Safe Diffusion Model Predictive Control for Interactive Robotic Crowd Navigation

Sepehr Samavi[*]
University of Toronto
Vector Institute
Canada

Anthony Lem
University of Toronto
Vector Institute
Canada

Fumiaki Sato
Konica Minolta Inc.
Japan

Sirui Chen
University of Toronto
Vector Institute
Canada

Qiao Gu
University of Toronto
Vector Institute
Canada

Keijiro Yano
Konica Minolta Inc.
Japan

Angela P. Schoellig
Technical University of Munich
University of Toronto
Vector Institute
Germany and Canada

Florian Shkurti
University of Toronto
Vector Institute
Canada

Abstract: To navigate a crowd safely without collisions, robots need to interact with humans by predicting potential future motion and duly reacting. While learning-based prediction models have proven to be a promising approach towards generating human trajectory forecasts, utilizing these models in a robot controller presents the challenges of accounting for the coupling of planned robot motion with human predictions and ensuring that both predictions and robot actions are safe. Towards addressing these challenges, we present a receding horizon crowd navigation method for single-robot multi-human environments. We propose a diffusion model to generate multi-modal human trajectory forecasts that we use to parameterize a Bilevel Model Predictive Control (MPC) problem, which jointly filters the predictions to satisfy safety constraints (lower-level) and solves for a robot plan (upper-level) that is coupled with the filtered predictions. We evaluate the open-loop trajectory prediction performance of our diffusion model on the commonly used ETH/UCY benchmark, and analyze the closed-loop performance of our robot navigation method in extensive real-robot experiments to demonstrate safe and efficient robot motion.

1 Introduction

For humans, walking or otherwise moving within crowds is considered a trivial task. For robots, however, there exist considerable challenges to achieving safe (i.e. collision-free) crowd navigation (e.g. in an environment similar to the *world* in Fig. 1). Inspired by this problem, the community has developed a variety of trajectory prediction methods (e.g. [1, 2, 3, 4]), which learn distributions over future human trajectories from historical data. However, one major challenge is incorporating these predictions in a robot controller, especially since forecasts are often multimodal and uncertain whereas the robot is only able to take a single action.

Classical approaches often use forecasts generated from a prediction model to solve for robot actions that avoid any possible future collisions (e.g [5, 6, 7, 8]). These approaches present a straightforward

[*]Corresponding author: Sepehr Samavi, `s.samavi@utoronto.ca`

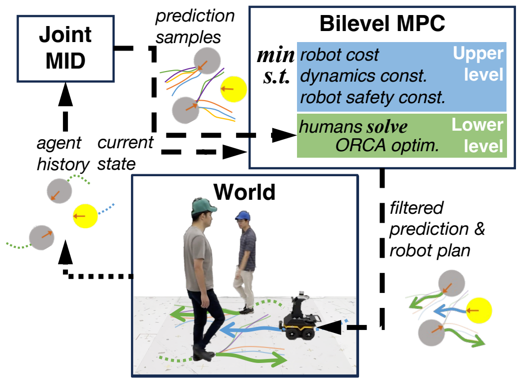


Figure 1: Block diagram of our method. The Joint Motion Indeterminacy Diffusion (JMID) model generates prediction samples that forecast the intended motion of multiple humans. The Bilevel MPC method uses a particle-filter inspired approach to jointly filter these predictions (lower-level) and produce robot actions that satisfy safety constraints (upper-level).

way to integrate accurate predictions into control and are safe in that they enable the robot to satisfy safety constraints [9, 7]. One major shortcoming is that they decouple human predictions from robot actions and have been shown, as a result, to exhibit the Freezing Robot Problem (FRP) [10, 7], where predictive uncertainty grows to the point that the robot cannot compute a safe path with sufficient certainty. Recent work has even shown that, in a decoupled framework, using a prediction model with high accuracy to replace one with lower accuracy, does not lead to better closed-loop navigation performance [11].

