Pedestrian Behaviour Model for Crowd Simulation, Learning, and Benchmarking of Robot Navigation

Subham Agrawal[1] Nils Dengler[1,2] Maren Bennewitz[1,2,3]

***Abstract*— The presence of robots amongst pedestrians affects them causing deviation to their trajectories. In order to better study and navigate around this issue, we introduce a simulation framework that repetitively measures and benchmarks the deviation in trajectory of pedestrians due to robots driven by different navigation algorithms. In this paper, we enhance the traditional Social Force Model (SFM) by incorporating a novel force component that accounts for the influence of robots on pedestrian behavior, resulting in the Social Robot Force Model (SRFM). This extended model improves the prediction accuracy of pedestrian trajectories disrupted by robots. The pedestrians are then simulated using the SRFM with and without forces affecting them due to the presence of the robot to objectively measure the deviation to their trajectory caused by the robot in 2 different scenarios.**

I. Introduction

The increasing presence of robots in everyday life has made the issue of social navigation of mobile robots among pedestrians more relevant than ever.

Research in the field of social navigation primarily revolves around three approaches: traditional model-based methods [1], data-driven learning approaches [1], and hybrid approaches that combine elements of both [2], [3]. Traditional model-based methods use physical models, such as force-based [1] or fluid motion [1], under specific assumptions to predict pedestrian movement and, in turn, guiding the robot. Recently, with advances in computing power, data-driven methods using machine learning have gained popularity. These methods involve collecting datasets of pedestrian and robot interactions to train models that predict pedestrian behavior and guide the robot accordingly.

Social robot navigation focuses on two main parts: the social aspect and the robot navigation. While robot navigation has been extensively studied and solved using both traditional and data-driven approaches, a repetitive measure for the affect of robots on pedestrians remains a dynamic area of research [1], [4]. Since pedestrian behavior in the context of social norms is not a simple modelling problem due to its complex nature, learning-based approaches have become the preferred approach for approximation. Typically, this involves collecting data from a teleoperated robot navigating through crowds in a socially compliant manner.

Recently, Hirose *et al.* [4] introduced a metric for measuring effect of robots on pedestrians by evaluating the counterfactual perturbation of human trajectories caused by the presence of a robot in the scene. However, the dynamic application of this metric remains a challenge, especially when relying on pre-recorded datasets, which limits the ability to test policies in scenarios different from those in the dataset.

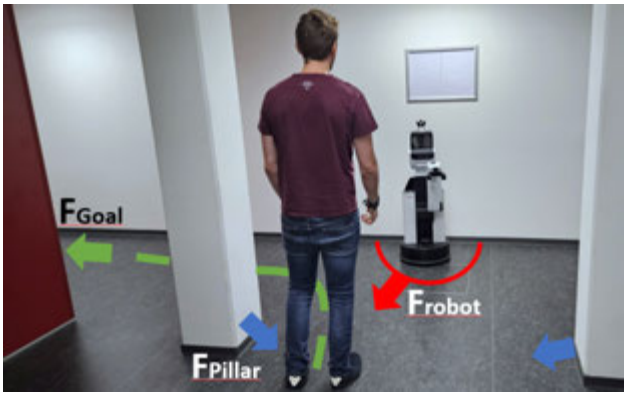


Fig. 1: Example scenario of a human navigating to a goal. While many of the forces used in the social force model are well studied, such as attraction to the goal or repulsion from obstacles, such as the pillar, the robot force is not well understood, even though it significantly affects human behavior.

To address these issues, we present a simulation framework designed to repetitively and objectively measure the effect of robots driven by different social navigation algorithms on pedestrian trajectory for evaluation and benchmarking. Furthermore, we introduce a new force-factor to the social force model (SFM) [5], [6] inspired by the work done by Ferrer *et al.*[7], to model the robot's influence to the pedestrian walking behavior without modeling it as a simple obstacle. This social robot force model (SRFM) is used to account for changes in the pedestrians' behavior that can't be modeled adequately by the original forces of the SFM. Our framework simulates pedestrian behavior based on real-world datasets according to our SRFM, and allows for the creation of scenarios not covered in the original dataset. This flexibility enables the development of navigation policies for specific scenarios such as complex hallways as well as generalized scenarios such as pedestrian crossings. To validate this claim, we developed and evaluated a reinforcement learning (RL) policy that uses SRFM to learn a human aware robot navigation behavior by minimizing the influence of robot force on the pedestrian trajectory.

