SICNav-Diffusion: Safe and Interactive Crowd Navigation with Diffusion Trajectory Predictions

Sepehr Samavi, Anthony Lem, Fumiaki Sato, Sirui Chen, Qiao Gu, Keijiro Yano,
Angela P. Schoellig, Florian Shkurti

Abstract— **To navigate crowds without collisions, robots must interact with humans by forecasting their future motion and reacting accordingly. While learning-based prediction models have shown success in generating likely human trajectory predictions, integrating these stochastic models into a robot controller presents several challenges. The controller needs to account for interactive coupling between planned robot motion and human predictions while ensuring both predictions and robot actions are safe (i.e. collision-free). To address these challenges, we present a receding horizon crowd navigation method for single-robot multi-human environments. We first propose a diffusion model to generate joint trajectory predictions for all humans in the scene. We then incorporate these multi-modal predictions into a SICNav Bilevel MPC problem that simultaneously solves for a robot plan (upper-level) and acts as a safety filter to refine the predictions for non-collision (lower-level). Combining planning and prediction refinement into one bilevel problem ensures that the robot plan and human predictions are coupled. We validate the open-loop trajectory prediction performance of our diffusion model on the commonly used ETH/UCY benchmark and evaluate the closed-loop performance of our robot navigation method in simulation and extensive real-robot experiments demonstrating safe, efficient, and reactive robot motion.**

I. INTRODUCTION

For humans, walking within crowds is considered a trivial task. For robots, however, there exist considerable challenges in achieving safe, collision-free, and efficient crowd navigation. In an environment like the one illustrated in Fig. 1, a robot must anticipate human movement and adjust its trajectory in real-time. This problem has inspired the development of a variety of trajectory prediction methods (e.g. [1]–[3]) that learn distributions of future human trajectories from historical data and demonstrate promising performance in *open-loop* prediction, generating trajectory forecasts for a test set of historical data excluded during training. However, incorporating open-loop predictions into a robot controller remains a challenge. Firstly, the robot's actions influence the future motion of humans, and vice versa, i.e. robot and human futures are coupled due to *interaction*. Secondly, the robot needs to account for potentially *multimodal* prediction distributions while choosing only one action to take. Finally, the robot needs to balance *opposing objectives* of efficiently reaching its goal, while ensuring safety.

To address interaction challenges, one method first predicts human and robot futures together, then during planning for

Sepehr Samavi, Anthony Lem, Sirui Chen, Qiao Gu, Angela P. Schoellig, and Florian Shkurti are with the University of Toronto Robotics Institute and the Vector Institute for Artificial Intelligence, Toronto, Canada. Angela P. Schoellig is also with the Technical University of Munich and the Munich Institute for Robotics and Machine Intelligence (MIRMI), Munich, Germany. Fumiaki Sato and Keijiro Yano are with Konica Minolta Inc., Japan. (*Corresponding author: Sepehr Samavi* sepehr@robotics.utias.utoronto.ca)
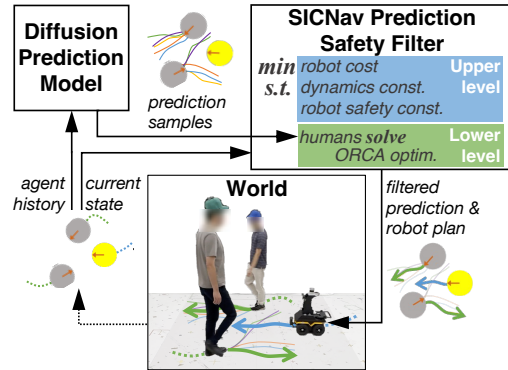
Fig. 1: For interactive robotic crowd navigation, we use the diffusion prediction model to forecast the future motion of humans as joint trajectory samples. The bilevel SICNav problem uses a particle-filter inspired approach to jointly refine predictions (lower-level) and optimize robot actions that satisfy safety constraints (upper-level). The robot uses solutions in receding horizon fashion. Video: `tiny.cc/sicnav_diffusion`

robot actions penalizes robot deviation from the prediction [4]. Yet the plan inevitably deviates as it is optimized towards the robot objective, reducing human prediction relevance. Other strategies dynamically model predictions, e.g. by representation as affine dynamical systems [5], or evaluating model gradients during planning [6], [7]. While these methods address prediction multimodality, they expose the critical trade-off between the safety and efficiency. They simplify the probabilistic nature of the planning problem by minimizing a single cost (e.g., expected cost [4], risk metric [6], or worst-case cost [7]). In contrast, introducing non-collision constraints maximizes efficiency while ensuring safety.

In this work, we introduce Safe and Interactive Crowd Navigation with Diffusion Trajectory Predictions (SICNav-Diffusion), a novel approach for safe robotic crowd navigation, illustrated in Fig. 1. We propose the Joint Motion Indeterminacy Diffusion (JMID) prediction model and combine with an extension of Safe and Interactive Crowd Navigation (SICNav) [8] to simultaneously plan robot actions while filtering the JMID predictions for safety. SICNav embeds Optimal Reciprocal Collision Avoidance (ORCA) [9], which models human collision-avoidance behavior, as constraints within the robot's planner to form a bilevel Model Predictive Control (MPC) problem. We extend the bilevel optimization problem such that the *lower-level* ORCA refines predictions to ensure non-collision, while the bilevel structure of the MPC couples the robot plan with these refined predictions. This way, the optimizer explores trajectory prediction modes generated by JMID and selects the one that best optimizes the robot objective, while guaranteeing non-collision.

Our contributions are threefold. First, we introduce JMID, where we extend a single-agent prediction model MID [3] to model a joint trajectory distribution for multiple humans. Second, we propose an extension to SICNav [8] to use multimodal prediction samples from JMID. Third, to evaluate our proposal, we first validate the open-loop prediction performance of JMID on the ETH/UCY trajectory benchmark [10], [11] and demonstrate JMID outperforms the joint prediction performance of state-of-the-art models. We then evaluate the closed-loop planning performance of our method in simulation and extensive in-lab robot experiments (240 runs) to compare our approach, SICNav-JMID, with other state-of-the-art prediction models in a single-robot multi-human environment. Finally, we demonstrate the effectiveness of our method in more than 8 km of outdoor real-world operation.

II. RELATED WORK

Trajectory forecasting: Human trajectory forecasting models use the history of agent positions in a scene to predict their future 2D bird's-eye-view trajectory for some horizon into the future. Recent methods focus on capturing the multi-modality of agent futures using a variety of learning-based architectures, e.g. Conditional Variational Autoencoders (CVAEs) with Gated Recurrent Units (GRUs) [1], autoregressive Transformers [2], or Diffusion [3]. Most predict the future of a single human in each inference [1], [3]. However, the future of all humans in the scene is coupled, thus, naively combining the individual forecasts for multiple humans may result in predictions of collisions.

