2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Social Navigation in Crowded Environments with Model Predictive Control and Deep Learning-Based Human Trajectory Prediction

Viet-Anh Le[1,2], Behdad Chalaki[3*], Vaishnav Tadiparthi[3*],
Hossein Nourkhiz Mahjoub[3], Jovin D'sa[3], and Ehsan Moradi-Pari[3]

***Abstract*— Navigating a robot among a crowd has received increasing attention from researchers over the last few decades, resulting in the emergence of numerous approaches aimed at addressing the problem of social navigation to date. Our proposed approach couples agent motion prediction and planning to avoid the freezing robot problem while simultaneously capturing multi-agent social interactions by utilizing a state-of-the-art trajectory prediction model i.e., social long short-term memory model (Social-LSTM). Leveraging the output of Social-LSTM for the prediction of future trajectories of pedestrians at each time-step given the robot's possible future actions, our framework computes the optimal control action using Model Predictive Control (MPC) for the robot to navigate among pedestrians. We demonstrate the effectiveness of our proposed approach in multiple scenarios of simulated social navigation and compare it against several state-of-the-art reinforcement learning-based methods.**

I. Introduction

In the diverse fields of robotics, which include autonomous driving, manipulation, navigation, etc., the most complex situations occur when robots need to operate alongside humans. This complexity primarily stems from the stochasticity in human behavior, which makes it challenging for robots to plan and carry out their tasks efficiently.

In the realm of robot navigation, the problem we tackle in this paper is typically referred to as *social navigation*, also known as *crowd navigation*. It has received a great deal of attention during the last decade [1]–[4] and aims to enable robots to achieve their navigation goals via interacting with their surrounding agents, *i.e.,* humans or other robots, in a way that those surrounding agents do not go through an unpleasant experience during these interactions. This is a complex problem due to the several factors that are contributing to the quality of these inter-agent interactions, among which safety, comfort, legibility, politeness, social competency, agent understanding, pro-activity, and responsiveness to the context have been considered the most important parameters by the social navigation research community [3].

To address some of the mentioned requirements in social navigation, different navigation approaches have been proposed in the literature including 1) Reactive methods; 2) Reinforcement Learning (RL)-based techniques; and 3) Optimization-based methods. Reactive-based methods in general are designed based on considering other agents as moving obstacles and taking into account their reactive behaviors with some assumptions on their collision avoidance strategies [5], [6]. On the other hand, several studies have utilized trajectory-based RL frameworks with deep neural networks. Some of the most important RL-based methods are Collision Avoidance with Deep RL (CADRL) [7], LSTM-RL [8] which is aimed at handling arbitrary numbers of agents, and SARL [9] for obtaining the collective impact of crowd through a self-attention mechanism, and recurrent graph neural network with attention mechanisms [10]. All of these efforts seek to train navigation policies for a single robot that maximizes a specially designed reward function while minimizing the possibility of collisions with other agents. However, the main challenge of RL-based methods is their vulnerability to cases that haven't been encountered during the policy's training phase. Additionally, optimization-based techniques including Model Predictive Control (MPC) are commonly employed in social navigation to optimize the behavior of a robot within a finite control horizon. For instance, Brito *et.al.,* [11] combined RL with an optimization-based method in which a learned policy provides long-term guidance to a local MPC planner. Several other studies have recently proposed MPC with various human prediction models, including constant velocity [12], intention-enhanced optimal reciprocal collision avoidance (iORCA) [13], social generative adversarial networks (GAN) [14], long short term memory (LSTM) [15], and Kalman filters [16]. However, all these studies separate prediction from planning by feeding the prediction to the robot planning module, without considering how human prediction will be affected by different actions of the robot, hence neglecting the potential interaction between the robot and humans.

Employing optimization-based techniques for social navigation is predicated on the assumption that prediction models for human trajectories are available. Recent advances in machine learning-based human trajectory prediction have opened new doors for social navigation problems and several of these have demonstrated the superiority of their prediction techniques over the previous works. Some of the most important ML-based prediction frameworks which have shown their potential in the context of social navigation are Social-LSTM [17], Social-GAN [18], Social-NCE [19], and sparse Gaussian processes [20]. Therefore, due to the superiority of these machine learning prediction models, their combination with MPC framework could potentially improve social navigation performance.

*Both authors contributed equally.
[1]Department of Mechanical Engineering, University of Delaware, DE 19716 USA (Email: vietale@udel.edu). [2]System Engineering, Cornell University, NY 14850 USA. This work was conducted during V.-A. Le's internship at Honda Research Institute.
[3]Honda Research Institute USA, Inc. (Email: {behdad_chalaki; vaishnav_tadiparthi; hossein_nourkhizmahjoub; jovin_dsa; emoradipari }@honda-ri.com).