II. Our Approach

In this work we present a novel simulation and benchmark environment for learning, evaluation, and comparing of social navigation policies under various known and new situations.

In this section, we introduce the extended social force

[1]: Humanoid Robots Lab, University of Bonn, Germany
[2]: The Lamarr Institute, Bonn, Germany
[3]: Center for Robotics, Bonn, Germany

model that includes the repulsion effect of navigating robots on pedestrians. We explain in detail our methods for learning the parameters of the social force, defining an objective social metric, setting up a simulation environment for training a navigation policy for mobile robots, and evaluating our learned policy against other state-of-the-art algorithms using this simulation environment.

A. Social Robot Force

The original SFM as defined by Equation 1, uses three major forces - attraction towards the goal f_a, repulsion from other pedestrians in the vicinity f_p, and repulsion from obstacles in the surroundings f_o. Our modified SFM, represented by Equation 2, introduces an additional force component - repulsion from the robot f_r.

$$F = f_a + f_p + f_o \tag{1}$$

$$F = f_a + f_p + f_r \tag{2}$$

For the sake of simplification, in the paper we omit the force of repulsion from obstacles. However, this factor is well studied and can be added to the equation for further research. The different force components are calculated in distinct ways. The attraction force towards the goal f_a is described in Equation 3. τ, representing the time a pedestrian takes to adjust their current velocity v_i to match the desired velocity v_0 (the ideal velocity desired by the pedestrian to move towards the goal) and has to be determined, i.e., learned. The repulsion forces from pedestrians f_p, obstacles f_o, and robots f_r share the same formula as depicted in Equation 4. In this function, the parameters to be learned are A, indicating the strength of the force, and B, representing the distance from which the force starts to have a significant effect. d is the sum of the radii of the interacting pedestrians. x is the actual distance between the interacting pedestrians. Additionally, an anisotropic value, as shown in Equation 5 is used to show that pedestrians experience stronger repulsive forces from those in front of them.

$$f_a = \frac{v_i - v_0}{\tau} \tag{3}$$

$$f_p = Ae^{\frac{d-x}{B}}\psi \tag{4}$$

$$\psi = \lambda + (1-\lambda)\frac{(1+cos(\phi))}{2} \tag{5}$$

This force decreases to the sides and becomes zero behind the pedestrian. This factor ψ is included as an extra multiplicative factor in the force from Equation 4 and helps in more accurately modeling pedestrian behavior. The value of λ is another parameter that needs to be learned from real-world data.

In this work, we learn all parameters based on trajectories of the JRDB dataset [8]. To learn the factors of the SFM independent from the robot force, we manually processed the dataset to categorize its trajectories into interaction and non-interaction types. Non-interaction category involve the trajectories of pedestrians far away from the robot (greater than 3 meters) and visually not affected by it. These trajectories are used to learn the parameters for the pedestrian repulsion component. For the repulsion force from robots, only the parameters A and B need to be learned from the interaction trajectories, which represent those trajectories where pedestrians are near the robot (empirically chosen to be less than 3 meters as this was found to be the minimum distance that effectively distinguishes the two types of trajectories) and its presence causes deviation in their path. These trajectories are used to learn the parameters for the robot repulsion force component while retaining the values for pedestrian repulsion part from the non-interaction trajectories. A non-linear least squared optimization technique from SciPy library [9] is used to learn the parameters in both the scenarios. The resulting parameter values are summarized in table 1.

Paper	A_p	B_p	1	τ	A_r	B_r
Ours	2	0.89	0.4	0.6	7.93	0.99
Ref	2.66	0.79	0.59	0.43	2.66	0.79

TABLE I: Parameter Values for Social Robot Force Model compared to [7]

B. Benchmarking Simulation

A notable feature of our benchmark simulation is the objective metric to measure the deviation in trajectory of the pedestrians caused by the presence of robots driven by social navigation algorithms. This metric is based on the prior research done by Hirose *et al.* [4]. The SRFM consists of various individual force components, making it efficient to evaluate a social navigation policy twice under the same scenario, with one key difference: the robot's impact on the pedestrian. To measure the counterfactual perturbations caused by the robot on the pedestrian's trajectory, we first set up the SRFM with its force components active and run the benchmark, recording the pedestrian trajectories. We then conduct a second run of the deterministic benchmark with the robot force component disabled, i.e., pedestrians driven by the SFM. This provides us with pedestrian trajectories for the same scenario but without the robot's influence. By comparing the differences between these two sets of trajectories and quantifying this difference using Fréchet distance [10], we can quantify and test the deviation caused by the robots driven by a social navigation policy. The general idea behind the metric in our benchmark is that any trajectory alteration due to the presence of robot deviates from the norm for the pedestrian who would have traversed a particular trajectory had the robot not been present.

