2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

SoLo T-DIRL: Socially-Aware Dynamic Local Planner based on Trajectory-Ranked Deep Inverse Reinforcement Learning

Yifan Xu, Theodor Chakhachiro, Tribhi Kathuria, and Maani Ghaffari

***Abstract*— This work proposes a novel framework for socially-aware robot navigation in dynamic, crowded environments using a Deep Inverse Reinforcement Learning. To address the social navigation problem, our multi-modal learning based planner explicitly considers social interaction factors, as well as social-awareness factors, into the DIRL pipeline to learn a reward function from human demonstrations. Moreover, we propose a novel trajectory ranking score using the sudden velocity change of pedestrians around the robot to address the sub-optimality in human demonstrations. Our evaluation shows that this method can successfully make a robot navigate in a crowded social environment and outperforms the state-of-art social navigation methods in terms of the success rate, navigation time, and invasion rate.**

I. Introduction

Robotic technology has enabled the development of Socially Assistive Robots (SARs) to assist humans in various social contexts [1]. Recently, service and guide robots have been deployed in museums [2], shopping malls [3], and airports [4], and are becoming an irreplaceable part of our daily life. Assistive robots are required to navigate in public spaces among people in a safe and socially acceptable manner. This problem is known in the literature as social navigation [5], [6].

The main challenge in social navigation is being able to plan safe and socially acceptable robot plans that can take into account the underlying social dynamics of humans in the scene [7]. As people, we infer these social dynamics implicitly when navigating in dense, dynamic environments. This work uses an explicit representation of these different social features that are inferred from human demonstrations and uses to create robot plans.

Some of the previous works, such as [8], treat everything on the way as obstacles to be avoided. More recent navigation methods, such as [9] and [10], use a socially-aware reward function in a deep reinforcement learning framework to accomplish planning tasks while maintaining a predefined social distance away from humans. However, such methods do not take into account other unknown obstacles(e.g., chairs and tables) in the environment into their reward function design. As a result, these methods cannot fully support navigation in complex, dynamic social environments. Inverse Reinforcement Learning (IRL) in socially-aware navigation has been applied to indoor service robots [7], [11]. Different from classical methods, IRL-based planners can exploit multi-modal reward function learning and use human demonstrations to train the reward network, thereby overcoming the inconvenience and inefficiency of hand-designed reward functions.

Funding for M. Ghaffari was provided by NSF Award No. 2118818.

The authors are with the University of Michigan, Ann Arbor, MI 48109, USA. {yfx, teochiro, tribhi, maanigj}@umich.edu

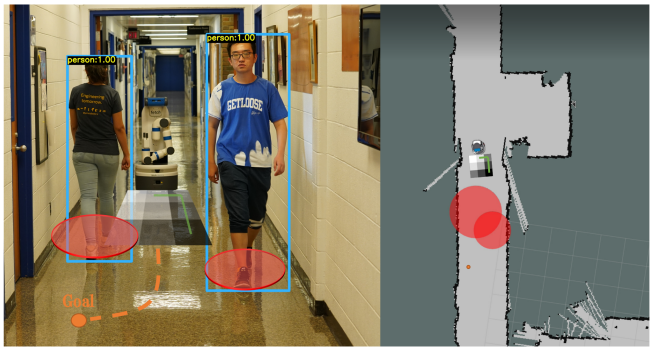

Fig. 1: The left figure shows our Fetch robot navigating in the corridor of the Naval Architecture and Marine Engineering building at the University of Michigan, while the right figure is the corresponding RViz visualization of the augmented map. Red circles represent the personal space of the two people. Blue bounding boxes are the output detection from YOLOv6. Grey cells beneath the robot constitute the reward map. The orange dashed line is the global path.

However, the state-of-the-art IRL-based social navigation methods such as [7], [11] assume the human demonstrations are optimal. Because people cannot access all information around the robot and perfectly infer pedestrians' intentions, this assumption is not always satisfied in practice. In fact, even in some sparsely crowded situations ensuring optimal behavior, the performance of these models cannot outperform the demonstrations.

In this paper, we propose a novel socially-aware dynamic local planner SoLo T-DIRL using Trajectory-ranked Maximum-Entropy Deep Inverse Reinforcement Learning (T-MEDIRL) [12]. Inspired by [7], [11], we extract features from the social dynamics between robots and pedestrians and unknown obstacle information in front of the robot. After collecting demonstrations by an operator, we use the sudden velocity change as a novel ranking score to address the sub-optimality in human demonstrations. The proposed planning pipeline can take social dynamics and complex environmental information into account.

In particular, this work has the following contributions.

1) A novel socially-aware T-MEDIRL-based local planner with multiple feature layers that can handle complex, indoor dynamic, and crowded scenarios.
2) A novel trajectory ranking score using the sudden velocity change of pedestrians around the robot to address

the sub-optimality in human demonstrations.