In this paper, we present our framework for robot navigation in crowded environments in which we integrate a machine-learning based trajectory prediction model *i.e.,* Social-LSTM [17] into an optimization-based planning in an MPC fashion. However, our main distinction with the reported work in the literature is that we couple the prediction and planning to avoid freezing robot problem [21] while capturing multi-agent social interactions among the robot and pedestrians. In our framework, we leverage a Social-LSTM model trained on a real human-trajectory dataset to predict the future behavior of human pedestrians and their interactions with the robot's possible actions. The solution of MPC framework coupled with the Social-LSTM model is the optimal control action for the robot to navigate among the crowd. To numerically solve the MPC problem coupled with the Social-LSTM , we utilize an iterative best-response (IBR) approach [22] inspired by the Nash equilibrium [23]. At each time-step, the method sequentially computes the neural network prediction and solves for the optimal control action of the robot. The performance of the proposed method is evaluated in simulations with different scenarios in comparison with baseline RL techniques to demonstrate the potency and the domain-invariant nature of the MPC approach.

This paper makes two main contributions to the body of literature. Our first contribution involves addressing the freezing robot problem by incorporating the Social-LSTM prediction model into the MPC framework in a recursive fashion, hence enabling a coupled prediction and planning. Second, to the best of our knowledge, in the context of social navigation, this study is the first that considers leveraging the iterative best response approach for solving the planning problem coupled with prediction.

The remainder of the paper is organized as follows. In Section II, we present our problem statement for social navigation, while we provide the specific details of the proposed framework in Section III. We present our simulation results in multiple benchmark scenarios together with analysis in Section IV. Finally, we draw concluding remarks and propose some directions for future research in Section V.

II. Problem Statement

We consider an environment $\mathcal{W} \subset \mathbb{R}^2$ where a single robot navigates among $N \in \mathbb{N}$ human pedestrians, as can be illustrated in Fig. 1. Let 0 be the index of the robot, while $\mathcal{H} = \{1, \dots, N\}$ denotes the set of human pedestrians in the environment.

At time-step $k \in \mathbb{N}$, let $\boldsymbol{s}_{0,k} = [s^x_{0,k}, s^y_{0,k}]^\top \in \mathcal{W}$, $\boldsymbol{v}_{0,k} = [v^x_{0,k}, v^y_{0,k}]^\top \in \mathbb{R}^2$, and $\boldsymbol{a}_{0,k} = [a^x_{0,k}, a^y_{0,k}]^\top \in \mathbb{R}^2$ be the vectors corresponding to position, velocity, and acceleration of the robot in Cartesian coordinates, respectively, where each vector consists of two components for $x-$ and $y-$ axis. Additionally, the robot needs to navigate from an initial position $\boldsymbol{s}_0^{\text{orig}} := [s^x_{0,0}, s^y_{0,0}]^\top$ called origin to a final goal position $\boldsymbol{s}_0^{\text{goal}} \in \mathcal{W}$ while avoiding collisions and any potential discomfort to other pedestrians $i \in \mathcal{H}$. Discomfort is a social conformity metric and is defined to be present if the robot's projected path intersects with a human's predicted path [24]. Let $\boldsymbol{x}^\top_{0,k} = [\boldsymbol{s}^\top_{0,k}, \boldsymbol{v}^\top_{0,k}]$ and $\boldsymbol{u}_{0,k} = \boldsymbol{a}_{0,k}$ be the state vector and control action for the robot at time-step k, respectively. Likewise, let $\boldsymbol{s}_{i,k} = [s^x_{i,k}, s^y_{i,k}]^\top \in \mathcal{W}$ be the position of human $i \in \mathcal{H}$ at time-step k in a vector form.

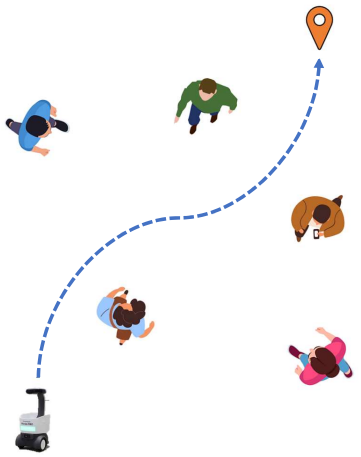


Fig. 1: An example of a robot navigating in a crowded environment.

Assumption 1. *We assume that the real-time position of each pedestrian can be determined through the use of onboard sensors or by obtaining data from a positioning system.*

III. Robot Navigation with Model Predictive Control

Our framework consists of two main components: (1) a Social-LSTM model [17] which learns the social interaction and predicts the future behavior of human pedestrians, and (2) an MPC to find the optimal control action for the robot.

A. Human Motion Prediction using Social-LSTM

Let $t \in \mathbb{N}$ be the current time-step, $H \in \mathbb{Z}^+$ is the control/prediction horizon length (with both equal to each other), and $\mathcal{I}_t = \{t, t+1, \dots, t+H-1\}$ be the set of time-steps in the control horizon. The human prediction model aims at predicting the trajectories of human pedestrians over a prediction horizon of length H given the current and past observations over $L \in \mathbb{Z}^+$ previous time-steps of all agents' trajectories including the robot's. Social-LSTM was developed in [17] for jointly predicting multi-step trajectory of multiple agents. It uses a separate LSTM network for each trajectory, then the LSTMs are connected to each other through a social pooling (S-pooling) layer.