To address this shortcoming, recent work has focused on jointly predicting all agents in the scene [2], [12], [13], for example, by introducing latent variables jointly inferred from all agents [2], or learning to build relational graphs between groups [12]. Some methods additionally guide the generative process using social force [14] to capture collision-avoidance [13], [15]. While these methods show promising results, guidance does not guarantee collision-free predictions despite adding a bias to the learned model. We extend [3] to jointly predict all agents, similar to these methods, however without introducing guidance. Instead, when using the forecasts our robot planner, SICNav, we leverage its interaction model [9] to refine the predictions to be collision-free, with the added benefit of coupling the refined predictions with the robot plan as well.

Combining forecasting with planning: Other works that incorporate predictions into robot planning focus on distinct challenges: interactivity of planning with predictions, accounting for multimodal prediction distributions, and ensuring safety through non-collision. Some methods elect to implicitly model interactions by using Reinforcement Learning (RL) to learn policies that implicitly reason about interactions in a learned policy with explicit [12], [16] or implicit [17], [18] predictions. While these methods can learn rich varied policies, even being deployed outdoors [18], it is challenging to evaluate the safety of such black-box policies.

Other methods [5], [7] explicitly model interactions between agents by leveraging prediction model gradients [7] in the planning step or outputting predictions as time-varying dynamics used in the planner [5]. While these methods

account for multimodal prediction distributions, they incorporate non-collision as an additional objective in the cost function rather than introducing constraints. Other research [19] ensures probabilistic non-collision constraint satisfaction via conformal prediction but considers unimodal predictions without modeling interactions in the planning step. A blended approach [20] addresses constraints primarily for the initial MPC step but remains reactive. In contrast, our method addresses the three challenges by selecting prediction modes based on whether the predictions are compatible (i.e. non-colliding) among the humans and with the robot plan, utilizing ORCA to refine the predictions. Our method incorporates this safety-filtering refinement into a bilevel MPC planner to explicitly model interactions between the humans and the robot while guaranteeing that generated action satisfies collision-avoidance constraints with the refined predictions.

III. METHODOLOGY

Fig. 1 presents a block diagram of our method. In short, we first use the history of agent motion to generate a set of JMID prediction samples, Υ. Then, we use the predictions and the latest measured state of the system as the initial state, $\mathbf{x}_o$, in a Bilevel MPC optimization problem that optimizes the robot's actions, $\mathbf{u}_{0:T-1}$ and system state, $\mathbf{x}_{0:T}$, containing the robot states and a refined version of the predicted human trajectories for a finite horizon T. We use this solution in receding horizon fashion: the robot executes the first robot action from the solution, $\mathbf{u}_0$, and repeats the MPC optimization with a newly measured state of the system, $\mathbf{x}_o$, and a newly sampled set of JMID predictions, Υ.

A. SICNav Bilevel MPC Problem

The environment consists of a robot, human agents, indexed by $j \in \{1,...,N\}$, and static obstacles in the form of line segments, indexed by $\tilde{l} \in \{N+1,...,N+M\}$. The state of the system is in continuous space and contains the state of the ego robot, the states of all N humans, and a set of weights describing the relative importance of each of S JMID samples in determining the intents of the humans.

Formally, at a discrete time step, t, the state is denoted as $\mathbf{x}_t = (\mathbf{x}_t^{(r)}, \tilde{\mathbf{x}}_t^{(1)},...,\tilde{\mathbf{x}}_t^{(N)}, \mathbf{w}_t) \in \mathcal{X}$, where the state of the robot, $\mathbf{x}_t^{(r)} \in \mathbb{R}^4$, consists of its 2D position, heading, and speed. The human state, $\tilde{\mathbf{x}}_t^{(j)} \in \mathbb{R}^4$, consists of 2D position, and 2D velocity $\forall j \in \{1,...,N\}$. We use a set of importance weights at each time step, $\mathbf{w}_t \in \mathbb{R}^S$, s.t. $\|\mathbf{w}_t\|_1 = 1$, to calculate a weighted average of S trajectory prediction samples obtained from JMID, which models future trajectories of all humans in the scene, $\mathbf{Y} = (\mathbf{Y}^{(1)},...,\mathbf{Y}^{(N)}) \in \mathbb{R}^{2TN}$, where $\mathbf{Y}^{(j)} = (\mathbf{y}_1^{(j)},...,\mathbf{y}_T^{(j)}) \in \mathbb{R}^{2T}$ is the predicted trajectory for individual agent j, composed of a sequence of predicted 2D positions $\mathbf{y}_t^{(j)} \in \mathbb{R}^2$. We denote the set of samples as $\Upsilon = \{\mathbf{Y}^{(1:N,s)}\}_{s=1}^S$, and refer to the break down of a particular sample s as $\mathbf{Y}^{(j,s)} = (\mathbf{y}_1^{(j,s)},...,\mathbf{y}_T^{(j,s)})$. We will also refer to the set of all samples for individual agent j at a specific time step t as $\Upsilon_t^{(j)} = \{\mathbf{y}_t^{(j,s)}\}_{s=1}^S$.

We separate the dynamics of the system into separate functions for the robot, the humans, and the evolution of

the importance weights through time. The robot dynamics, $\mathbf{x}_{t+1}^{(r)}=\mathbf{f}(\mathbf{x}_t^{(r)},\mathbf{u}_t)$, are modeled as a kinematic unicycle model, where the control input for the system, $\mathbf{u}_t\in\mathbb{R}^2$, is a vector of the linear and angular velocity of the robot for time step t. The dynamics of each human $\forall j\in\left\{1,...,N\right\}$, $\tilde{\mathbf{x}}_{t+1}^{(j)}=\mathbf{h}(\tilde{\mathbf{x}}_t^{(j)},\tilde{\mathbf{u}}_t^{(j)})$, are modeled as kinematic integrators, where $\tilde{\mathbf{u}}_t^{(j)}$ denote actions optimal with respect to ORCA. Note that actions $\tilde{\mathbf{u}}_t^{(j)}$ are used to obtain the safety-filtered predictions of the humans. Finally, the importance weights evolve according to $\mathbf{w}_{t+1}=\mathbf{g}(\mathbf{x}_t;\boldsymbol{\Upsilon}_t^{(1)},...,\boldsymbol{\Upsilon}_t^{(N)})$. We describe the method for finding the refined predicted human actions and the evolution of importance weights in Section III-C.