C. Learning a Navigation Policy

To demonstrate the advantage of SRFM and our simulation framework, we use reinforcement learning (RL) to train a policy that navigates the robot through a crowd of pedestrians. While reaching a certain goal in the environment, the robot is tasked to cause the least disturbance to the pedestrians velocity and trajectory.

In the following we elaborate on the action, observation and reward functions that we used to train and evaluate the RL agent in the context of SRFM and social norms.

1) Action: We use a continuous action space for the agent that consists of linear and angular velocities (v, w), with a range of $v \in [-0.5, 0.5]\ \mathrm{m\,s^{-1}}$ and $w \in [-\pi, \pi]\ \mathrm{rad\,s^{-1}}$. The robot is allowed to move freely through the environment with no restriction to its direction to enable fast evasive movements to make room for the pedestrians if they approach the robot.

2) Observation: To reduce the complexity of the agent's observation space, all components are either relative to the robot's position or a boolean. As policy observation of the environment, we use the distance and angle to the navigation goal (2D) as well as of each pedestrian with in a predefined social zone (4*10D), as described in Sec. II-A. We also provide relative pedestrian velocity for each pedestrian in the observation. We use an upper bound of the ten closest pedestrians, as other research suggests an upper bound of interfering humans within a scene of nine [3]. For scenarios with less than ten pedestrians within the robot's social zone, the observation is padded with zeros. In order to make learning more robust, the last action taken by the robot (v, w), as well as the success (1D) and termination criteria (boolean, 3D), are also included in the observation function. In total, we use an observation space of size 48.

3) Reward: We keep the reward as simple as possible to encourage faster training, since it is an important part for training convergence. Our reward function r_{total} consists of three components,

$$r_{total} = r_{term} + -k_1 \cdot r_{dist} - k_2 \cdot r_{div}, \tag{6}$$

with k_1, k_2 as scaling factors. r_{term} is a large, sparse termination reward, which is positive when the episode is successful, and otherwise negative. Additionally, we are penalizing collisions with walls or pedestrians double as much as reaching the maximum time step limit. r_{dist} is the Euclidean distance to the goal position, normalized to the range $[0, 1]$, To benefit from our SRFM definition, we use r_{div} to penalize the agent for divergence of any pedestrian from its path predicted by the SFRM. We calculate the reference path by setting the robot force $f_r = 0$.

4) Training method: To train the policy, we use the Twin-Delayed Deep Deterministic Policy Gradient (TD3) algorithm implementation of stable baselines 3 [12] with a maximum of $2e6$ training steps and a learning rate of $1e-4$. As TD3 is an off-policy RL algorithm, we set the length of its experience replay buffer to $1e6$. Each episode consists of 750 steps, where start and goal positions of the robot and the pedestrians are assigned randomly while considering a certain minimum distance d=2 between any of them to avoid unlikely or dangerous situations where the robot spawns on top of or very close to a pedestrian or vice versa.

III. Experiments and Results

In this section, we evaluate the influence of the SRFM in terms of prediction accuracy on a given dataset, as well as its usability for robot navigation policies. For evaluation and training of the policy, we use the simulation environment as described in Sec. II-B.

A. SRFM Prediction Accuracy

The trained parameters for the SRFM were tested on a subset of trajectories derived from the JRDB dataset. The testing was done in two phases. First, we tested the accuracy of our learned SFM on pedestrian trajectories that are not disrupted by the robot, which resulted in an Average Displacement Error (ADE) of $0.70m$. Afterwards, we checked the accuracy of our SRFM on pedestrian trajectories that are disrupted by the robot which results in ADE of $0.59m$. Additionally, we tested the SRFM without the robot force to see if it matches the performance of the SFM on disrupted trajectories. Since interactions bring a new dynamic to the trajectories of humans, we wanted to test if this changes the influence of other forces to the model. Without robot force, the learned parameters showed an ADE of $0.75m$ on interaction trajectories, whereas, using the robot force showed great improvement with $ADE = 0.59m$ on the same trajectories. Therefore, it is shown that the robot force of the SRFM has a great impact on the prediction accuracy, while minimizing the influence to other parameters, such as pedestrian repulsion f_p of the model. The results are presented in Table III.