We consider recursive prediction for the pedestrians' positions over the next control horizon using the single-step Social-LSTM model denoted by $\phi(\cdot) : \mathbb{R}^{2(N+1)(L)} \to \mathbb{R}^{2N}$ as follows: [25]

$$\boldsymbol{s}_{1:N,k+1} = \phi(\boldsymbol{s}_{0:N,k-L+1:k}), \forall k \in \mathcal{I}_t. \tag{1}$$

In (1), at each time-step, predicted positions of pedestrians computed from the previous time-steps are used recursively as the inputs of the Social-LSTM model. Furthermore, the Social-LSTM-based predicted positions of the robot are

disregarded as they are computed using the solution of the MPC problem. For further details on the architecture design and implementation of Social-LSTM, the readers are referred to [17].

Remark 1. *It should be noted that while in this work we employ the Social-LSTM model [17] as a human prediction model, our framework can be integrated with alternative deep learning models such as [18], [19], [26].*

B. Model Predictive Control for Social Navigation

In this section, we formulate an MPC problem to navigate the robots while taking into account the trajectory prediction model of surrounding pedestrians. For ease of notation, henceforth, we use $\boldsymbol{u}_0$, $\boldsymbol{x}_0$, and $\boldsymbol{s}_i$, $\forall i \in \mathcal{H}$ instead of $\boldsymbol{u}_{0,t:t+H-1}$, $\boldsymbol{x}_{0,t+1:t+H}$ and $\boldsymbol{s}_{i,t+1:t+H}$, respectively, to denote the vectors concatenating the variables over the control horizon.

The system dynamics of the robot for all $k \in \mathcal{I}_t$ is given by the following discrete-time double-integrator model

$$\begin{aligned} \boldsymbol{s}_{0,k+1} &= \boldsymbol{s}_{0,k} + \tau \boldsymbol{v}_{0,k} + \frac{1}{2}\tau^2 \boldsymbol{a}_{0,k}, \\ \boldsymbol{v}_{0,k+1} &= \boldsymbol{v}_{0,k} + \tau \boldsymbol{a}_{0,k}, \end{aligned} \tag{2}$$

where $\tau \in \mathbb{R}^+$ is the sampling time period.

The speed and control input of the robot at each time-step k are bounded by:

$$\begin{aligned} -v_{\max} \leq v^x_{0,k}, v^y_{0,k} \leq v_{\max}, \\ -a_{\max} \leq a^x_{0,k}, a^y_{0,k} \leq a_{\max}, \end{aligned} \tag{3}$$

where $v_{\max} \in \mathbb{R}^+$ and $a_{\max} \in \mathbb{R}^+$ are the maximum velocity and maximum acceleration, respectively. We formulate the total objective function in MPC by a weighted sum of multiple distinct objectives, representing a diverse set of performance criteria for the robot. In particular, to navigate the robot to the goal, we include tracking minimization to the desired trajectory

$$J^{\text{goal}}(\boldsymbol{s}_0) = \sum_{k=t}^{t+H-1} (\boldsymbol{s}_{0,k+1} - \boldsymbol{s}^{\text{ref}}_{0,k+1})^\top (\boldsymbol{s}_{0,k+1} - \boldsymbol{s}^{\text{ref}}_{0,k+1}), \tag{4}$$

where $\boldsymbol{s}^{\text{ref}}_{0,k+1}$ is the desired position at time $k+1$. We compute the desired trajectory based on the straight line to the robot's goal as follows

$$\boldsymbol{s}^{\text{ref}}_{0,k+1} = \boldsymbol{s}^{\text{ref}}_{0,k} + \min\left\{\tau v_{\max}, \left\|\boldsymbol{s}^{\text{goal}}_0 - \boldsymbol{s}^{\text{ref}}_{0,k}\right\|\right\} \frac{\boldsymbol{s}^{\text{goal}}_0 - \boldsymbol{s}^{\text{ref}}_{0,t}}{\left\|\boldsymbol{s}^{\text{goal}}_0 - \boldsymbol{s}^{\text{ref}}_{0,t}\right\|}, \tag{5}$$

for $k \in \mathcal{I}_t$ and $\boldsymbol{s}^{\text{ref}}_{0,t} = \boldsymbol{s}_{0,t}$.