The stage cost, $l(\mathbf{x}_t,\mathbf{u}_t)=\mathbf{x}_t^\top\mathbf{Q}\mathbf{x}_t+\mathbf{u}_t^\top\mathbf{R}\mathbf{u}_t$, penalizes deviation from the robot's goal position and excessive control effort with positive semidefinite matrices $\mathbf{Q}$ and $\mathbf{R}$. The terminal penalty, $l_T(\mathbf{x}_T)=\beta\mathbf{x}_T^\top\mathbf{Q}\mathbf{x}_T$, where $\beta\geq1$, ensures stability of the controller (see Theorem 2.41 in [21]) with β selected to be sufficiently large.

We add constraints of the form $\mathbf{x}_t^\top\mathbf{P}_j\mathbf{x}_t\geq d_j^2, \forall j\in\left\{1,...,N\right\}$, where $\mathbf{P}_j\in\mathbb{R}^{n\times n}$ extracts the positions of the robot and the jth human from the state and d_j is the minimum distance to avoid collisions. We similarly add quadratic constraints to avoid collisions between the robot and static obstacles, which are represented as piece-wise linear functions. We also bound the input of the system to enforce the kino-dynamic limits of the real-world robot. Finally, for each time-step, we constrain the prediction of each human (indexed j) to lie in the solution set of the ORCA optimization problem [9], $\mathcal{O}^{(j)}(\mathbf{x}_t;\boldsymbol{\Upsilon}_{t+1}^{(j)})$, described in Section III-C. In short, the ORCA Quadratically Constrained Quadratic Program (QCQP) solves for a predicted human velocity minimizing distance to an intended velocity, obtained from the JMID samples for that human, subject to avoiding collisions with other agents and static obstacles.

With these definitions, the MPC problem for the robot can be formulated as an optimization problem,

$$\underset{\substack{\mathbf{x}_{0:T},\mathbf{u}_{0:T-1},\\ \tilde{\mathbf{u}}_{0:T-1}^{(1:N)},\mathbf{w}_{0:T-1}}}{\text{minimize}}\quad \sum_{t=0}^{T-1}l(\mathbf{x}_t,\mathbf{u}_t)+l_T(\mathbf{x}_T) \tag{1a}$$

$$\text{subject to}\quad \mathbf{x}_0=\mathbf{x}_o \tag{1b}$$

$$\mathbf{x}_{t+1}^{(r)}=\mathbf{f}(\mathbf{x}_t^{(r)},\mathbf{u}_t) \tag{1c}$$

$$\mathbf{u}_{min}^{(r)}\leq\mathbf{u}_t\leq\mathbf{u}_{\max}^{(r)} \tag{1d}$$

$$\Delta\mathbf{u}_{min}^{(r)}\leq\mathbf{u}_t-\mathbf{u}_{t-1}\leq\Delta\mathbf{u}_{\max}^{(r)} \tag{1e}$$

$$\mathbf{x}_t^\top\mathbf{P}_l\mathbf{x}_t\geq{d_l}^2 \tag{1f}$$

$$\tilde{\mathbf{u}}_t^{(j)}\in\mathcal{O}^{(j)}(\mathbf{x}_t;\boldsymbol{\Upsilon}_{t+1}^{(j)}) \tag{1g}$$

$$\tilde{\mathbf{x}}_{t+1}^{(j)}=\mathbf{h}(\tilde{\mathbf{x}}_t^{(j)},\tilde{\mathbf{u}}_t^{(j)}) \tag{1h}$$

$$\mathbf{w}_{t+1}=\mathbf{g}(\mathbf{x}_t;\boldsymbol{\Upsilon}_t^{(1)},...,\boldsymbol{\Upsilon}_t^{(N)}) \tag{1i}$$

where all the constraints (1c)-(1h) are defined for each time step, $\forall t\in\left\{0,...,T-1\right\}$, and the constraints (1g)-(1h) are defined for each human, $\forall j\in\left\{1,...,N\right\}$. The robot collision constraint (1f) is defined for each human agent and each static obstacle in the environment, $\forall l\in\left\{1,...,N,...,N+M\right\}$. The constraints (1g) are in the solution sets of constrained ORCA optimization problems, thus constituting the *lower-*
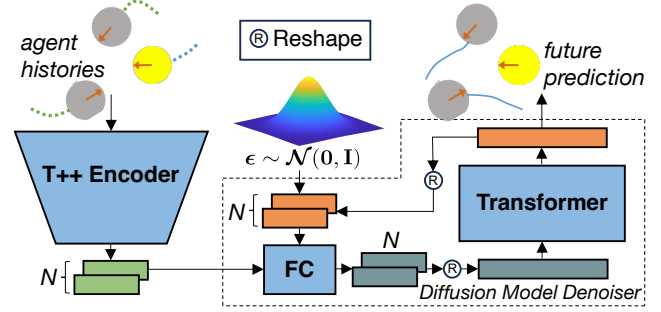

Fig. 2: Block diagram of JMID. Embeddings of agent histories (gray pedestrians and yellow robot) are generated for each agent in parallel by the encoder of Trajectron++ (T++). The embeddings, along with one sample per agent from a multivariate standard normal distribution, are passed in parallel through fully connected (FC) layers. Outputs are concatenated and passed through a Transformer. Denoising (dashed box) is repeated for K denoising steps to generate the final predictions.

level problems of the bilevel problem (1).

B. Diffusion Model of Joint Motion of Humans

We propose a human trajectory forecasting model that predicts future trajectories for multiple agents jointly. The model generates trajectory samples, $\boldsymbol{\Upsilon}$, which are used to parameterize the the lower-level ORCA problems, $\mathcal{O}$ in (1). Our model extends MID [3], a diffusion-based trajectory forecaster that predicts future motions for each human individually. The original model is an encoder-decoder network with two components: a borrowed encoder from Trajectron++ (T++) [1] and a Transformer-based decoder that learns a reverse diffusion process [22]. To make a prediction, MID requires the history of the target human and its neighboring agents. For the rest of the paper, we refer to MID as Individual Motion Indeterminacy Diffusion (iMID).