These shortcomings reveal a second challenge for robotic crowd navigation: agents influence each other and need to *interactively* coordinate movement. In other words, the future motions of all agents in the scene are coupled. To address these shortcomings, more sophisticated closed-loop prediction and planning methods jointly generate predictions and robot motion. Some methods do away with the separation of a prediction module from control and instead use Reinforcement Learning (RL) techniques to learn policies that implicitly reason about predictions and interactions (e.g. [12, 13, 14]). While these methods can learn rich varied policies from data, they often combine the opposing objectives of safety and performance into a single reward specification (e.g. in collision-avoiding RL [14]), leading to difficult-to interpret and potentially unsafe output, or need to rely on post-processing of the policy's actions to satisfy safety constraints (e.g. via safety filters [15]). Other methods who produce more interpretable results use less sophisticated prediction models of humans that are explicitly integrated into a planning and control framework to model cooperation (e.g. Gaussian Processes (GPs) [16, 17]) or competition (e.g. linear Inverse Reinforcement Learning (IRL) [18]). These methods, however, often rely on simplifying assumptions about human behavior and are not designed to handle multimodal human predictions.

In this work, we propose a novel method that combines a diffusion model for human trajectory prediction with a Bilevel MPC crowd navigation method to address the challenges of safe and interactive crowd navigation. Our method uses a particle-filter inspired approach to jointly filter multimodal human trajectory forecasts and produce robot actions that satisfy explicit safety constraints.

The contributions of this letter are threefold, First, we present JMID, an extension of single-agent prediction model Motion Indeterminacy Diffusion (MID) [4], that generates joint forecast samples of the intended motion of multiple humans. We demonstrate that JMID outperforms the joint prediction performance of state-of-the-art models on the ETH/UCY trajectory benchmark [19, 20]. Second, propose SICNav-MID, an extension of a Bilevel MPC crowd navigation method Safe and Interactive Crowd Navigation (SICNav) [21]. We extend the interactive controller using a particle-filter inspired approach to jointly filter the forecast samples produced by JMID and produce robot actions such that the all agents satisfy explicit safety constraints. Finally, we evaluate closed-loop planning performance in extensive real-robot experiments (240 runs) to compare using our approach, SICNav-JMID, with other state-of-the-art prediction models in single single-robot multi-human environment.

2 Problem Formulation

The environment consists of a robot, simulated human agents, indexed $j \in \mathbb{I}_{1:N} = \{1, \dots, N\}$, and static obstacles in the form of line segments and indexed, $\tilde{l} \in \left\{N+1, \dots, N+M\right\}$. The objective of the robot is to move to a goal position starting from an initial configuration, $\mathbf{x}_0$, without any collisions with static obstacles or humans.

The system state is continuous and concatenates the states of all agents, one robot and N humans. At time step t, the state is, $\mathbf{x}_t = (\mathbf{x}_t^{(r)}, \mathbf{x}_t^{(1)}, \dots, \mathbf{x}_t^{(N)}, \mathbf{w}_t) \in \mathcal{X}$, where the state of the robot, $\mathbf{x}_t^{(r)} \in \mathbb{R}^4$, consists of its 2D position, heading, and linear velocity at $t-1$. The state of the j^{th} human, $\mathbf{x}_t^{(j)} \in \mathbb{R}^6$, consists of its 2D position, 2D velocity, and their predicted goal estimate. The predicted goal estimate is a 2D position estimating the human's intended position at the next time step, $t+1$, obtained from a weighted average of S predicted positions for $t+1$, $\boldsymbol{\Upsilon}_{t+1}$, using the weights $\mathbf{w}_t \in \mathbb{R}^S \text{s.t.} \left\|\mathbf{w}_t\right\| = 1$.

These predicted positions are obtained from JMID, which outputs a set of forecast samples for a finite horizon, $\boldsymbol{\Upsilon}_{1:T} = \left\{\mathbf{y}_{1:T}^{(1)}, \dots, \mathbf{y}_{1:T}^{(S)}\right\}$ given a history of states, $\mathbf{x}_{-H:0} := (\mathbf{x}_{-H}, \mathbf{x}_{-H+1}, \dots, \mathbf{x}_0)$. Each sample $\mathbf{y}^{(s)} := \left(\check{\mathbf{p}}_{1:T}^{(1,s)}, \dots, \check{\mathbf{p}}_{1:T}^{(j,s)}, \dots, \check{\mathbf{p}}_{1:T}^{(N,s)}\right)$ contains probable future positions for *all* humans in the scene for T time steps into the future, $\check{\mathbf{p}}_{1:T}^{(j,s)} = \left(\check{\mathbf{p}}_1^{(j,s)}, \dots, \check{\mathbf{p}}_T^{(j,s)}\right)$.