In addition, we minimize the acceleration and jerk rates of the robot's motion by the following objectives

$$J^{\text{acce}}(\boldsymbol{u}_0) = \sum_{k=t}^{t+H-1} \boldsymbol{u}^\top_{0,k} \boldsymbol{u}_{0,k}, \tag{6}$$

and

$$J^{\text{jerk}}(\boldsymbol{u}_0) = \sum_{k=t}^{t+H-1} (\boldsymbol{u}_{0,k} - \boldsymbol{u}_{0,k-1})^\top (\boldsymbol{u}_{0,k} - \boldsymbol{u}_{0,k-1}). \tag{7}$$

To encourage safety between the robot and the pedestrians, we impose the following constraint that the distance between the robot and each pedestrian $i \in \mathcal{H}$ be greater than a safe speed-dependent distance

$$\|\boldsymbol{s}_{0,k+1} - \boldsymbol{s}_{i,k+1}\|_2^2 \geq d^2_{\min} + \rho \|\boldsymbol{v}_{0,k+1}\|_2^2, \tag{8}$$

where $d_{\min} \in \mathbb{R}^+$ is the minimum allowed distance and $\rho \in \mathbb{R}^+$ is a scaling factor. The above constraint implies that the robot should keep further distances from the humans while moving at higher speed. We include the collision avoidance constraint as a soft constraint in the objective function by using a smoothed max penalty function as follows

$$J^{\text{coll}}(\boldsymbol{x}_0, \boldsymbol{s}_i) = \sum_{k=t}^{t+H-1} \operatorname{smax}\Big(d^2_{\min} + \rho \|\boldsymbol{v}_{0,k+1}\|_2^2 - \|\boldsymbol{s}_{0,k+1} - \boldsymbol{s}_{i,k+1}\|_2^2\Big), \tag{9}$$

where the smoothed max penalty function is defined as

$$\operatorname{smax}(x) = \frac{1}{\mu} \log\big(\exp(\mu x) + 1\big),$$

with $\mu \in \mathbb{R}^+$ as a parameter that manipulates the smoothness of the penalty function.

The MPC objective function can be given by a weighted sum of those features as follows

$$\begin{aligned} J(\boldsymbol{u}_0, \boldsymbol{x}_0, \boldsymbol{s}_{1:N}) = \omega^{\text{goal}} J^{\text{goal}}(\boldsymbol{x}_0) + \omega^{\text{acce}} J^{\text{acce}}(\boldsymbol{u}_0) \\ + \omega^{\text{jerk}} J^{\text{jerk}}(\boldsymbol{u}_0) + \sum_{i \in \mathcal{H}} \omega^{\text{coll}} J^{\text{coll}}(\boldsymbol{x}_0, \boldsymbol{s}_i), \end{aligned} \tag{10}$$

where $\omega^{\text{goal}}, \omega^{\text{acce}}, \omega^{\text{jerk}}$, and $\omega^{\text{coll}} \in \mathbb{R}^+$ are positive weights. Note that the penalty weight ω^{coll} chosen should be sufficiently large. Hence, the MPC formulation for each time-step t is formulated as follows

Problem 1. *At time-step $t \in \mathbb{N}$, robot 0 solves the following MPC problem, the solution of which provides the best control actions for the subsequent H steps, represented as $\boldsymbol{u}_0 = \boldsymbol{u}_{0,t:t+H-1}$. However, at time-step t, the robot alone executes the initial control action $\mathbf{u}_{0,t}$ and disregards the subsequent actions.*

$$\underset{\boldsymbol{u}_0}{\text{minimize}}\ J(\boldsymbol{u}_0, \boldsymbol{x}_0, \boldsymbol{s}_{1:N}), \tag{11a}$$

$$\text{subject to: (1), (2), and (3)}, \forall k \in \mathcal{I}_t, \tag{11b}$$

$$\text{given:}\quad \boldsymbol{s}_{0:N,t-L+1:t}. \tag{11c}$$

C. Iterative Best-Response Implementation

In order to solve the MPC problem (11) coupled with the Social-LSTM model, it is possible to employ gradient-based techniques that require the computation of gradients through back-propagating the LSTM's gradients [25]. Alternatively, we use an iterative best-response technique [22], [27] inspired by the Nash equilibrium concept. This approach involves successively computing the neural network prediction and solving the MPC problem at each time-step for several iterations or until convergence is achieved. We use superscript $j \in \mathbb{N}$ in $\boldsymbol{u}^{(j)}_0$ and $\boldsymbol{x}^{(j)}_0$ to represent the outcomes at the j'th iteration. If the algorithm converges,

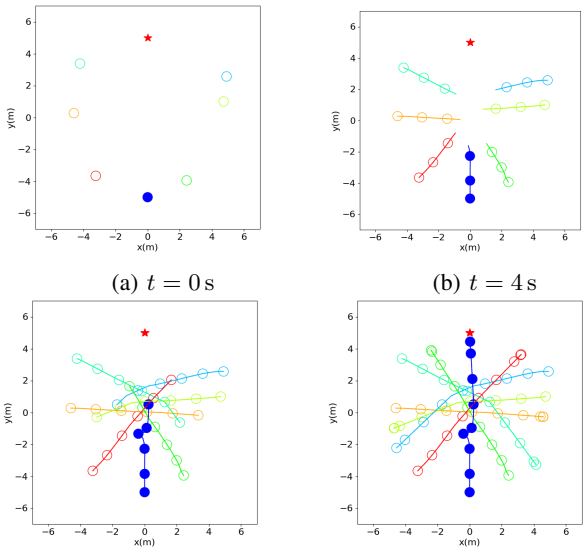
the resultant state is considered a Nash equilibrium [22], [27]. The iterative best-response algorithm for solving MPC problem with the recursive prediction model is detailed in Algorithm 1. At $t = 0$, we initialize $\boldsymbol{u}_0^{(0)} = \mathbf{0}$, and at every time-step $t > 1$, the optimization is warm-started with the solution of the previous time-step.