We extend iMID to Joint-MID (JMID), which produces *joint* samples for *all* humans in the scene, i.e. each sample contains a forecast of horizon T for all humans. While iMID models the future trajectory distribution of each agent independently, $\mathbf{Y}^{(j)}\sim p(\mathbf{Y}^{(j)}|\mathbf{x}_{-H:0}),\forall j\in\left\{1,...,N\right\}$, JMID models the joint future trajectory distribution of all agents $\mathbf{Y}\sim p(\mathbf{Y}^{(1)},...,\mathbf{Y}^{(N)}|\mathbf{x}_{-H:0})$. A block diagram of JMID is illustrated in Fig. 2. To obtain joint samples, we first query the encoder in parallel to generate a latent embedding for each human in the scene. We pass these together with noisy future trajectories through fully-connected layers to generate new latent embeddings. For iMID, the new embeddings would have been passed through the transformer decoder in parallel. Instead, we concatenate them and then pass them to the transformer. We do not use positional encoding between embeddings so transformer outputs are equivariant to concatenation ordering. Finally, we split the denoised output of the transformer into one prediction per agent, generating the joint samples $\boldsymbol{\Upsilon}$. To train the model, we follow [3] to maximize the variational lower bound. However, we alter the loss inputs' dimensions to account for the model output's varying dimension (i.e. number of humans being predicted) for each scene.
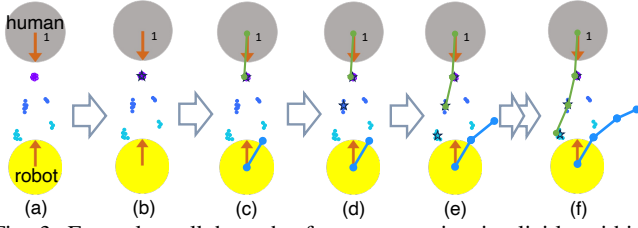
3

Fig. 3: Example walkthrough of steps occurring implicitly within the SICNav-JMID optimization (1) demonstrating our method's ability to select a mode of predictions to satisfy safety constraints. (a) Multicolored scatter illustrates JMID predictions with $S=9$ future trajectory samples for human 1 ($\mathbf{y}_1^{(1,s)}$ purple to $\mathbf{y}_3^{(1,s)}$ cyan). (b) Purple star illustrates the weighted average for the intent $\hat{\mathbf{y}}_1^{(1)}$ found using (4). (c) Solution from $t=0$ to $t=1$ with robot plan (blue line) and refined human prediction (green line) results in the robot going right and human left. (d)-(f) After the weights are updated using the human position $\tilde{\mathbf{x}}_1^{(1)}$ in (5)-(6), the process from (b) and (c) is repeated for subsequent steps resulting in the mode with the human going left being selected in the refined predictions.

C. Refining Prediction Samples to Satisfy Constraints

Lower Level ORCA Optimization Problem: Within the bilevel problem (1), our method refines the JMID predictions to satisfy explicit constraints using the sequences of ORCA optimization problems [9] embedded in (1g). The ORCA optimization problem for agent j is formulated as,

$$\mathcal{O}^{(j)}(\mathbf{x}_t; \boldsymbol{\Upsilon}_{t+1}^{(j)}) := \underset{\mathbf{v}}{\arg\min} \left\{ \left\| \mathbf{v} - \mathbf{v}_{\text{intent}}(\mathbf{x}_t; \boldsymbol{\Upsilon}_{t+1}^{(j)}) \right\|_2^2 : \atop \text{collision-avoidance} \right\} \quad (2)$$

where the optimization variable $\mathbf{v} \in \mathbb{R}^2$ is the velocity of agent j for one time step. The non-collision constraints are derived using a velocity obstacle approach (see Sec. 4 of [9]) and define linear collision avoidance constraints between agents j and agent $l \neq j$ and for each static obstacle. The constraints are linear with respect to $\mathbf{v}$ and parameterized by $\mathbf{x}_t$. The cost is based on a velocity vector from the agent's current position towards the agent's intended position at the next time step, $\hat{\mathbf{y}}_{t+1}$, obtained from JMID samples,

$$\mathbf{v}_{\text{intent}}(\mathbf{x}_t; \boldsymbol{\Upsilon}_{t+1}^{(j)}) = \frac{1}{\delta t} \left(\hat{\mathbf{y}}_{t+1}^{(j)} - \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tilde{\mathbf{x}}_t^{(j)} \right), \quad (3)$$

where δt is the discretization period in (1). To estimate this intended position, we find the weighted average of the human's predicted positions at time $t+1$ from the JMID samples,

$$\hat{\mathbf{y}}_{t+1}^{(j)} = \sum_{s=1}^{S} w_t^{(s)} \mathbf{y}_{t+1}^{(j,s)}. \quad (4)$$

Note that there is one importance weight per joint sample and not per human.

In (1h), we use the solution of (2), $\tilde{\mathbf{u}}_t^{(j)}$, to find the refined predicted position of the agent at the next time step, $\tilde{\mathbf{x}}_{t+1}^{(j)}$. This way, if the intended velocity for the agent is feasible with respect to constraints in (2), then the optimal velocity found by solving the problem is the one that moves the agent to the estimated intended position at the next time step. Otherwise, the ORCA optimization problem will find the *closest feasible* position to the estimated intent, acting as a safety filter for the JMID predictions.

Evolution of Importance Weights: At $t=0$, we initialize

the importance weights based on the likelihood estimates of the JMID samples, which we obtain by fitting Kernel Density Estimates (KDEs) [23]. For subsequent steps, in (1i), we compare the ORCA-refined positions, $\tilde{\mathbf{x}}_t^{(j)}$, found by solving (2) at the previous time step with the JMID sample's predicted positions, $\mathbf{y}_t^{(j,s)}$,

$$\tilde{w}_t^{(s)} = \exp\left(-\frac{1}{N\sigma} \sum_{j=1}^{N} \left\| \mathbf{y}_t^{(j,s)} - \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tilde{\mathbf{x}}_t^{(j)} \right\|_2^2 \right), \quad (5)$$

where σ is a hyperparameter. We normalize these weights, $\bar{w}_t^{(s)} = \tilde{w}_t^{(s)} / \sum_{s=1}^{S} \tilde{w}_t^{(s)}$, then incorporate the influence of the importance weights from the previous step, $w_{t-1}^{(s)}$, and normalize once again,

$$w_t^{(s)} = \frac{w_{t-1}^{(s)} \bar{w}_t^{(s)}}{\sum_{s=1}^{S} w_{t-1}^{(s)} \bar{w}_t^{(s)}}. \quad (6)$$

This way, the importance weights filter out prediction and robot plan combinations that would violate safety constraints.

Fig. 3 illustrates an example of refined human predictions and associated robot plan. For agent 1, we see its JMID prediction has two modes: right or left. As the robot jointly optimizes its plan and refines the human predictions through the horizon, at $t=0$ the robot plan moves right. Following the mode where the human also going to the right would result in a collision with the robot's plan (blue line). Since the ORCA problem avoids collisions, the refined human prediction moves closer to the left mode as t increases. As the variables in (1a) are optimized, the robot plan and ORCA cause the human predictions to move into non-colliding prediction modes.

IV. OPEN-LOOP PREDICTION EXPERIMENTS

We validate open-loop performance on the popular ETH/UCY benchmark. Our evaluation follows the standard leave-one-scene-out cross-validation protocol with the standard history length of 3.2s and forecast length of 4.8s.

