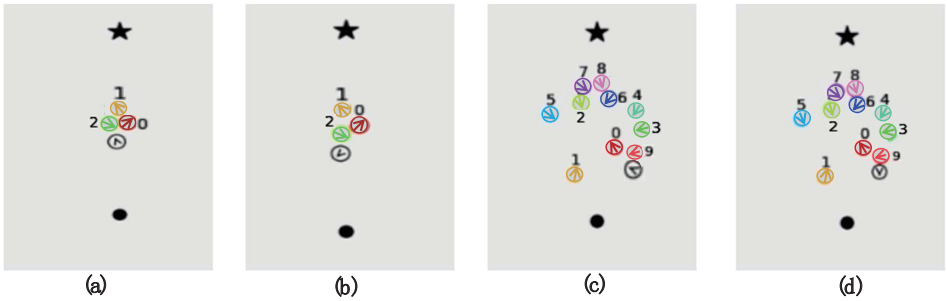

Figure 8. (**a**–**d**) depict how the robot acts in different crowd navigation scenes, among which (**a**,**b**) show the situation with 3 passers-by while (**c**,**d**) show the situation with 10 passers-by.

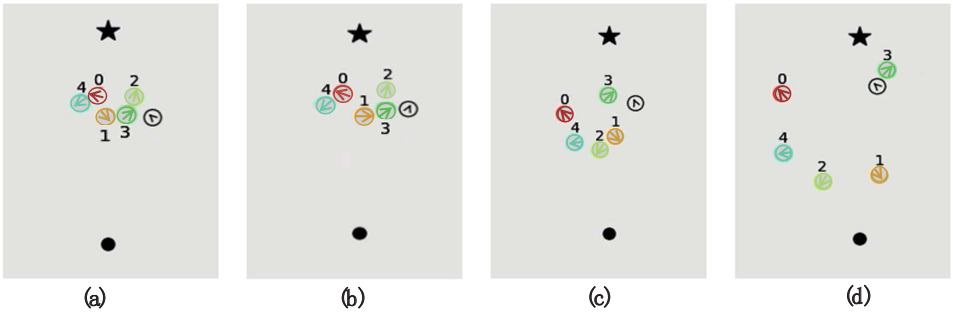

Figure 9. The robot acts by K-step tree planning strategy with different depths. In (**a**,**b**), the depth is 1 while it is 2 in (**c**,**d**).

5. Discussion

In this study, we have implemented a deep reinforcement learning model based on SSTGCN, which is used to solve safe navigation of robots in the crowd. In graph learning module, latent features are taken advantage to reason about the relations among agents, and SSTGCN is used to calculate the interaction features, where the graph convolution network layer is used to update the spatial features and the LSTM layer is used to capture the time-varying action characteristics. In addition, value estimation module calculates state representation and state prediction module predicts the next state. In the robot forward planning module uses k-step planning to estimate the quality of state. The model make full use of the predicted human action trajectory for k-step planning to complete the path planning of the robot.

The experiment results show that our model has great advantages over the model-based navigation strategy. ORCA strategy is one of them [22]. When the number of human increases, the navigation performance of the model-based strategy decreases sharply, and robot freezing behavior often occurs. In the comparison of strategies based on learning, our model is also better than other models, such as LSTM and SARL [13]. The main reason is that our model uses graph neural network to model human–robot interaction, which better reflects the association between pose information, and the depth planning tree of our model completes k steps prediction which can improve the accuracy of robot path planning in the future. Compared with MP-RGL-Multistep [18], which also uses graph neural network, our model has obvious advantages in robustness. The LSTM layer in our model takes advantage of hidden states to capture these time-varying motion characteristics by updating the characteristic matrix of graph convolution network, so as to improve the accuracy of decision-making.

The SSTGCN used in this model is composed of convolution layer and LSTM layer. There are still some disadvantages in processing feature information. During the experiment, we find that with the increase of the number of humans, the training time increases more. After analysis, it is found that the model take a long time to calculate the graph convolution neural network and LSTM . A feasible solution of this problem is to design

a lighter neural network with processing time characteristics to replace the role of LSTM layer in SSTGCN.

6. Conclusions

In this paper, we design a new structure of SSTGCN for the deep reinforcement learning to solve the crowd navigation . It is an advantageous to use the SSTGCN processing the human–robot interaction model because of the association relationship of spatiotemporal information. The model can make full use of the predicted human action trajectory for deep k-step planning to complete the robot path planning. Our model is superior to the baseline model and performs well in the face of robot freezing behavior. We focus on the theoretical research and preliminary verification of the algorithm in the current phase. Therefore, physical experiment has not been carried out . This will become the focus of our next work. In the future, we will train the robot in the environment with static obstacles and humans, and realize the robot navigation task in the complex real environment.