Algorithm 1 Iterative Best-Response MPC Implementation

Require: t, H, $j_{\max} \in \mathbb{N}$, $\epsilon \in \mathbb{R}^+$, $\boldsymbol{u}_0^{(0)} := \boldsymbol{u}_{0,t:t+H-1}^{(0)}$, $\boldsymbol{s}_0^{(0)} := \boldsymbol{s}_{0,t+1:t+H}^{(0)}$, $\boldsymbol{s}_{1:N,t-L+1:t}^{(0)}$

1: **for** $j = 1, 2, \ldots, j_{\max}$ **do**
2: Predict $\boldsymbol{s}_{1:N}^{(j)} := \boldsymbol{s}_{1:N,t+1:t+H}^{(j)}$ recursively by (1) given $\boldsymbol{s}_0^{(j-1)}$.
3: Solve (11) given $\boldsymbol{s}_{1:N}^{(j)}$ to obtain $\boldsymbol{u}_0^{(j)}$ and $\boldsymbol{x}_0^{(j)}$.
4: **if** $\left\|\boldsymbol{u}_0^{(j)} - \boldsymbol{u}_0^{(j-1)}\right\| \leq \epsilon$ **then**
5: **return** $\boldsymbol{u}_0^{(j)}$
6: **end if**
7: **end for**
8: **return** $\boldsymbol{u}_0^{(j_{\max})}$

IV. Simulation Results



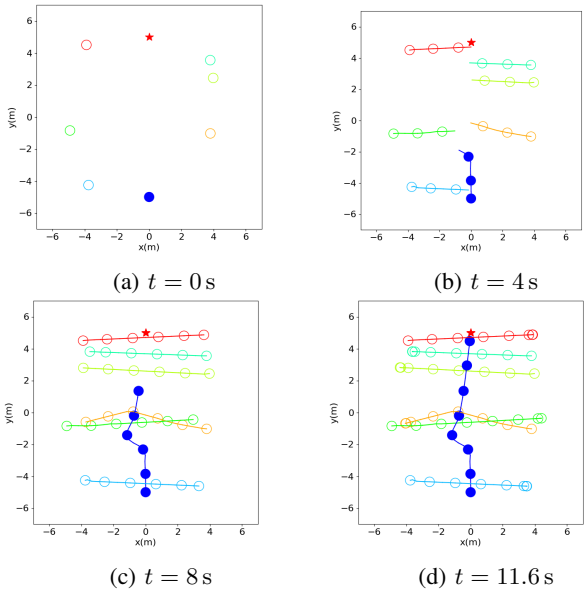
(a) $t = 0\,\mathrm{s}$ (b) $t = 4\,\mathrm{s}$ (c) $t = 8\,\mathrm{s}$ (d) $t = 12.4\,\mathrm{s}$

Fig. 2: Trajectories of the robot (proposed framework) and human pedestrians at several time-steps in a circle-crossing simulation with the robot visible to the humans. The destination of the robot is marked by a red star.

The planning algorithm was implemented in Python in which CasADi [28] and the IPOPT solver [29] are used for formulating and solving the MPC problem, respectively.

We used the following parameters for the MPC problem $\tau = 0.4\,\mathrm{s}$, $H = 8$, $L = 8$, $v_{\max} = 1.0\,\mathrm{m/s}$, $a_{\max} = 2.0\,\mathrm{m/s^2}$, $d_{\min} = 0.8\,\mathrm{m}$, $\rho = 0.5\,\mathrm{s^2}$, $\mu = 30$, $\omega^{\mathrm{goal}} = 10.0$, $\omega^{\mathrm{acce}} = 10^{-1}$, $\omega^{\mathrm{jerk}} = 10^{-1}$, $\omega^{\mathrm{coll}} = 10^{7}$. The simulations were executed on an MSI computer with an Intel Core i9 CPU, 64 GB RAM, and a GeForce RTX 3080 Ti GPU. For social navigation simulations, we used the CrowdNav environment[1] [9] in which the human pedestrians are simulated using Optimal Reciprocal Collision Avoidance (ORCA) [5]. We utilize Trajnet++ benchmark[2] [26] for training Social-LSTM models using synthetic ORCA dataset.

We demonstrate the effectiveness of the proposed method by the trajectories of the robot and human pedestrians in a circle-crossing simulation with 6 human pedestrians in Fig. 2. As shown in the figure, the robot can effectively navigate among the humans and reach the goal in $15.2\,\mathrm{s}$ without colliding with any of them. The robot is visible to the humans during motion to ensure that the interactive behaviors are captured by the prediction module. Similarly, the trajectory of a single robot as it navigates among 6 pedestrians in a square-crossing scenario is depicted in Fig. 3. Notably, the robot is able to reach its destination successfully at $11.6\,\mathrm{s}$ without violating any safety constraints.