3) The system has been evaluated and implemented in a ROS [13] pipeline and is available for download at https://github.com/UMich-CURLY/Fetch_IRL

II. Related Work

A. Socially-aware Navigation

In social environments, it is not sufficient for robots to plan a collision-free path [14], [15]. The navigation behaviors should consider the context of interacting with humans, such as social norms [16]. To support this claim, [9] proposes a model-based method that employs the social force model [17] in robot navigation. In [18], besides considering the pedestrian model, the authors also take interactive social information about human–objects and human group interactions into planner design. Recent methods of local planner design such as [10], [19], [20] use a socially-aware reward function in a deep reinforcement learning framework to accomplish planning tasks while maintaining a social distance away from humans. However, one of the limitations of these methods is that they fail to integrate other unknown obstacles (e.g., chairs and tables) in the environment into their reward function design. Another downside is using a single-feature reward function that does not generalize to a more complex problem, making their performance highly dependent on the accuracy of human detection in real-life applications. To overcome these limitations, we propose a multi-modal local planner that considers both the socially-aware factors and static as well as dynamic obstacle information, so that a complex navigation problem in a social environment can be resolved.

B. IRL-based Navigation

Instead of using a handcrafted reward function, MEDIRL-based methods generate rewards from demonstrations that can overcome the inaccuracy and incompleteness of forming rewards by hand. This is especially useful for social navigation problems where handcrafting a reward for interaction is hard. [21] presents MEIRL based on the principle of maximum entropy [22] and IRL [23] to find the policy with the highest entropy subject to feature matching. [11] proposes the structure of the IRL-based local planner in a crowded environment. [7] successfully combines a learned human cooperative navigation model and MEDIRL into socially compliant mobile robot navigation. However, the IRL-based methods highly rely on the quality of human demonstrations, resulting in bad navigation performance if the latter is not optimal. [12] proposes an energy-based trajectory ranking loss in the training process of MEDIRL to eliminate the effect of non-optimal and sub-optimal demonstrations. For our proposed method, we extend the usage of T-MEDIRL to the socially-aware navigation field using the sudden velocity change of nearby people as the trajectory ranking loss. As a result, our navigation model not only considers socially-aware factors but is also less reliant on the quality of demonstrations.

C. Human Trajectory Prediction

In this work, a human trajectory prediction module is used as one of our feature layers to navigate the crowd safely and plan a socially acceptable path. Some physics-based methods generate the predicted trajectory by modeling people's motion using Newton's second law [24]. [25] predicts using a linear walking model and current velocity. [26] uses a cyclist dynamic model which contains driving force and resistant force from acceleration, inclination, rolling, and air to predict people's trajectory. However, these model-based methods rely on the model's accuracy and ignore the randomness of people's walking speed and directions. Recently, a pattern-based method that can predict the trajectory by discovering statistical behavioral patterns from the pedestrian walking dataset is becoming popular. [27] proposed to predict trajectory by training adversarially against a recurrent discriminator. We use Social LSTM [28] in one of our feature layers to learn general human movement and predict future trajectories for our proposed navigation method.

This work creates a dynamic multi-modal local planner to handle socially-aware navigation tasks in a crowded environment. The feature map used in our local planner encodes different social dynamics information containing social distance and people's walking intentions. Moreover, we design sudden velocity change as the trajectory ranking loss added to the training process to rank the demonstrations and improve the performance of our reward model.

III. Methodology

We separate our work into offline training and online planning. We construct a multi-layer feature map from sensory information in the training part. Then we set different navigation goals and navigate the robot through the crowd to get demonstrations. After getting more than 100 demonstrations, we apply the T-MEDIRL algorithm to train our IRL model until convergence. In the planning process, we get the reward for each state for policy and value iteration to get a local policy inside the grid. We then execute the policy and navigate the crowd using a PID controller.

A. Problem Statement

We model the navigation problem as a three-layer process. The first layer is a planner generating the order of navigation goals that take the shortest time and distance by solving an optimization problem [29]. The second layer is a global planner outputting the waypoints from the robot to each goal using a Voronoi-based planner [30]. The third layer is our dynamic, socially aware local planner, which controls the robot going through each waypoint while avoiding collisions with all obstacles and humans.

The navigation structure of our local planner can be modeled as a Markov Decision Process (MDP). An MDP can be defined as a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma\}$, where $\mathcal{S}$ represents the state space of the system, $\mathcal{A}$ represents the action space, $\mathcal{T}$ is the transition probability, r is the reward function and $\gamma \in [0, 1)$ is the discount factor. The objective