Author Contributions: Conceptualization, Y.L.; methodology, Y.L.; software, Y.L.; validation, Y.L.; formal analysis, Y.L., J.H. and X.R.; investigation, Y.L.; resources, J.H.; data curation, Y.L.; writing—original draft preparation, Y.L.; writing—review and editing, J.H. and Y.L.; visualization, Y.L.; supervision, J.H.; project administration, J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Key Research and Development Program Project (No. 2020YFB1005900)

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bai, H.; Cai, S.; Ye, N.; Hsu, D.; Lee, W.S. Intention-aware online POMDP planning for autonomous driving in a crowd. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 454–460.
2. Kretzschmar, H.; Spies, M.; Sprunk, C.; BurgardAuthor, W. Socially compliant mobile robot navigation via inverse reinforcement learning *Int. J. Robot. Res.* **2016**, *35*, 1289–1307. [CrossRef]
3. Vemula, A.; Muelling, K.; Oh, J. Modeling cooperative navigation in dense human crowds. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1685–1692.
4. Fox, D.; Burgard, ; W.; Thrun, S. The dynamic window approach to collision avoidance. *IEEE Robot. Autom. Mag.* **1997**, *4*, 23–23. [CrossRef]
5. Phillips, M.; Likhachev, M. Sipp: Safe interval path planning for dynamic environments. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 5628–5635.
6. Kim, S.; Guy, S.J.; Liu, W.; Wilkie, D.; Lau, R.W.; Lin, M.C.; Manocha, D. Brvo: Predicting pedestrian trajectories using velocity-space reasoning. *Int. J. Robot. Res.* **2015**, *34*, 201–207. [CrossRef]
7. Trautman, P.; Krause, A. Unfreezing the robot: Navigation in dense, interacting crowds. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 797–803.
8. Ferrer, G.; Garrell, A.; Sanfeliu, A. Robot companion: A social-force based approach with human awareness-navigation in crowded environments. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 1688–1694.
9. Ferrer, G.; Sanfeliu, A. Behavior estimation for a complete framework for human motion prediction in crowded environments. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 5940–5945.
10. Mehta, D.; Ferrer, G.; Olson, E. Autonomous navigation in dynamic social environments using Multi-Policy Decision Making. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 1190–1197.
11. Chen, Y.F.; Liu, M.; Everett, M.; How, J.P. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In Proceedings of the2017 IEEE International Conference on Robotics and Automation (ICRA),Singapore, 29 May–3 June 2017; pp. 285–292.

12. Everett, M.; Chen, Y.F.; How, J.P. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 3052–3059.
13. Chen, C.; Liu, Y.; Kreiss, S.; Alahi, A. Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 6015–6022.
14. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2017**, *20*, 61–80. [CrossRef]
15. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
16. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-local neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7794–7803.
17. Grover, A.; Al-Shedivat, M.; Gupta, J.; Burda, Y.; Edwards, H. Learning policy representations in multiagent systems. Proceedings of the 35th International Conference on Machine Learning, PMLR, Stockholm Sweden, 10-15 July 2018; pp. 1802–1811.
18. Chen, C.; Hu, S.; Nikdel, P.; Mori, G.; Savva, M. Relational graph learning for crowd navigation. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October–24 January 2021; pp. 10007–10013.
19. Helbing, D.; Molnar, P. Social force model for pedestrian dynamics. *Phys. Rev. E* **1995**, *51*, 4282. [CrossRef] [PubMed]
20. Ferrer, G.; Zulueta, A.G.; Cotarelo, F.H.; Sanfeliu, A. Robot social-aware navigation framework to accompany people walking side-by-side. *Auton. Robot.* **2017**, *41*,775–793. [CrossRef]
21. Van den Berg, J.; Lin, M.; Manocha, D. Reciprocal velocity obstacles for real-time multi-agent navigation. In Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, USA, 19–23 May 2008; pp. 1928–1935.
22. Berg, J.V.D.; Guy, S.J.; Lin, M.; Manocha, D. Reciprocal n-body collision avoidance. In *Robotics Research*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 3–19.
23. Long, P.; Liu, W.; Pan, J. Deep-learned collision avoidance policy for distributed multiagent navigation. *IEEE Robot. Autom. Lett.* **2017**, *2*, 656–663. [CrossRef]
24. Tai, L.; Zhang, J.; Liu, M.; Burgard, W. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 1111–1117.
25. Chen, Y.F.; Everett, M.; Liu, M.; How, J.P. Socially aware motion planning with deep reinforcement learning. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1343–1350.
26. Chen, Y.; Liu, C.; Shi, B.; Liu, M. Robot Navigation in Crowds by Graph Convolutional Networks With Attention Learned From Human Gaze. *IEEE Robot. Autom. Lett.* **2020**, *5*, 2754–2761. [CrossRef]
27. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
28. Ibrahim, M.S.; Mori, G. Hierarchical relational networks for group activity recognition and retrieval. In Proceedings of the European Conference on Computer Vision (ECCV), Munich , Germany , 8–14 September 2018; pp. 721–736.
29. Vemula, A.; Muelling, K.; Oh, J. Social attention: Modeling attention in human crowds. In Proceedings of the 2018 IEEE international Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 4601–4607.
30. Kipf, T.; Fetaya, E.; Wang, K.C.; Welling, M.; Zemel, R. Neural relational inference for interacting systems. Proceedings of the 35th International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 2688–2697.
31. Mohamed, A.; Qian, K.; Elhoseiny, M.; Claudel, C. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 14424–14432.
32. Finn, C.; Levine, S. Deep visual foresight for planning robot motion. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 2786–2793.
33. Oh, J.; Singh, S.; Lee, H. Value prediction network. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
34. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018; pp. 1–6.
35. Everett, M.; Chen, Y.F.; How, J.P. Collision avoidance in pedestrian-rich environments with deep reinforcement learning. *IEEE Access* **2021**, *9*, 10357–10377. [CrossRef]
36. Battaglia, P.W.; Hamrick, J.B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. Relational inductive biases, deep learning, and graph networks. *arXiv* **2018**, arXiv:1806.01261.
37. Graves, A. Long short-term memory. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Berlin, Germany, 2012; pp. 37–45.
38. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–489. [CrossRef] [PubMed]