Models: We compare our proposed joint prediction method, **JMID** with the following baselines: **iMID** [3], the original individual prediction method as an ablation study on whether JMID results in better scene-level performance; **AgentFormer** [2], to compare our method with a state-of-the-art autoregressive transformer-based method for joint human trajectory forecasting; and **T++** [1], a CVAE-based trajectory prediction model, whose encoder is adopted by iMID and JMID. For inference with the diffusion models, iMID and JMID, we use a Denoising Diffusion Probabilistic Model (DDPM) [24] version of the models, where denoising is conducted for 100 steps. However, the inference process may not run in real time with DDPMs, thus we also evaluate the models with a widely-used accelerated sampling version of the models, Denoising Diffusion Implicit Model (DDIM) [22], where denoising can be conducted with a variable number of steps, 2 in our case. We also use DDIM for real-time deployment.

Metrics: We evaluate predictive performance using,

- Best-of-20 Average Displacement Error (ADE): the lowest Euclidean distance between the ground truth

TABLE I: Human trajectory forecasting results on the ETH/UCY Dataset. SAD is SADE and SFD is SFDE. All metrics are reported in meters and lower is better ↓. **Bold** is best, <u>underline</u> second-best. On average JMID outperforms the other methods on scene-level metrics.

Method	ETH ADE/FDE	ETH SAD/SFD	Hotel ADE/FDE	Hotel SAD/SFD	Univ ADE/FDE	Univ SAD/SFD	Zara1 ADE/FDE	Zara1 SAD/SFD	Zara2 ADE/FDE	Zara2 SAD/SFD	Average ADE/FDE	Average SAD/SFD
T++	0.53/0.92	0.59/1.09	0.15/0.23	**0.20**/0.35	0.28/0.56	0.52/1.10	0.29/0.57	0.38/0.78	0.16/0.31	0.27/0.57	0.28/0.52	0.39/0.78
AgentFormer	**0.45/0.75**	**0.48/0.79**	**0.14/0.22**	0.24/0.46	**0.25/0.45**	0.62/1.31	**0.18/0.30**	**0.28/0.56**	**0.14/0.24**	0.30/0.62	**0.23/0.39**	0.38/0.75
iMID (DDIM)	0.63/1.02	0.66/1.13	0.24/0.41	0.28/0.52	0.29/0.56	0.47/1.00	0.26/0.52	0.35/0.74	0.20/0.37	0.32/0.66	0.32/0.58	0.42/0.81
iMID (DDPM)	0.54/0.86	0.58/0.96	0.21/0.35	0.26/0.47	0.29/0.54	0.52/1.10	0.24/0.43	0.35/0.70	0.19/0.33	0.33/0.67	0.29/0.50	0.41/0.78
JMID (DDIM)	0.59/1.00	0.61/1.05	0.21/0.34	0.24/0.40	0.34/0.70	**0.45/0.96**	0.23/0.45	0.29/0.59	0.19/0.39	0.24/0.50	0.31/0.58	0.37/0.70
JMID (DDPM)	0.53/0.91	0.54/0.93	0.18/0.27	**0.20/0.32**	0.32/0.66	0.47/1.01	0.22/0.41	0.30/0.58	0.17/0.32	**0.23/0.47**	0.28/0.51	**0.35/0.66**

trajectory and 20 samples obtained from the model, for each agent.

- Best-of-20 Final Displacement Error (FDE): the lowest Euclidean distance between positions at the final prediction step, T, of the ground truth trajectory and 20 samples obtained from the model, for each agent.
- Best-of-20 Scene-level Average Displacement Error (SADE) [25]: the lowest average ADE across all agents in the scene for 20 joint-prediction samples.
- Best-of-20 Scene-level Final Displacement Error (SFDE) [25]: the lowest FDE across all agents in the scene for 20 joint-prediction samples.

While ADE and FDE evaluate the performance of the model at the individual agent level (i.e. marginal distributions of each agent's future), SADE and SFDE evaluate the performance of the model at the scene level (i.e. joint distribution of all agents' futures). SADE and SFDE only evaluate predictions for different agents that come from the same joint sample, whereas ADE and FDE can mix-and-match predictions across different joint samples. Our end-goal is to operate a robot among a crowd of humans, so we place a greater emphasis on the scene-level metrics because they evaluate the joint behavior and interactions between agents that a model has learned.

Implementation: Our JMID model uses the same number of layers and dimensions as iMID [3]. We train our model using the Adam optimizer with a learning rate of 10^{-4} and a batch size of 64.

Results and Discussion: Table I summarizes the results of our experiments. Our method, JMID, outperforms the baselines in the scene-level metrics on most test splits and on average. While AgentFormer performs the best in the individual-level metrics, as discussed previously, the scene-level metrics are more meaningful than the individual-level metrics because they evaluate how well a model has learned the interactions between pedestrians. Therefore, our proposed method performs the best in the most meaningful metrics.

V. CLOSED-LOOP EXPERIMENTS

A. Simulation Experiments

Testing environment: We evaluate the performance of our planner in the CrowdSimPlus simulator, a commonly used crowd navigation simulation environment [8], [26]. We generate 500 random initial and final positions for the humans in the environments with a total of $N=3$ simulated humans. The simulated testing area is a $1.75m$-wide corridor with initial positions and goals on either end. In order to ensure a high density of agents in the corridor, we constrain the space using static obstacles, formulated as line-segments. The simulated humans are modeled as ORCA agents, however with randomly sampled attributes for radius buffer (unobservable amount added to radius), collision avoidance time horizon (see [9]), goal location, and maximum velocity. None of these attributes are observable to the robot.

Navigation Methods: We compare four variants of our SICNav Bilevel MPC (1), each using a different human trajectory forecasting model: **SICNav-JMID** using our prosposed JMID samples, **SICNav-iMID** using individual agent samples from iMID [3], **SICNav-AF** using joint samples from AgentFormer [2], and **SICNav-CVG** a non-learning-based constant-velocity-goal baseline which projects the human's current velocity forward in time to infer their goal. To evaluate the effect of SICNav's bilevel modeling of interactions we compare with **MPC-CVMM** which has the same cost and constraint functions as SICNav except using Constant Velocity Motion Model (CVMM) human predictions that remain constant during the MPC optimization, neglecting interactions. Nonetheless CVMM predictions have been shown to be a competitive paradigm for prediction [27]. We also compare our method to the reinforcement learning methods Socially Attentive Reinforcement Learning (**SARL**) [26] and Relational Graph Learning (**RGL**) [17], as well as a classical reactive baseline, Dynamic Window Approach (**DWA**) [28] which was used on some of the oldest crowd-navigating robots.