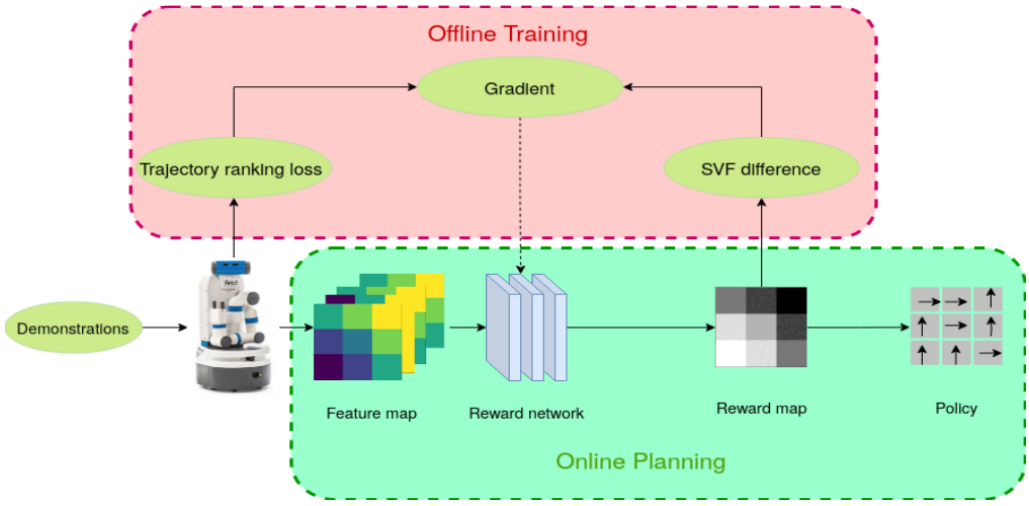

Fig. 2: The structure plot of the system. From the plot, our system is divided into training and planning parts. The gradient for training is constructed using trajectory ranking loss and SVF difference. The reward network we use is a 3-layer fully-connected network whose input is a feature map, and the output is a reward map.

of the MDP is to find an optimal policy $\pi^* : \mathcal{S} \to \mathcal{A}$ that maximizes the expected future reward:

$$\pi^* = \arg\max \ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r\left(t \mid \pi\right)\right]. \tag{1}$$

For our problem setting, the state space, $\mathcal{S}$, is defined as local grid cells in front of the robot. Each cell in the grid cells is one state in the state space. Thus, the number of states equals the number of local grid cells. For example, for a $m \times m$ grid, we will have m^2 states. The action space, $\mathcal{A}$, is defined as a set of discrete actions that move from one cell to its adjacent cells, i.e., $\mathcal{A} := \{\uparrow, \downarrow, \leftarrow, \rightarrow, stop\}$. The action set can be easily extended to eight actions, including diagonal directions. The transition probability $\mathcal{T}(s, a, s')$ is deterministic in our problem setting. In (2), the reward of agents passing from one state to another is calculated using a fully-connected neural network f with a set of parameter θ whose input is the feature vector ϕ from that state. We define the feature vectors $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^n$ that map a state and an action to an n-dimensional feature vector.

$$r(s, a) = f(\phi(s, a); \theta). \tag{2}$$

B. T-MEDIRL

MEDIRL is used to find a set of weights that maximize the total reward of demonstrations. The IRL problem can be framed as MAP estimation, which maximizes the joint posterior distribution of observing expert demonstrations, $\mathcal{D}$, under a given reward structure and of the model parameters θ [31]. The joint log-likelihood $\mathcal{L}$ is given by (3).

$$\begin{aligned} \mathcal{L}(\theta) &= \log P(\mathcal{D}, \theta | r) = \log P(\mathcal{D}|r) + \log P(\theta) \\ &= \mathcal{L}_{\mathcal{D}} + \mathcal{L}_{\theta}. \end{aligned} \tag{3}$$

Equation (3) contains a data term $\mathcal{L}_{\mathcal{D}}$ and a model regularizer $\mathcal{L}_{\theta}$. After applying the chain rule, the MEDIRL gradient can be written as (4)

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta} &= \frac{\partial \mathcal{L}_{\mathcal{D}}}{\partial \theta} + \frac{\partial \mathcal{L}_{\theta}}{\partial \theta} = \frac{\partial \mathcal{L}_{\mathcal{D}}}{\partial r} \cdot \frac{\partial r}{\partial \theta} + \frac{\partial \mathcal{L}_{\theta}}{\partial \theta} \\ &= (\mu_{\mathcal{D}} - \mathbb{E}[\mu]) \cdot \frac{\partial r}{\partial \theta} + \frac{\partial \mathcal{L}_{\theta}}{\partial \theta}. \end{aligned} \tag{4}$$

$\mu_{\mathcal{D}}$ is the average State Visitation Frequencies (SVF) calculated from the training data. $\mathbb{E}[\mu]$ is the expected SVF calculated from the current prediction model.