We separate the dynamics of the system into separate functions for the robot and the humans. 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 $\forall j \in \mathbb{I}_{1:N}$, $\mathbf{x}_{t+1}^{(j)} = \mathbf{h}(\mathbf{x}_t^{(j)}, \tilde{\mathbf{u}}_t^{(j)})$, are modeled as kinematic integrators, where $\tilde{\mathbf{u}}_t^{(j)}$ denote the human actions. Note that these actions are not an input to the system, but rather predictions of the human actions. Finally, we have the dynamics of the importance weights, $\mathbf{w}_{t+1} = \mathbf{g}(\mathbf{x}_t^{(1)}, \dots, \mathbf{x}_t^{(N)}; \boldsymbol{\Upsilon}_t)$. We describe the method for finding the predicted human actions and the dynamics function for the importance weights in Section **??**.

3 Methodology

Fig. 1 illustrates the block diagram of our method. In short, we first use the history of agent motion to generate a set of JMID prediction samples, $\boldsymbol{\Upsilon}_{1:T}$ and also obtain the latest state of the system, $\mathbf{x}_0$. Then, we use the initial state and predictions in a Bilevel MPC optimization problem that optimizes the robot's trajectory (upper-level) in addition to optimizing a filtered version of the predicted human trajectories (lower-level), $(\mathbf{x}_{1:T}, \mathbf{u}_{0:T-1})$. We use this solution in receding horizon fashion, i.e. the robot executes the first robot action, $\mathbf{u}_0$, and repeats the process with a newly measured state of the system, $\mathbf{x}_0$, and a newly sampled set of JMID predictions, $\boldsymbol{\Upsilon}_{1:T}$.

3.1 SICNav Bilevel MPC Problem

We first describe the MPC bilevel optimization problem for the robot, which is an extension of the SICNav [21] problem to incorporate data-driven prediction samples. The components of the stage cost penalize deviation from the robot's goal position and excessive control effort, $l(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{x}_t^\top \mathbf{Q} \mathbf{x}_t + \mathbf{u}_t^\top \mathbf{R} \mathbf{u}_t$, with positive semidefinite matrices $\mathbf{Q}$ and $\mathbf{R}$. We also define a terminal penalty as, $V_f(\mathbf{x}_T) = \beta \mathbf{x}_T^\top \mathbf{Q} \mathbf{x}_T$, where $\beta \geq 1$ is selected to be sufficiently large to ensure stability of the controller (See Theorem 2.41 in [22]). To avoid collisions between the robot and any human agents we add constraints of the form, $\mathbf{x}_t^\top \mathbf{P}_j \mathbf{x}_t \geq d_j^2, \forall j \in \mathbb{I}_{1:N}$, where, $\mathbf{P}_j \in \mathbb{R}^{n \times n}$ is a matrix that extracts the positions of the robot and each human agent j from the state and d_j is the minimum permissible distance between the two agents. For each static obstacle, we implement a piece-wise function that calculates the closest point on the line segment to the position of the robot and add a quadratic constraint similar to the robot-agent collision constraints. We also bound the input of the system in order to meet the kino-dynamic limits of a real-world robot.

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

$$\begin{aligned}
\underset{\substack{\mathbf{x}_{0:T}, \mathbf{u}_{0:T-1}, \\ \tilde{\mathbf{u}}^{(0:N)}_{0:T-1}, \mathbf{w}_{0:T-1}}}{\text{minimize}} \quad & \sum_{t=0}^{T-1} l(\mathbf{x}_t, \mathbf{u}_t) + V_f(\mathbf{x}_T) && \text{(1a)} \\
\text{subject to} \quad & \mathbf{x}_0 = \mathbf{x}_0 && \text{(1b)} \\
& \mathbf{x}^{(r)}_{t+1} = \mathbf{f}(\mathbf{x}^{(r)}_t, \mathbf{u}_t) && \text{(1c)} \\
& \mathbf{u}^{(r)}_{min} \leq \mathbf{u}_t \leq \mathbf{u}^{(r)}_{\max} && \text{(1d)} \\
& \Delta\mathbf{u}^{(r)}_{min} \leq \mathbf{u}_t - \mathbf{u}_{t-1} \leq \Delta\mathbf{u}^{(r)}_{\max} && \text{(1e)} \\
& \mathbf{x}_t^\top \mathbf{P}_l \mathbf{x}_t \geq {d_l}^2 && \text{(1f)} \\
& \tilde{\mathbf{u}}^{(j)}_t \in \mathcal{O}^{(j)}(\mathbf{x}_t; \boldsymbol{\Upsilon}_{t+1}) && \text{(1g)} \\
& \mathbf{x}^{(j)}_{t+1} = \mathbf{h}(\mathbf{x}^{(j)}_t, \tilde{\mathbf{u}}^{(j)}_t) && \text{(1h)} \\
& \mathbf{w}_{t+1} = \mathbf{g}(\mathbf{x}^{(j)}_t; \boldsymbol{\Upsilon}_t) && \text{(1i)}
\end{aligned}$$