Implementation: The prediction models JMID, iMID, and AgentFormer are implemented in PyTorch and wrapped with ROS to deploy inference on the robot at 10 Hz. We pretrain the models on the JRDB dataset [29], which contains a robot agent in its scenes. The training details are the same as Sec. IV. We then fine-tune the models using data collected from training scenarios in the simulator.

Performance metrics: To evaluate robot navigation performance, we adopt the following metrics from literature [8]:

- *Success Rate:* The rate of scenarios in which the robot is able to arrive at its goal within $30s$. It measures the success of the robot's navigation through the crowd.
- *Average Navigation Time:* The average time to completion, which indicates the robot's navigation efficiency.
- *Collision Frequency:* The frequency the robot is in a collision state with another agent, which indicates the safety of the robot.
- *Freezing Frequency:* The frequency of the robot stopping completely, which measures how often the robot can potentially get stuck in a deadlock with the human agents (encounters the Freezing Robot Problem (FRP)).

Results and Discussion: Table II summarizes the results of our simulation experiments. SICNav-JMID outperforms the other methods in success rate and freezing frequency. The results suggest the joint prediction of human trajectories

TABLE II: Quantitative comparison in 500 randomly generated test scenarios with 1 robot and 3 simulated humans following a randomized ORCA policy. Arrows indicate desired relative values. **Bold** is best, <u>underline</u> second-best.

Approach	Success Rate ↑	Avg Nav Time (s) ↓	Collision Freq. (s^{-1}) ↓	Frozen Freq. (s^{-1}) ↓
SICNav-JMID	**1.00**	<u>6.82</u>	0.006	**0.20**
SICNav-iMID	<u>0.98</u>	7.54	0.007	<u>0.38</u>
SICNav-AF	0.97	7.67	0.007	0.43
SICNav-CVG	0.95	8.86	**0.005**	0.73
MPC-CVMM	0.97	**6.53**	0.018	0.55
SARL [26]	0.89	7.09	<u>0.006</u>	0.96
RGL [17]	0.89	6.92	0.007	1.62
DWA [28]	0.85	11.52	0.032	0.93
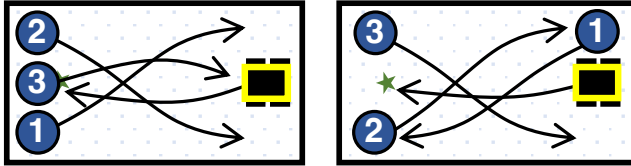
(a) Variant 1   (b) Variant 2

Fig. 4: Real-robot experiment scenarios with variants 1 (a) and 2 (b). The green star indicates the robot's goal, numbered circles indicate human start positions, and arrows their goals. In a scenario with $N \in \{1,2,3\}$ humans, positions greater than N are empty

with JMID is more effective than individual predictions with Motion Indeterminacy Diffusion (MID) or the joint predictions from AgentFormer. The results also suggest the simple projection of human velocity with SICNav-CVG is not as effective as the learning-based methods resulting in a lower success rate and higher navigation time. The MPC-CVMM baseline has a lower navigation time than the SICNav variants, however at the cost of a much higher collision frequency. SARL and RGL have a much lower success rate and freeze more often than the SICNav. The reactive DWA method has the lowest success rate, highest navigation time, and the highest collision frequency.

B. In-Lab Robot Experiments

Testing Environment: We also evaluate the performance of our approach on a real robot in a controlled environment similar to [8]. We use a Clearpath Jackal differential drive robot pictured in Fig. 1. The robot weighs $20kg$ and measures $46cm$W$\times 60cm$L$\times 50cm$H. Our operating environment is an indoor space measuring $5m \times 9m$. We conduct a total of 240 runs with a varying number of humans $N \in \{1,2,3\}$ and two configuration variants, illustrated in Fig. 4 (i.e. six scenarios). For each run of a particular scenario, all human agents were instructed to move from their labelled starting positions to the opposite corner of the room (illustrated by the arrows). In variant 1 (Fig. 4a) all humans start from the opposite side of the room from the robot while in variant 2 one human moves in the same direction as the robot (Fig. 4b). These scenarios are designed to cause a conflict zone in the middle of the room where all agents need to interact. We repeat 10 runs per variant and navigation method being tested. We use a VICON motion capture system to measure the positions of the robot and human agents in the environment and use an Extended Kalman Filter (EKF) and Kalman Filters (KFs) to track the positions and velocities of the robot and humans, respectively.

Navigation Methods: We evaluate the four variants of

(a) Robot Navigation Efficiency
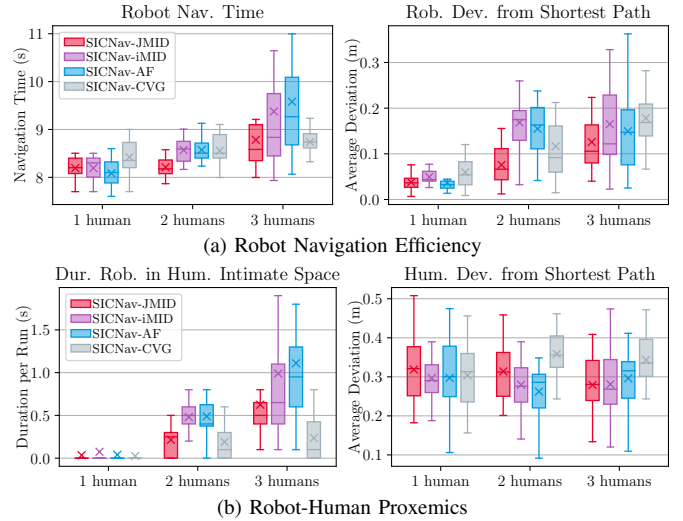

(b) Robot-Human Proxemics

Fig. 5: Box-and-whisker plots of robot navigation performance in terms of (a) efficiency and (b) proxemics in scenarios with varying numbers of humans. Median values are illustrated by the line in the box-and-whisker plots and mean values are indicated by the ×. Our method SICNav-JMID (a) results in the most efficient robot navigation with respect to both metrics, and (b) spends the least time in the intimate space of humans compared to other learning-based methods. SICNav-CVG spends less time in intimate space because it causes humans to deviate from their shortest paths at higher human densities.

SICNav tested in the simulation experiments: SICNav-JMID, SICNav-iMID, SICNav-AF, and SICNav-CVG.

Implementation: We use the Robot Operating System (ROS) 1 middleware. The robot accepts linear and angular velocity commands to track by its low level controller. We run the algorithms on a laptop with 24-core Intel Core i9-14900HX CPUs and an NVIDIA RTX 4080 GPU. The prediction models are fine-tuned as in the simulation experiments, however with data collected with the robot. The inference code for the models is wrapped in a ROS node that publishes the most recent prediction samples at 10 Hz, with inference taking $<0.1s$ using 2-step DDIM, as in Sec. IV. The controller ROS node also runs the controller in real-time replanning at 10 Hz (solve-times taking $<0.1s$) with CasADi as the modeling language and the Acados [30] solver for optimization. We discretize time at $\delta t = 0.25s$ and use an MPC horizon of $T = 2s$.