Note that in the existing MEDIRL framework, the gradient for training can only come from the difference between the SVF of demonstrations and the current prediction model. Hence, it highly relies on the quality of the demonstrations. Inspired by [12], we add a trajectory loss term to the MEDIRL framework to solve this problem. As shown in Fig. 2, given a set of demonstrations, the objective is to find a policy with higher rewards for the high-ranked demonstrations. Instead of using one demonstration to train in one epoch, we use two demonstrations randomly chosen from the training dataset and add a pair-wise trajectory ranking loss as

$$\mathcal{L}_{ij} = -\sum_{r_i < r_j} \log \frac{\exp r_j}{\exp r_i + \exp r_j}, \tag{5}$$

where, r_i, r_j are the trajectory rewards for trajectories i, j respectively. The trajectory reward relies on the robot's behavior when it passes through the crowd since a lack of social awareness can cause disruptions among the crowd which are hard to define and measure quantitatively. To extrapolate beyond the sub-optimal demonstrations and take the negative effects into our trajectory loss design, we propose to use the Sudden Velocity Changes Rate (SVCR) of each demonstration for trajectory ranking in the T-MEDIRL framework. The SVCR, ϵ_s, is defined as the number of persons, n_s, inside the grid cells whose velocity changes exceed a threshold divided by the length of trajectory l_R, i.e., $\epsilon := \frac{n_s}{l_R}$.

For a crowded environment, using SVCR can help us

TABLE I: Variable Definition.

Variable	Definition	Variable	Definition
$\mathbf{v}_{t,n}$	n-th pedestrian's linear velocity at time step t, $\mathbf{v}_{t,n} \in \mathbb{R}^2$	$\omega_{t,n}$	n-th pedestrian's angular velocity at time step t, $\omega_{t,n} \in \mathbb{R}^2$
v_{thrd}	Linear velocity threshold for judging velocity sudden change	ω_{thrd}	Angular velocity threshold for judging velocity sudden change
S	The metric space of grid cells in current demonstration	l_R	The length of trajectory in current demonstration
N	Total number of pedestrians	ϵ_s	The velocity sudden change rate
n_s	The number of persons having sudden velocity change within grid cells	β	The gradient factor to control the steepness of the function
$\phi(i,j)$	The feature value of the cell at (i,j)	α	Adjustment factor
ρ_{den}	The crowd density around one person within 0.2m	d_{ij}	The distance between the cell and the nearest person
γ	Discount factor for predicted trajectory feature	d_{social}	The social distance of the nearest person
n_t	The order of cells that the trajectory extended to	d_{ij}	The distance between the cell and the nearest person
$\mathbf{x}_{t,n}$	n-th pedestrian's position at time step t, $x_{t,n} \in \mathbb{R}^2$		
N_t	the number of trajectory segments		

Algorithm 1 Sudden Velocity Change Rate

Input: $S, l_R, \mathbf{x}_{t,n}, v_{\text{thrd}}, \omega_{\text{thrd}}, \mathbf{v}_{t,n}, \omega_{t,n}, \mathbf{v}_{(t-1),n}, \omega_{(t-1),n}$
Output: ϵ_s

$n_s \leftarrow 0$ ▷ Initialization
for n_t **do** = 1 to N_t
  for n = 1 to N **do**
    $\Delta\mathbf{v}_{t,n} \leftarrow \mathbf{v}_{(t-1),n} - \mathbf{v}_{t,n}$
    $\Delta\omega_{t,n} \leftarrow \omega_{(t-1),n} - \omega_{t,n}$ ▷ Calculate linear and angular velocity change
    if $\mathbf{x}_{t,n}$ in S **then** ▷ Whether inside grid cells
      if $||\Delta\mathbf{v}_{t,n}|| \geq v_{\text{thrd}}$ **or** $||\Delta\omega_{t,n}|| \geq \omega_{\text{thrd}}$ **then** ▷ Whether velocity changes exceed threshold
        $n_s \leftarrow n_s + 1$
      end if
    end if
  end for
end for
return $\epsilon_s \leftarrow \frac{n_s}{l_R}$ ▷ Normalization

rank robot trajectories that are disruptive to the humans in the scene and avoid reinforcing that behavior to our IRL agent. The algorithm for calculating SVCR is shown in Algorithm 1, and Table I shows the definitions of the variables.

C. Features Extraction

LiDAR and RGB-D cameras can detect obstacle information in a dynamic environment. We use this information to construct four distinct feature layers: distance to the goal, location of unknown static obstacles, predicted trajectory of people, and social distance feature layer. Besides the first two features, which can help the robot navigate to the goal without colliding with unknown obstacles, we use the other two social-awareness features to make the robot consider social dynamics.

1) Distance to Goal feature: The distance to the goal feature is calculated between each cell and the nearest waypoint. After normalization, this feature can be used to propose the direction of navigation for the robot.

2) Unknown Obstacle feature: From the LiDAR information, we can extract the unknown obstacle feature by using Breshenham's Algorithm. This feature can help the robot avoid static obstacles other than people.

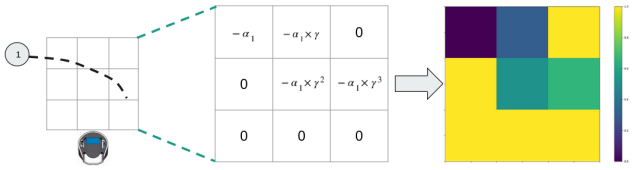

Fig. 3: The trajectory prediction feature layer. The grey circle on the left figure is one pedestrian about to pass through the grid cells in front of the robot. The dashed line is the trajectory predicted from social LSTM. The middle figure contains the raw feature values calculated according to the predicted trajectory. The figure on the right is the heatmap of the normalized feature layer.