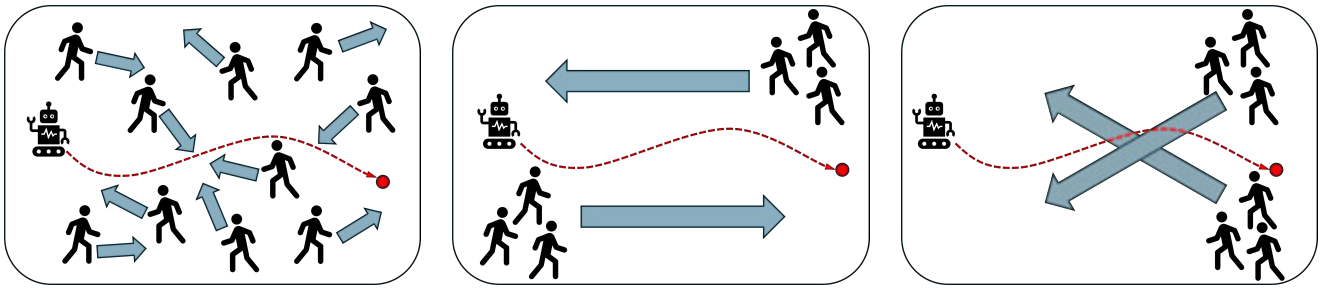
B. Simulation Setup

For Training and Evaluation of the RL agent, we defined different environments as described in the following.

1) Training: As training environment, we define a free space of 15×15 meters within our simulation. The robot's start and goal positions are randomly generated while maintaining a minimum distance of 5 meters. During training, 10 pedestrians are randomly sampled within the environment boundaries while maintaining a minimum distance of $2m$ (the robot's social zone) from the robot's start position. In addition, each pedestrian's goal is randomly sampled, restricting it to be outside the robot's goal's social zone and keeping a minimum distance of 7 meters from that pedestrian's spawn location. In order to provide an active and crowded training environment, once the pedestrian has reached its goal, but the training episode has not ended, it is assigned a new goal as described above. Since the robot does not keep track of pedestrian instances, we artificially generate a higher pedestrian population in this way, as they can interfere with the robot's path multiple times. An example environment for training is shown in Fig. 2a.

2) Evaluation: To evaluate the policy after training, we define two different scenarios as described by Francis *et al.* [11] as parallel traffic and a variant of the circular crossing. While keeping the same environmental boundaries as for training, we adjust the start and end sampling of each pedestrian and the robot.

In **Scenario 1**, we simulate a sidewalk or similar walking area by maintaining the parallel flow of pedestrians bypassing each other. For this scenario, the start and end positions



(a) Training environments (b) Footpath (c) Crosswalk

Fig. 2: Overview of the different used environments in this work. (a) shows the training environment that is used for the RL agent. (b) and (c) visualize the two common pedestrian stream "Footpath" and "cross walk" as described in [11].

	Approach	Fréchet dist. ↓	Traj. Length ↓	Time ↓	Min. Robot Dist. ↑
Scenario 1	Ours	**0.72** $\pm$ 0.05	8.89 $\pm$ 0.04	16.41 $\pm$ 0.34	$*$**1.51** $\pm$ 0.05
	Baseline	0.73 $\pm$ 0.01	**8.71** $\pm$ 0.01	$*$**13.99** $\pm$ 0.08	1.41 $\pm$ 0.01
Scenario 2	Ours	$*$**1.05** $\pm$ 0.01	$*$**14.23** $\pm$ 0.07	$*$**15.88** $\pm$ 0.11	$*$**1.94** $\pm$ 0.01
	Baseline	1.81 $\pm$ 0.06	15.17 $\pm$ 0.13	18.50 $\pm$ 0.48	1.27 $\pm$ 0.04

TABLE II: Performance of our against a baseline approach in terms of the Fréchet distance, trajectory length, the time taken to complete the trajectory as well as the minimum distance from a pedestrian to the robot. All values are averaged over 20 runs with $*$ indicating significance according to the independent t-test with p=0.05.