Results and Discussion: We evaluate the robot's navigation efficiency using two metrics: the deviation from the shortest path to the goal and the robot's navigation time (i.e. time to reach the goal in each run). Fig. 5a shows the box-and-whisker plots of the robot navigation efficiency metrics for the different navigation methods. We observe for both metrics, in scenarios with $N = 1$, all methods behave similarly. However, as the number of humans increases to $N \in \{2,3\}$, using SICNav-JMID results in both a lower deviation from the shortest path to the goal and a lower navigation time compared to the other methods.

We also analyze the interaction patterns between the robot and humans when using the different navigation methods. First, we analyze the effect of the robot's navigation on the humans by looking at the deviation of the humans from their shortest paths to the goal. Fig. 5b shows the box-and-whisker

(a) Top view snapshots of SICNav-CVG robot
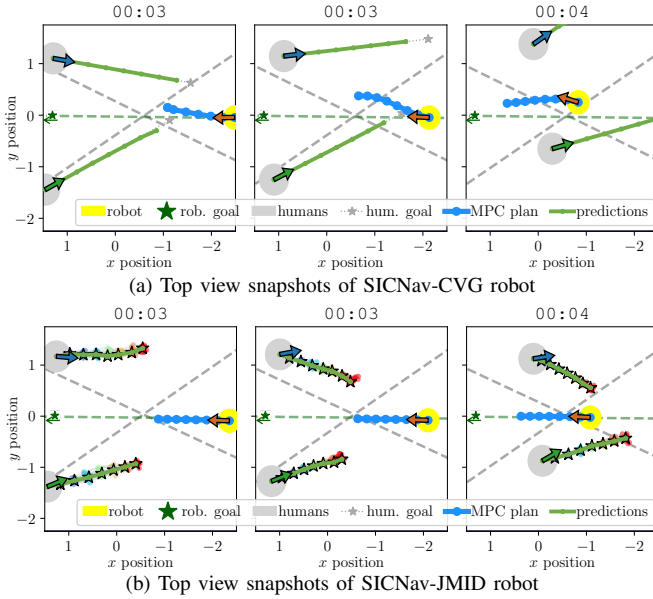

(b) Top view snapshots of SICNav-JMID robot

Fig. 6: We illustrate two similar scenarios with SICNav-CVG (a) and SICNav-JMID (b). We plot temporal snapshots of the scenarios with dashed lines (gray for humans, green for robot) indicating each agent's shortest-path-to-goal. The SICNav-CVG robot produces solutions that drive towards the blue-hat human, causing both the human and robot to deviate from their shortest paths, while the JMID predictions cause SICNav-JMID to produce solutions where all agents stay closer to their shortest-paths-to-goal.

plots of the humans' deviation from their shortest paths to the goal for the different navigation methods (right). We observe in all three scenarios, the learning-based methods SICNav-JMID, SICNav-iMID, and SICNav-AF perform similarly on this metric. However, in the scenarios with $N \in \{2,3\}$, SICNav-CVG causes a higher deviation of the humans from their shortest paths to the goal.

Second, to analyze proxemics, we look at the duration of time the robot spends within the *intimate space* [31], i.e. when the robot is within $0.45m$ of a human. Fig. 5b illustrates the box-and-whisker plots of the duration of time the robot spends in the intimate space of a human per run for the different navigation methods (left). In the scnarios with $N{=}1$, we see all methods spend almost no time in the personal space of the human. In the scenarios with $N \in \{2,3\}$, we observe SICNav-JMID spends less time in the intimate space of the humans compared to the other learning-based methods, SICNav-iMID and SICNav-AF. However, the non-learning-based baseline SICNav-CVG spends the least time in the personal space of the humans. Recalling the results from Fig. 5b, we see that this lower amount of time spent in the personal space of the humans comes at the cost of a higher deviation of the humans from their shortest paths.

We interpret this result as meaning SICNav-CVG causes the humans to *run away* or *avoid* getting close to the robot, which is undesirable. We illustrate an example occurrence of this phenomenon from the runs of the experiments in Fig. 6. We analyze two similar scenarios with SICNav-CVG (6a) and SICNav-JMID (6b). In the first snapshots for both methods, we observe the blue-hat agent's heading (blue arrow) deviates more from their optimal path (gray dashed line) than the green-hat agent's heading. With CVG
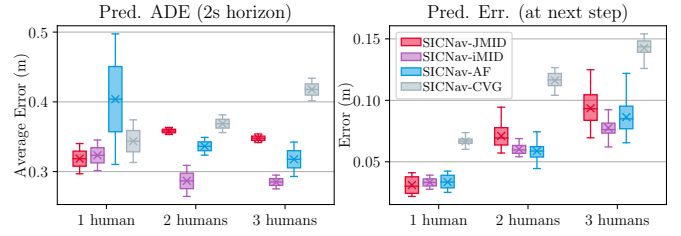

Fig. 7: Box-and-whisker plots of prediction performance in the closed loop experiments. We evaluate for all pedestrians Average Displacement Error (ADE) over the MPC horizon (left) and prediction error at the next step the robot is used (right). The learning-based methods outperform the CVG baseline in terms of both metrics. Surprisingly, despite outperforming all methods in terms of navigation performance, the JMID model does not consistently outperform the other learning-based methods in terms of the prediction metrics.

predictions (6a), the blue hat agent is predicted to deviate far from their optimal path. As a result, at the next control step (middle snapshot), the SICNav-CVG robot turns toward the blue-hat agent forcing the blue-hat agent to deviate even further from their optimal path in the third control step (final graph in the snapshots). In contrast, with JMID (6b) predictions, at the second control step (middle graph in the snapshots), the JMID model predicts the blue-hat agent intends to return toward their optimal path, despite the agent's heading pointing away from the path. As such, the robot's MPC plan does not turn towards the blue-hat agent, allowing the robot and blue-hat agent to stay nearly optimal.

Finally, we evaluate prediction performance in the closed-loop experiments. Fig 7 illustrates the ADEs over the 2s horizon (left) as well as the predictive error at the next control step (right). To describe the latter metric, recall we use the MPC solutions in receding horizon fashion, thus the action from a solution obtained at time t is only used until the next solution becomes available at $t + \delta t$. We evaluate the predictive error at this moment as it is the latest time the solution is used by the robot. We see for both metrics, the learning-based methods outperform the CVG baseline. Surprisingly, although the trajectory prediction results from JMID do not consistently outperform the other learning-based methods in every scenario, JMID nonetheless achieves the best closed-loop navigation performance. This observation highlights the importance of joint prediction in crowd navigation, yet also suggests the possible mismatch between open-loop prediction metrics and closed-loop robot navigation performance, as also shown in [4].