where all the constraints (1c)-(1h) are defined for each time step, $\forall t \in \mathbb{I}_{0:T-1}$, and the constraints (1g)-(1h) are defined for each human, $\forall j \in \mathbb{I}_{1:N}$. The robot collision constraint (1f) is defined for each human agent static obstacle in the environment. To solve this bilevel problem, we follow [21] to reformulate to a single level by replacing the lower level with its Karush-Kuhn-Tucker (KKT) optimality conditions. We propose a human trajectory forecaster that predicts future trajectories for multiple agents jointly. Our method extends a diffusion-based trajectory forecaster, MID [4], which generates predicted future motion samples for individual humans in the scene. The original model has two components: the encoder of Trajectron++ (T++) [1] and a Transformer that learns a reverse diffusion process [23]. During inference, MID requires the history of the human we intend to predict and their neighboring agents to be passed through the encoder to produce a latent embedding. The embedding is subsequently used to condition the Transformer model, which is responsible for de-noising, i.e. generating forecast samples for the individual human.

Our extension, Joint-MID or JMID, produces *joint* samples for *all* humans in the scene, i.e. each sample contains a forecast for all agents. To obtain joint samples, we query the encoder in parallel for each human in the scene to obtain a latent embedding for each human. We then concatenate these embeddings and pass them through the Transformer, which does not use positional encoding between the embeddings and is thus invariant to the concatenation ordering. To train the model, we follow [4] to maximize the variational lower bound. However, we alter the dimension of the loss inputs to account for the varying dimension of model outputs (i.e. varying number of humans being predicted) for each scene.

3.2 Filtering Prediction Samples to Satisfy Constraints

We propose to filter the JMID predictions to satisfy explicit constraints using a sequence of Optimal Reciprocal Collision Avoidance (ORCA) optimization problems [24]. Previous work [25, 21] has shown that in multiagent pedestrian scenarios, solving a series of ORCA optimization problems can produce accurate predictions of motion when given an estimate of the intended goal of each agent. For the sake of brevity, we provide a brief overview of the ORCA problem in App. A, more details can be found in Sec. III-B of [21].

We treat the solution as the filtered predicted human action at t, i.e. $\tilde{\mathbf{u}}^{(j)}_t = \mathcal{O}(\mathbf{x}_t; \boldsymbol{\Upsilon}^{(j)}_{t+1})$, and use the single-integrator dynamics model for the humans $\mathbf{x}^{(j)}_{t+1} = \mathbf{h}(\mathbf{x}^{(j)}_t, \tilde{\mathbf{u}}^{(j)}_t)$, to find the state of the agent at the next time step $t+1$. This way, the unconstrained optimum velocity for agent j is the one that moves the agent to the estimated intended position at the next time step, $\hat{\mathbf{p}}^{(j)}_{t+1}$. To find the unconstrained optimum, we first find the weighted average of the predicted positions of agent j at

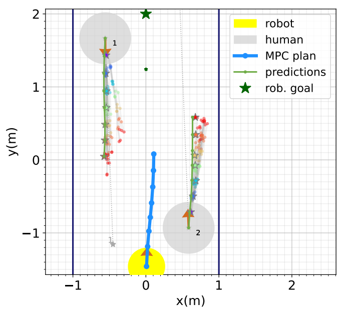


Figure 2: Example solution of our method. The diffusion model predicts $S = 20$ samples of the future trajectory for the two simulated human agents in the scene, illustrated as the multicolored scatter plots. Colors indicate the time step of the predicted intent, with violet for $t = 1$ and ending with red for $t = T$, in the order of the rainbow. The coupling between the forecasts of the different agents is not illustrated. The stars following the rainbow color-scheme indicate the intended position estimates for the respective time step.

time $t + 1$ from the JMID samples, $\boldsymbol{\Upsilon}_{t+1}^{(j)} = \left\{\check{\mathbf{p}}_{t+1}^{(j,s)}, \forall s \in \mathbb{I}_{1:S}\right\}$, using the importance weights $\mathbf{w}_t$,

$$\hat{\mathbf{p}}_{t+1}^{(j)} = \sum_{s=1}^{S} w_t^{(s)} \check{\mathbf{p}}_{t+1}^{(j,s)}. \tag{2}$$

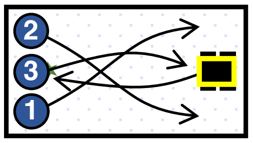
If the estimated intent is infeasible, the ORCA optimization problem will find the *closest feasible* position to the estimated intent.