(a) $t = 0\,\mathrm{s}$ (b) $t = 4\,\mathrm{s}$ (c) $t = 8\,\mathrm{s}$ (d) $t = 11.6\,\mathrm{s}$

Fig. 3: Trajectories of the robot (proposed framework) and human pedestrians at several time-steps in a square-crossing simulation with the robot visible to the humans. The destination of the robot is marked by a red star.

To further validate the performance of the proposed method in comparison with different navigation algorithms, we collect and compare the following metrics:

- **Success rate**: The percentage of simulations in which the robot successfully reaches its respective destinations without collisions and before scenario timeout.
- **Collision rate**: The percentage of simulations in which the minimum distance between the robot and the pedestrians is below $0.6\,\mathrm{m}$, indicating a violation of personal space.

[1] https://github.com/vita-epfl/CrowdNav
[2] https://github.com/vita-epfl/trajnetplusplusbaselines

TABLE I: Statistical results in Circle and Square-Crossing Scenarios - Success Rates

Method ↓	# Humans	Success Rate (%)							
		5		6		7		8	
	Scenario	C ○	S □	C ○	S □	C ○	S □	C ○	S □
MPC		98.9	**99.6**	98.1	**99.5**	97.9	**98.7**	98.0	**98.8**
CADRL		98.0	85.2	98.2	78.7	96.6	73.8	95.8	68.5
SARL		**99.5**	92.9	**99.6**	90.7	**98.9**	89.9	**99.4**	88.9

TABLE II: Statistical results in Circle and Square-Crossing Scenarios - Collision Rates

Method ↓	# Humans	Collision Rate (%)							
		5		6		7		8	
	Scenario	C ○	S □	C ○	S □	C ○	S □	C ○	S □
MPC		1.1	0.4	1.9	0.5	2.1	1.1	2.0	1.0
CADRL		0.2	0.1	0.2	0.1	0.1	0.0	0.0	0.0
SARL		**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**

TABLE III: Statistical results in Circle and Square-Crossing Scenarios - Discomfort Rates

Method ↓	# Humans	Discomfort Rate (%)							
		5		6		7		8	
	Scenario	C ○	S □	C ○	S □	C ○	S □	C ○	S □
MPC		**0.3**	**0.0**	**0.7**	**1.0**	**0.9**	**0.9**	**0.8**	**1.0**
CADRL		12.2	6.6	12.7	10.5	13.7	12.2	14.6	13.3
SARL		4.2	8.4	7.3	12.6	9.8	16.1	12.3	16.6

TABLE IV: Statistical results in Circle and Square-Crossing Scenarios - Average Travel Time

Method ↓	# Humans	Average Travel Time (s)							
		5		6		7		8	
	Scenario	C ○	S □	C ○	S □	C ○	S □	C ○	S □
MPC		13.0	**11.26**	13.62	**11.51**	14.09	**11.74**	14.67	**12.01**
CADRL		13.22	12.67	13.69	13.15	14.22	13.64	14.71	13.99
SARL		**12.87**	12.98	**13.26**	13.29	**13.68**	13.4	**14.09**	13.66

TABLE V: Statistical results in Circle and Square-Crossing Scenarios - Average Squared Jerk

Method ↓	# Humans	Integrated Jerk (m^2/s^6)							
		5		6		7		8	
	Scenario	C ○	S □	C ○	S □	C ○	S □	C ○	S □
MPC		**1.53**	**0.97**	**1.58**	**1.05**	**1.65**	**1.12**	**1.67**	**1.17**
CADRL		20.93	23.82	21.35	25.47	22.02	26.89	22.59	28.04
SARL		22.92	21.82	24.47	22.78	26.98	23.44	28.21	23.82

- **Discomfort rate**: The percentage of simulations that the robot's projected path intersects with a pedestrian's projected path [24]. The projected path is defined as a line segment from the current position along with the direction of the velocity and the length proportional to the speed. We consider overlap over the next 1.2 s.
- **Average travel time**: The time in seconds it takes for the robot to reach its destination averaged over all successful simulations.
- **Integrated Jerk**: The average squared jerk in m^2/s^6 integrated over the robot's trajectory. It is computed as $\frac{1}{T}\int_{t=0}^{T} \dddot{x}(t)^2 \; dt$, where the jerk $\dddot{x} = \frac{d^3x(t)}{dt^3}$ is computed in discrete time and T is the duration of a successful trajectory in seconds [24].

Among the above metrics, the success rate and average time to the destination describe the path quality of the navigation algorithms. On the other hand, the collision, discomfort rates, and the average jerk are related to social conformity and naturalness of the robot's motion [24]. If the simulation reports neither a success nor a collision, it means a timeout has occurred wherein the robot has not been able to traverse to its destination within 30 seconds.