C. Outdoor Robot Experiments

We deploy a robot running SICNav-JMID outdoors using only on-board sensing to demonstrate the viability of SICNav-MID in the absence of a motion capture system for perception and localization and in the face of larger crowds than the in-lab experiments outlined in Sec. V-B. We use a Clearpath Jackal robot only using two sensors: an Ouster OS1-128 lidar and wheel odometry. For Simultaneous Localization and Mapping (SLAM) we use Cartographer [32]. For human perception and tracking, we use YOLOv9 to generate 2D detections on images generated by the lidar [33] and aUToTrack for human tracking in 3D [34]. For

obstacle map generation and global path planning we use the ROS 1 `move_base` framework. We run SICNav-JMID with implementation details and compute as in Sec. V-B. The computer also runs the object detection and tracking software but we run sensor driver and localization software on the robot's on-board computer.

We operate the robot in sidewalk and campus environments in 21 scenarios for a total 8.86 km over 2 hours 26 minutes. In each scenario, an operator provides a robot goal on the map using a wireless tablet. The robot then generates a global path to the goal and SICNav-JMID begins tracking the path. In our experiments, the robot encountered up to 8 pedestrians at a time and required minimal manual takeovers, successfully driving autonomously 99.03% of the time. We provide qualitative analysis in the video: tiny.cc/sicnav_diffusion.

VI. Conclusion

In this work we propose a novel method for joint human trajectory forecasting and robot navigation in crowded environments. We introduce the JMID model, which generates joint trajectory forecasts for all agents in the scene. We then propose a Bilevel MPC method that uses these forecasts to simultaneously refine the predictions for safety and optimize the robot's actions. We validate the open-loop performance of our prediction model on the ETH/UCY benchmark and show it outperforms state-of-the-art methods in terms of joint forecasting. We also conduct real-robot experiments to show our method outperforms state-of-the-art methods in terms of robot navigation efficiency and safety in a controlled environment and demonstrate the viability of our method in uncontrolled outdoor environments.

References

[1] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-Feasible Trajectory Forecasting With Heterogeneous Data," in *European Conference on Computer Vision*, 2020.

[2] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *IEEE/CVF International Conference on Computer Vision*, 2021.

[3] T. Gu, G. Chen, J. Li, *et al.*, "Stochastic trajectory prediction via motion indeterminacy diffusion," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 092–17 101.

[4] S. Poddar, C. Mavrogiannis, and S. S. Srinivasa, "From Crowd Motion Prediction to Robot Navigation in Crowds," in *IEEE/RSJ International Conference on Intelligent Robots and System*, 2023.

[5] B. Ivanovic, A. Elhafsi, G. Rosman, A. Gaidon, and M. Pavone, "MATS: An Interpretable Trajectory Forecasting Representation for Planning and Control," in *Conference on Robot Learning*, 2020.

[6] H. Nishimura, B. Ivanovic, A. Gaidon, M. Pavone, and M. Schwager, "Risk-sensitive sequential action control with multi-modal human trajectory forecasting for safe crowd-robot interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 11 205–11 212.

[7] S. Schaefer, K. Leung, B. Ivanovic, and M. Pavone, "Leveraging neural network gradients within trajectory optimization for proactive human-robot interactions," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 9673–9679.

[8] S. Samavi, J. R. Han, F. Shkurti, and A. P. Schoellig, "SICNav: Safe and interactive crowd navigation using model predictive control and bilevel optimization," *IEEE Transactions on Robotics*, vol. 41, pp. 801–818, 2024.

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

[10] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by Example," *Computer Graphics Forum*, vol. 26, no. 3, pp. 655–664, 2007.

[11] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *IEEE International Conference on Computer Vision*, 2009, pp. 261–268.

[12] J. Li, C. Hua, J. Yao, *et al.*, "Multi-agent dynamic relational reasoning for social robot navigation," 2024. arXiv: 2401.12275.

[13] H. Chen, J. Ding, Y. Li, Y. Wang, and X.-P. Zhang, "Social Physics Informed Diffusion Model for Crowd Simulation," *AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, pp. 474–482, 2024.

[14] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.

[15] J. Yue, D. Manocha, and H. Wang, "Human Trajectory Prediction via Neural Social Physics," in *European Conference on Computer Vision*, 2022.

[16] S. Liu, P. Chang, Z. Huang, *et al.*, "Intention Aware Robot Crowd Navigation with Attention-Based Interaction Graph," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 12 015–12 021.

[17] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, "Relational Graph Learning for Crowd Navigation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2020, pp. 10 007–10 013.

[18] Z. Xie and P. Dames, "DRL-VO: Learning to Navigate Through Crowded Dynamic Scenes Using Velocity Obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023.

[19] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 5116–5123, 2023.

[20] L. Heuer, L. Palmieri, A. Rudenko, A. Mannucci, M. Magnusson, and K. O. Arras, "Proactive model predictive control with multi-modal human motion prediction in cluttered dynamic environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023, pp. 229–236.

[21] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Nob Hill Publishing, LLC, 2019.

[22] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," in *International Conference on Learning Representations*, 2021.

[23] L. Thiede and P. Brahma, "Analyzing the Variety Loss in the Context of Probabilistic Trajectory Prediction," in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9953–9962.

[24] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[25] E. Weng, H. Hoshino, D. Ramanan, and K. Kitani, "Joint metrics matter: A better standard for trajectory forecasting," in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 20 315–20 326.

[26] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *International Conference on Robotics and Automation*, 2019, pp. 6015–6022.

[27] C. Scholler, V. Aravantinos, F. Lay, and A. Knoll, "What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1696–1703, Apr. 2020.

[28] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[29] R. Martin-Martin, M. Patel, H. Rezatofighi, *et al.*, "JRDB: A dataset and benchmark of egocentric robot visual perception of humans in built environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[30] R. Verschueren, G. Frison, D. Kouzoupis, *et al.*, "Acados—a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, vol. 14, no. 1, pp. 147–183, 2022.

[31] E. T. Hall, *The hidden dimension*. New York: Anchor Books, 1990.

[32] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1271–1278.

[33] C.-Y. Wang and H.-Y. M. Liao, "Yolov9: Learning what you want to learn using programmable gradient information," 2024.

[34] K. Burnett, S. Samavi, S. L. Waslander, T. D. Barfoot, and A. P. Schoellig, "aUToTrack : A Lightweight Object Detection and Tracking System for the SAE AutoDrive Challenge," in *Conference on Computer and Robot Vision*, 2019.