3) Predicted Trajectory feature: Since humans are unlikely to stand still in a public environment, adding their walking intention and making the robot aware of the walking direction are crucial to social navigation. Previous IRL-based method only uses the velocity and walking direction of the nearby people in the feature map as a criterion for judging their walking intention [7]. However, this feature is based on the constraint that people walk at a constant speed and fixed direction. To relax the constraints and fully use the past trajectories and social context, we propose using Social-LSTM [28] to predict these trajectories and encode the trajectory information into one of our feature layers.

The spatio-temporal feature layer value is calculated according to the predicted trajectories from the Social-LSTM algorithm. As shown in Fig. 3, instead of setting the binary value to each cell, we add a discount factor according to the order of cells to which the trajectories extended. We calculate the prediction feature of each cell using

$$\phi_{ij} = -\sum_{n=1}^{N} \alpha_{nij} \cdot \gamma^t, \tag{6}$$

$$\alpha_{nij} = \begin{cases} 1 & \text{if the trajectory of } n\text{-th pedestrian in cell}(i,j) \\ 0 & \text{otherwise} \end{cases}. \tag{7}$$

See TABLE I for the variable definitions.

4) Social Distance feature: Social distance is the minimum physical distance between two people who feel comfortable in a social context [32]. Keeping a good social distance between robots and people is vital for socially aware navigation. However, most RL-based methods [33] consider the social distance as fixed, which can cause the freezing-robot problem [34]. Inspired by [10], we encode a resilient social distance feature layer into our feature map for training. Since people are more likely to be comfortable with others coming closer in a high-density environment than in a low-density environment, instead of treating social distance as a fixed value, we treat the social distance of a person as a function of the surrounding crowd density as

$$d_{\text{social}} = \frac{1.577}{(\rho_{\text{den}} - 0.8824)^{0.215}} - 0.967. \tag{8}$$

The work of [10] derives this equation by doing a curve

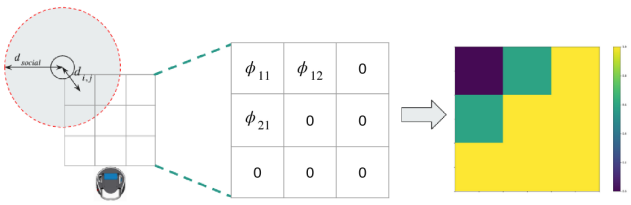

Fig. 4: The social distance feature layer. The big gray circle with a red dashed line boundary is the socially comfortable area of the person. The middle grid cells represent the calculated feature layer. The figure on the right is the heatmap of the normalized feature layer. $\phi_{ij} = \alpha \times \frac{d_{i,j}^{\beta} - d_{\text{social}}^{\beta}}{d_{\text{social}}^{\beta}}$.

fitting over the Asia and Pacific Trade Center (ATC) dataset [35]. As shown in Fig. 4, we add a social distance feature according to the distance between individuals in the crowd and the robot as well as the social distance of the people. To take the social distance into our reward map, the feature value function can be separated into two cases: inside the social area and outside it. The feature value of each grid cell is calculated below:

$$\phi_{ij} = \begin{cases} \alpha \times \frac{d_{ij}^{\beta} - d_{\text{social}}^{\beta}}{d_{\text{social}}^{\beta}} & d_t \leq d_{\text{social}} \\ 0 & d_t > d_{\text{social}} \end{cases} \tag{9}$$

IV. Results and Discussion

Our evaluation can be divided into socially aware feature evaluation and trajectory ranking loss evaluation. Sec. IV-A introduces our simulation environment and training dataset. Sec. IV-B evaluates our socially aware feature map by comparing our method with other socially aware navigation methods. In Sec. IV-C, we evaluate the trajectory ranking loss by comparing our method (SoLo T-DIRL) with the original MEDIRL-based method (SoLo DIRL).

A. Training and Simulation Setup

For the training of SoLo T-DIRL, we collect our data in a Gazebo environment with several unknown obstacles and random people walking around. The moving pedestrian simulator is the open-source PedSim simulator [36], which uses the social-force model to imitate pedestrians. The pedestrians in that simulator have virtual sensors and can avoid dynamic obstacles using the social-force model.

We randomly choose several goals and generate more than 100 demonstrations containing pedestrians with different velocities ranging from slow to fast. The dataset contains 60% optimal demonstrations (no collision and invasion into personal areas) and 40% sub-optimal demonstrations (some invasion into personal areas). We can generate feature maps and their corresponding trajectories according to the demonstration data and different sensor information from LiDAR and an RGB-D camera. Our feature map size for training is $3\,\text{m} \times 3\,\text{m}$. When the robot moves through different grid cells, we collect the number of sudden velocity changes of each demonstration for calculating trajectory ranking loss in (5). The trajectory of each demonstration is also collected for calculating SVCR combined with the number of sudden velocity changes. The size of grid cells is 3×3, whose resolution is $1\,\text{m}/\text{cell}$, and there is no overlapping between two different grid cells.