Model	Force	Test Trajectory	ADE
SFM	$f_a + f_p$	Non-interaction	0.70m
SRFM	$f_a + f_p$	Interaction	0.75m
SRFM	$f_a + f_p + f_r$	Interaction	**0.59m**

TABLE III: Error in trajectory prediction on the JRDB dataset between the social force model and our social robot force model

of the pedestrians are set to create a natural parallel flow, as can be seen in Fig. 2b. The start and end positions of the robot are fixed and shown as a red circle and a green star.

For **Scenario 2** a pedestrian crossing situation is created. The pedestrian start and goal positions are set to form a natural cross-flow, as shown in 2c. The robot start and goal positions are set up to directly disrupt the flow at the pedestrian crossing point, making it more difficult for the robot to minimize its influence on the pedestrian trajectories.

C. RL Agent Performance

As a baseline, we trained another agent in the same way, neglecting the path deviation penalty for pedestrians within the social zone. While this agent still avoids pedestrians, it does not incorporate any knowledge of its influence on their trajectories. The results of the evaluation of both policies in both scenarios are shown in Tab. II.

Scenario 1 shows marginal improvement in Fréchet distance when using our trained RL agent compared to the baseline. However, the baseline outperforms our agent in other objective metrics such as trajectory length of pedestrians and the time taken for the trajectories. This can be explained by the fact that although the baseline policy tries to avoid pedestrian due to the learned collision penalty, it does not care about pedestrian deviation and tries to reach the goal as soon as possible. This leaves behind an empty region for the pedestrians to move freely and reach their goal sooner. Our trained RL agent on the other hand tries to respect pedestrian deviation and in turn causes delays and longer trajectories for the pedestrians in a simple scenario like Scenario 1. The more complex scenario as shown in Scenario 2 presents the capabilities of our learned agent where it outperforms the baseline model in every metric significantly, showing a clear and distinct advantage in its prediction capabilities.

IV. Conclusion

In this paper, we introduced a simulation framework to objectively measure and benchmark the deviation in trajectory of pedestrians due to robots driven by different navigation algorithms. By extending the traditional Social Force Model (SFM) to include robot influence on pedestrian behavior, our Social Robot Force Model (SRFM) offers enhanced prediction accuracy for pedestrian trajectories disrupted by robots. Experiments showed a low Average Displacement Error (ADE) for the prediction accuracy of the SRFM, and our reinforcement learning policy trained with SRFM demonstrated improved results causing less deviation to pedestrian trajectories. We will release the source code and simulation framework for use by the research community to encourage further development, replication, and validation of our findings and evaluation of learned navigation policies.

References

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

[2] G. Zhang, Z. Yu, D. Jin, and Y. Li, "Physics-infused machine learning for crowd simulation," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2439–2449.

[3] S. Hossain, F. T. Johora, J. P. Müller, S. Hartmann, and A. Reinhardt, "Sfmgnet: A physics-based neural network to predict pedestrian trajectories," *arXiv preprint arXiv:2202.02791*, 2022.

[4] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "Sacson: Scalable autonomous control for social navigation," *IEEE Robotics and Automation Letters*, 2023.

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

[6] P. Regier, I. Shareef, and M. Bennewitz, "Improving navigation with the social force model by learning a neural network controller in pedestrian crowds," in *2019 European Conference on Mobile Robots (ECMR)*. IEEE, 2019, pp. 1–6.

[7] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1688–1694.

[8] R. Martin-Martin, M. Patel, H. Rezatofighi, A. Shenoi, J. Gwak, E. Frankel, A. Sadeghian, and S. Savarese, "Jrdb: A dataset and benchmark of egocentric robot visual perception of humans in built environments," *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[9] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[10] J. de Heuvel, N. Corral, B. Kreis, J. Conradi, A. Driemel, and M. Bennewitz, "Learning depth vision-based personalized robot navigation from dynamic demonstrations in virtual reality," 2023.

[11] A. Francis, C. Pérez-d'Arpino, C. Li, F. Xia, A. Alahi, R. Alami, A. Bera, A. Biswas, J. Biswas, R. Chandra, *et al.*, "Principles and guidelines for evaluating social robot navigation algorithms," *arXiv preprint arXiv:2306.16740*, 2023.

[12] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, 2021.