We compare our proposed MPC with two notable RL algorithms including CADRL [7] and SARL [9]. This evaluation was based on 8000 simulations with varying numbers of human agents and randomized initial conditions. We trained both RL algorithms using the circle-crossing scenario in the CrowdNav environment with a similar implementation

and parameters of [9] including the imitation learning step. However, to thoroughly evaluate their generalization capability, we conducted tests in both circle-crossing and square-crossing scenarios, and summarized our results in Tables I-V.

Table I depicts the success rate of a robot navigating in both circle and square-crossing scenarios among pedestrians, crowd sizes ranging from 5 to 8. We conducted 1000 simulations for each combination of scenarios and number of human pedestrians, using randomized initial conditions generated from random seeds that were different from the training seeds for CADRL and SARL. As can be seen, our proposed approach consistently yielded a greater success rate in comparison to the RL-based strategies across all scenarios. While the performance of CADRL and SARL is comparable to our proposed approach in the circle-crossing scenario, their success rate drops over 10%-30% in the square-crossing scenario which crossing pattern is different from their training environment. The decrease in success rate is due to a significant timeout rate, indicating the incapability of the policies to generate a feasible path. This clearly illustrates the significant reliance of RL techniques on the domains in which they have been trained.

Tables II and III summarize the collision and discomfort rates for our proposed approach and RL-based navigation methods. Based on Table II, it can be observed that MPC results in collision rates of less than 2% across all scenarios. For RL-based navigation methods, the collision rate is less than 0.6%, however, they exhibit a higher discomfort rate than the MPC approach as shown in Table III. In particular, the CADRL policy encounters growing discomfort rates with increasing crowd densities. MPC on the other hand, shows a lower discomfort rate, which implies MPC approach can be more socially conscious of the human pedestrians' motion. Additionally, the lower collision rate observed in RL-based policies could be attributed to the simplicity of their robot dynamics and directly controlling the speed of the robot, potentially neglecting dynamic constraints such as allowed acceleration. Namely, RL-based policies exhibit higher occurrences of timeouts, indicating that to avoid collisions, the robot may stay at its position and not move towards the goal even after humans have reached their destinations.

In Table IV, we provide results of the average travel time for the MPC approach and RL-based methods. Overall, the performance of MPC and RL algorithms in circle-crossing simulations is highly comparable. However, we observe higher travel times for the RL techniques in square-crossing simulations. Unlike the RL algorithms, the control policy in the proposed MPC formulation does not need any pre-training. Furthermore, RL algorithms are highly susceptible to domain shift. As mentioned earlier, for this set of evaluations, RL-based methods have been trained in the circle-crossing scenario but tested in both the circle-crossing and a square-crossing scenario.

In Table V, we can observe the results of average jerk for our evaluated approaches. In all scenarios, the jerk observed for the MPC technique is significantly lower than the RL baselines. Avoiding jerky movements can enhance human understanding of robots' actions and make them more acceptable [30]. Since the motion of the robot following the proposed policy is inherently less jerky, we can infer that this leads to a more natural and sociable navigation among human pedestrians.

Next, we investigated the sensitivities of the metrics to the different parameters used in the objective function for the proposed approach. We ran four sets of 100 simulations with varying crowd sizes of 5 and 10 pedestrians each in a circle-crossing scenario. The parameter w^{coll} representing the weight on collision avoidance, denoted as w^{coll}, was held constant at a value of 10^7. When w^{goal} is set to a low value of 0.1, we observe a significant increase in average travel time on the introduction of penalties on acceleration and jerk. However, this effect is nullified on increasing w^{goal} to 10. This observation is consistent across different crowd sizes. Moreover, as the crowd density increases, there is a marked increase in the travel time and a slight decrease in the success rates. The difficulty induced due to the multi-objective nature of the optimization is apparent in the way these costs compete against one another in high crowd densities.

TABLE VI: The effects of parameters on social navigation performance in a circle-crossing scenario over 100 simulations

# Humans	w^{acce} w^{jerk}	w^{goal}	Suc. (%)	Coll. (%)	Disc. (%)	Avg. Time (s)
5	0	0.1	99	1	1	13.2
5	0.1	0.1	99	1	0	17.9
5	0	10	100	0	2	12.9
5	0.1	10	99	1	0	12.9
10	0	0.1	99	1	1	16.5
10	0.1	0.1	95	2	1	18.7
10	0	10	96	4	1	15.9
10	0.1	10	97	3	1	15.5

To assess the effects of horizon length on the performance, we conducted another set of experiments for circle and square-crossing scenarios. For each combination of scenario and control horizon length, we performed 100 simulations with a crowd size of 6 and present the findings in Table VII. Consistent with our expectations, our finding indicates that a longer control horizon has a direct positive impact on the success rate. However, it is evident that there exists a trade-off between the extension of the control horizon and the cost of computational processes. In this case, 8 turns out to be a suitable choice for the control horizon. We also observe a slight increase in the average number of iterations until convergence with an increase in the control horizon.