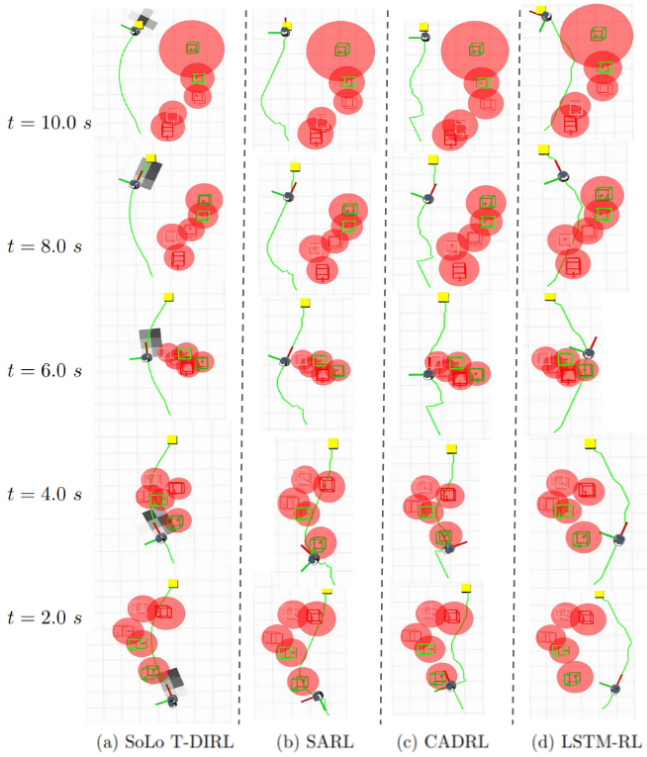

Fig. 5: The comparison of SoLo T-DIRL with other baseline models in a circle crossing simulation environment. From the first row to the last row: SoLo T-DIRL (Ours), SARL[37], CADRL[19], LSTM-RL[38]. The yellow block is the goal position, and the green line is the robot's trajectory. The red area of each person represents the personal area. Each column represents the navigation process using the corresponding method. The navigation process is from bottom to up.

B. Socially-aware Features Evaluation

To evaluate the socially aware factors, we let the robot and people do a circle crossing in an open area, where all the humans and robot are randomly positioned on a circle of radius $4\,\text{m}$, and their goal positions are on the opposite side of the same circle. In this scenario, the robot learns to avoid taking risks to pass through the crowd and learns to keep a comfortable social distance.

We compare our model with several other state-of-the-art socially aware navigation methods widely used in the literature: SARL, CADRL, and LSTM-RL. As shown in Fig. 5, the navigation path of different methods is compared where the pedestrian paths are similar for an exact comparison. Among these methods, our navigation trajectory generated by our SoLo T-DIRL method is the smoothest and the safest. Because we use the trajectory prediction as one of our feature layers shown in Fig. 3, the robot will choose to pass the crowd on the opposite side of the walking direction of people. Also, since our social distance is also taken into our feature map, our robot will tend to keep a proper social distance from people. However, when encountering people in the center

space, CADRL passes through them aggressively, which can cause an invasion of social areas of pedestrians and even collision. LSTM-RL behaves in a risky manner by walking through from the right side of the scene, which may cause a collision by interfering in people's way. When encountering people in the front, SARL changes its orientation suddenly to avoid the crowd, which causes a delay in navigation time.

In TABLE II, we compare our method with SARL, CADRL, and LSTM-RL in four ways. As expected, the table shows that our method exceeds the other three methods. Because our robot can predict people's intentions and plan a safe path, it will take fewer risks, saving time and increasing the success rate. Also, our method outperforms the baselines for invasion rate by adding a flexible social distance feature to our feature map. However, SARL and CADRL use the fixed social distance ($0.2\ \mathrm{m}$) in their reward function, which can cause the robot to invade people's social area when the social distance is more than $0.2\ \mathrm{m}$. LSTM-RL will slow down when the robot approaches people, increasing the navigation time.

C. Trajectory Ranking Loss Evaluation

Besides the feature map, we also evaluate the effect of our trajectory ranking loss in reducing the influence of suboptimal demonstrations. For comparison, we train our network without adding trajectory ranking loss as the baseline. In the classical IRL problem, the model's performance can be evaluated by comparing the reward with the ground truth. However, knowing the ground truth reward in a real-world problem is difficult. Inspired by [12], we use two different metrics for evaluation.

The first is the sudden velocity change rate. To evaluate whether SoLo T-DIRL can improve robot behavior in the social environment, we calculate SVCR in the test dataset. The test dataset is collected in different scenarios with a different number of people walking around. Besides maximizing the reward, SoLo T-DIRL is expected to minimize SVCR, leading to less disturbance during navigation.