V. Conclusions

This work presented a control framework for navigating an individual robot in crowded environments. Our control framework is a combination of MPC and a human trajectory prediction model based on Social-LSTM. In order to evaluate the performance of our proposed approach, we

TABLE VII: Computation Time Increase with Control Horizon

Scenario	H	Iter	Comp. Time (s)	Suc.	Coll.	Disc.
C ○	4	2.4	0.1	95	5	4
S □	4	2.1	0.1	99	0	2
C ○	8	2.8	0.26	98	2	2
S □	8	2.4	0.22	99	1	1
C ○	12	3.1	0.56	99	1	0
S □	12	2.5	0.47	100	0	3

conducted extensive simulations and compared our approach against several state-of-the-art RL algorithms. We showed that the performance of our proposed approach in benchmark scenarios is highly comparable, and in contrast to the RL algorithms, our control policy is not liable to suffer from a distribution shift. A possible extension is to investigate the effectiveness of the proposed control framework in dealing with multiple robots navigating in a coordinated manner through crowds of humans. Additionally, to bridge the gap between simulation and reality, another future research direction is to investigate the variety of human behaviors and to incorporate more measures of sociability into the planning framework.

REFERENCES

[1] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[2] K. Charalampous, I. Kostavelis, and A. Gasteratos, "Recent trends in social aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 93, pp. 85–104, 2017.

[3] A. Francis, C. Pérez-D'Arpino, C. Li, F. Xia, A. Alahi, R. Alami, A. Bera, A. Biswas, J. Biswas, R. Chandra, H.-T. L. Chiang, M. Everett, S. Ha, J. Hart, J. P. How, H. Karnan, T.-W. E. Lee, L. J. Manso, R. Mirksy, S. Pirk, P. T. Singamaneni, P. Stone, A. V. Taylor, P. Trautman, N. Tsoi, M. Vázquez, X. Xiao, P. Xu, N. Yokoyama, A. Toshev, and R. Martín-Martín, "Principles and guidelines for evaluating social robot navigation algorithms," 2023.

[4] C. Mavrogiannis, F. Baldini, A. Wang, D. Zhao, P. Trautman, A. Steinfeld, and J. Oh, "Core challenges of social robot navigation: A survey," *ACM Transactions on Human-Robot Interaction*, vol. 12, no. 3, pp. 1–39, 2023.

[5] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE international conference on robotics and automation*. Ieee, 2008, pp. 1928–1935.

[6] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[7] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1343–1350.

[8] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3052–3059.

[9] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6015–6022.

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

[11] B. Brito, M. Everett, J. P. How, and J. Alonso-Mora, "Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4616–4623, 2021.

[12] T. Akhtyamov, A. Kashirin, A. Postnikov, and G. Ferrer, "Social robot navigation through constrained optimization: a comparative study of uncertainty-based objectives and constraints," *arXiv preprint arXiv:2305.02859*, 2023.

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

[14] S. Poddar, C. Mavrogiannis, and S. S. Srinivasa, "From crowd motion prediction to robot navigation in crowds," *arXiv preprint arXiv:2303.01424*, 2023.

[15] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[16] V. Vulcano, S. G. Tarantos, P. Ferrari, and G. Oriolo, "Safe robot navigation in a crowd combining nmpc and control barrier functions," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 3321–3328.

[17] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[18] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2255–2264.

[19] Y. Liu, Q. Yan, and A. Alahi, "Social nce: Contrastive learning of socially-aware motion representations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 118–15 129.

[20] P. Trautman, "Sparse interacting gaussian processes: Efficiency and optimality theorems of autonomous crowd navigation," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 327–334.

[21] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 335–356, 2015.

[22] J. L. V. Espinoza, A. Liniger, W. Schwarting, D. Rus, and L. Van Gool, "Deep interactive motion prediction and planning: Playing games with motion prediction models," in *Learning for Dynamics and Control Conference*. PMLR, 2022, pp. 1006–1019.

[23] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. SIAM, 1998.

[24] J. Wang, W. P. Chan, P. Carreno-Medrano, A. Cosgun, and E. Croft, "Metrics for evaluating social conformity of crowd navigation algorithms," in *2022 IEEE International Conference on Advanced Robotics and Its Social Impacts (ARSO)*. IEEE, 2022, pp. 1–6.

[25] P. Gupta, D. Isele, D. Lee, and S. Bae, "Interaction-aware trajectory planning for autonomous vehicles with analytic integration of neural networks into model predictive control," *arXiv preprint arXiv:2301.05393*, 2023.

[26] P. Kothari, S. Kreiss, and A. Alahi, "Human trajectory forecasting in crowds: A deep learning perspective," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7386–7400, 2021.

[27] G. Williams, B. Goldfain, P. Drews, J. M. Rehg, and E. A. Theodorou, "Best response model predictive control for agile interactions between autonomous ground vehicles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2403–2410.

[28] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.

[29] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[30] F. Papenmeier, M. Uhrig, and A. Kirsch, "Human understanding of robot motion: the role of velocity and orientation," *International Journal of Social Robotics*, vol. 11, pp. 75–88, 2019.