The second is the accuracy of classification. The basic idea of using trajectory ranking loss is to choose better demonstrations using a criterion other than the reward function. The demonstrations with a higher trajectory loss should output a lower discounted cumulative reward. Randomly picking up two different demonstrations, if the discounted cumulative reward of the demonstration with higher trajectory ranking loss is less than the other, we say this pair is considered "correct". The accuracy for classification is defined as the number of correct pairs over the number of total pairs.

From TABLE III, our SoLo T-DIRL both does a better job in accuracy and SVCR evaluation, which shows the effect of trajectory ranking loss on reward regulation. It is worth mentioning that SoLo T-DIRL does a better job decreasing SVCR than human teleoperation because people may perform suboptimal demonstrations in the navigation process and cause high SVCR. However, SoLo T-DIRL has put a lower weight on these suboptimal demonstrations, so the final training result is better than human teleoperation.

TABLE II: Quantitative results of socially aware features. "Time" means the average time spent on the navigation process. "Success" means the rate at which the robot reaches the goal without collision. "Invasion" means the number of invasions in the social distance per meter.

Method	Time	Success	Invasion
SoLo T-DIRL (Ours)	**10.41s**	**100%**	**0.0112**
SARL [37]	10.58s	100%	0.0235
CADRL [19]	10.82s	94%	0.1202
LSTM-RL [38]	11.29s	98%	0.0627

TABLE III: Quantitative results for trajectory ranking loss. We compare SoLo T-DIRL with the original SoLo DIRL method and human teleoperation. "-" means there is no meaning to get accuracy from human observation.

Method	Accuracy	SVCR
SoLo T-DIRL	**60.6%**	**0.2538**
SoLo DIRL	38.8%	0.2968
Human	-	0.2672

D. Limitations and Future Work

This work relies on handcrafted features for learning our reward function. Although handcrafted features are explainable, we cannot consider all social dynamics. To model social dynamics, using deep neural networks to extract socially-aware features directly from sensor information [39], [40] is a promising future direction. Furthermore, a hybrid approach where explainable features ensure the necessary inputs and raw sensory data enables lossless training is also an interesting future study. To incorporate the robot dynamics into the planning pipeline, the addition of inertial features to the feature layers [12] or using Model Predictive Control (MPC) [41], [42] with robot dynamics constraints are interesting future research directions.

Finally, a real-world experiment was conducted in a tight indoor environment, as shown in Fig. 1. However, with the lack of real-world robot data as well as the gap between simulation and real life, the performance of the model is lower than the simulation. Therefore, to replicate the success of the simulation experiments on the real robot, more work on the perception algorithms is required. Another possible direction is to develop a real-time world model that takes into account the dynamics and semantics of the scene [43], [44].

V. Conclusion

We developed a socially-aware dynamic local planner based on the T-MEDIRL method to handle sub-optimal demonstrations. In particular, we proposed a novel trajectory ranking loss using the sudden velocity change rate. The ranking strategy provides a built-in value system for the robot to outperform expert demonstrations systematically. Our experiments showed promising results in developing a socially assistive robot that can operate among people in everyday activities. We are also excited about exploring the role of language and visual cues in human-robot interactions within the context of social navigation as future work.

REFERENCES

[1] D. Feil-Seifer and M. Mataric, "Defining socially assistive robotics," in *Proc. Int. Conf. Rehabilitation Robotics*, 2005, pp. 465–468.

[2] N. Gasteiger, M. Hellou, and H. S. Ahn, "Deploying social robots in museum settings: A quasi-systematic review exploring purpose and acceptability," *Int. J. Advanced Robot. Systems*, vol. 18, no. 6, p. 17298814211066740, 2021.

[3] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita, "An affective guide robot in a shopping mall," in *Proc. IEEE Int. Conf. Human Robot Interaction*, 2009, pp. 173–180.

[4] R. Triebel, K. O. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore, H. Hung, O. A. I. Ramírez, M. Joosse, H. Khambhaita, T. P. Kucner, B. Leibe, A. J. Lilienthal, T. Linder, M. Lohse, M. Magnusson, B. Okal, L. Palmieri, U. Rafi, M. van Rooij, and L. Zhang, "Spencer: A socially aware service robot for passenger guidance and help in busy airports," in *Proc. Int. Conf. Field Service Robot.*, 2015.

[5] Y. Gao and C.-M. Huang, "Evaluation of socially-aware robot navigation," *Frontiers in Robotics and AI*, vol. 8, 01 2022.

[6] P. Teja Singamaneni, A. Favier, and R. Alami, "Human-aware navigation planner for diverse human-robot interaction contexts," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2021, pp. 5817–5824.

[7] B. Kim and J. Pineau, "Socially adaptive path planning in human environments using inverse reinforcement learning," *Int. J. Soc. Robot.*., vol. 8, pp. 51–66, 01 2016.

[8] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, pp. 23–33, 1997.

[9] G. Ferrer, A. Garrell, and A. Sanfeliu, "Social-aware robot navigation in urban environments," in *Proc. European Conf. Mobile Robot.*, 2013, pp. 331–336.

[10] X. Lu, H. Woo, A. Faragasso, A. Yamashita, and H. Asama, "Socially aware robot navigation in crowds via deep reinforcement learning with resilient reward functions," *Advanced Robotics*, vol. 36, no. 8, pp. 388–403, 2022.

[11] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *Int. J. Robot. Res.*, vol. 35, no. 11, pp. 1289–1307, 2016.

[12] L. Gan, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning," *IEEE Robotics and Automation Lett.*, vol. 7, no. 4, pp. 8807–8814, 2022.

[13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[14] J. J. Park and B. Kuipers, "A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2011, pp. 4896–4902.

[15] T. Kruse, A. Kirsch, E. A. Sisbot, and R. Alami, "Exploiting human cooperation in human-centered robot navigation," in *Proc. IEEE Int. Conf. Robot and Human Interactive Communication*, 2010, pp. 192–197.

[16] C. Johnson and B. Kuipers, "Socially-aware navigation using topological maps and social norm learning," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 151–157.

[17] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, 05 1998.

[18] X.-T. Truong and T. D. Ngo, "Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 4, pp. 1743–1760, 2017.

[19] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2017, pp. 285–292.

[20] 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.

[21] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, 2008.

[22] R. D. Rosenkrantz, *Where Do We Stand on Maximum Entropy? (1978)*. Dordrecht: Springer Netherlands, 1989, pp. 210–314.

[23] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[24] A. Rudenko, L. Palmieri, M. Herman, K. Kitani, D. Gavrila, and K. Arras, "Human motion trajectory prediction: a survey," *Int. J. Robot. Res.*, vol. 39, p. 027836492091744, 06 2020.

[25] M. Kollmitz, K. Hsiao, J. Gaa, and W. Burgard, "Time dependent planning on a layered social cost map for human-aware robot navigation," in *Proc. European Conf. Mobile Robot.*, 2015, pp. 1–6.

[26] S. Zernetsch, S. Kohnen, M. Goldhammer, K. Doll, and B. Sick, "Trajectory prediction of cyclists using a physical model and an artificial neural network," in *IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 833–838.

[27] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 2255–2264.

[28] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 961–971, 2016.

[29] B. Fu, T. Kathuria, D. Rizzo, M. Castanier, X. J. Yang, M. Ghaffari, and K. Barton, "Simultaneous human-robot matching and routing for multi-robot tour guiding under time uncertainty," *J. Auton. Veh. Sys.*, vol. 1, no. 4, oct 2021.

[30] T. Kathuria, Y. Xu, T. Chakhachiro, X. J. Yang, and M. Ghaffari, "Providers-clients-robots: Framework for spatial-semantic planning for shared understanding in human-robot interaction," in *Proc. IEEE Int. Conf. Robot and Human Interactive Communication*. IEEE, 2022, pp. 1099–1106.

[31] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," *arXiv preprint arXiv:1507.04888*, 2015.

[32] G. A. Akerlof, "Social distance and social decisions," *Econometrica*, vol. 65, no. 5, pp. 1005–1027, 1997.

[33] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2017, pp. 1343–1350.

[34] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2010, pp. 797–803.

[35] D. Brščić, T. Kanda, T. Ikeda, and T. Miyashita, "Person tracking in large public spaces using 3-d range sensors," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 6, pp. 522–534, 2013.

[36] B. Okal and K. O. Arras, "Towards group-level social activity recognition for mobile robots," in *IROS Assistance and Service Robotics in a Human Environments Workshop*, 2014.

[37] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2019, pp. 6015–6022.

[38] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2018, pp. 3052–3059.

[39] A. Staroverov, D. Yudin, I. Belkin, V. Adeshkin, Y. Solomentsev, and A. Panov, "Real-time object navigation with deep neural networks and hierarchical reinforcement learning," *IEEE Access*, vol. 8, pp. 195 608–195 621, 01 2020.

[40] X. Yao, J. Zhang, and J. Oh, "Following social groups: Socially compliant autonomous navigation in dense crowds," *arXiv preprint arXiv:1911.12063*, 2019.

[41] S. Teng, Y. Gong, J. W. Grizzle, and M. Ghaffari, "Toward safety-aware informative motion planning for legged robots," *arXiv preprint arXiv:2103.14252*, 2021.

[42] S. Teng, D. Chen, W. Clark, and M. Ghaffari, "An error-state model predictive control on connected matrix Lie groups for legged robot control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2022, pp. 1–8.

[43] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3D scene graph construction and optimization," in *Proc. Robot.: Sci. Syst. Conf.*, 2022.

[44] J. Wilson, J. Song, Y. Fu, A. Zhang, A. Capodieci, P. Jayakumar, K. Barton, and M. Ghaffari, "MotionSC: Data set and network for real-time semantic mapping in dynamic environments," *IEEE Robotics and Automation Lett.*, vol. 7, no. 3, pp. 8439–8446, 2022.