To govern the evolution of weights through the MPC horizon in Eq. (1i), we leverage the fact that the ORCA filtering step shifts the predicted position to the closest feasible position to the estimated intent. Note that we describe a sequential process of calculations done at each time step, however, the sequence of steps that we describe is fully integrated into the MPC optimization problem (1). Thus, these sequential steps through the MPC horizon are implicitly encoded into the problem, rather explicitly sequentially calculated. At $t = 0$, we obtain the intended goal of each agent by weighing the JMID position samples for $t = 1$ equally, i.e. $w_0^{(s)} = \frac{1}{S} \forall s \in \mathbb{I}_{1:S}$, and averaging (see Eq. (2)). Then, we obtain the subsequent position of the agents $\forall j \in \mathbb{I}_{1:N}$, $\mathbf{p}_1^{(j)}$, via ORCA solutions (1g), and human dynamics (1h). To update the importance weights, we compare these ORCA-filtered positions, $\mathbf{p}_t^{(j)}$ with the predicted JMID samples positions, $\check{\mathbf{p}}_t^{(j,s)}$, for $t = 1$, $\tilde{w}_t^{(s)} = \exp\left(-\frac{1}{N\sigma}\sum_{j=1}^{N}\left\|\mathbf{p}_t^{(j)} - \check{\mathbf{p}}_t^{(j,s)}\right\|_2^2\right)$, 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 weight from the previous step, $\hat{w}_{t-1}^{(j,s)}$, and normalize once again, $w_t^{(s)} = w_{t-1}^{(s)} \bar{w}_t^{(s)} / \sum_{s=1}^{S} w_{t-1}^{(s)} \bar{w}_t^{(s)}$. Now with these updated weights, we use Eq. (2), to repeat the process all the way until $t = T - 1$.

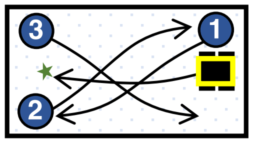
Fig. 2 illustrates an example of filtered human predictions and associated robot plan for a particular scene and JMID samples. In the case of agent 1, we can see that the second mode of the JMID predictions (to the right) would result in a collision with the robot if the robot follows its plan (blue line). Following our approach excludes the JMID samples associated with the the second mode for agent 1 from influencing the estimated intended trajectory of agent 1.

4 Closed-loop Real-Robot Experiments

Testing Environment: We evaluate the performance of our approach on a real robot in a controlled environment similar to [21]. We use a Clearpath Jackal differential drive robot pictured in Fig. 1. The robot weighs $20kg$ and measures $46cm$ W $\times 60cm$ L $\times$ $\times$ $50cm$ 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. 3 (i.e. six scenarios). For each run of a particular scenario, all 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. 3a) all

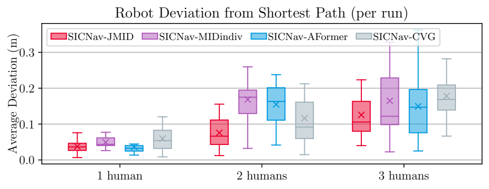


(a) Variant 1

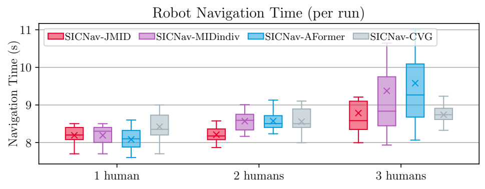


(b) Variant 2

Figure 3: 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 goal. In a scenario with $N \in \{1, 2, 3\}$ humans, positions greater than N are empty



(a) Robot deviation from shortest path to goal



(b) Robot navigation time to goal

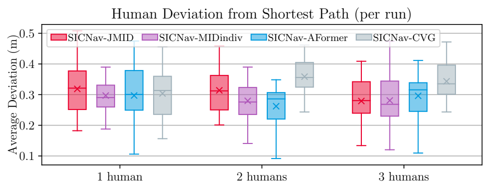
Figure 4: Box-and-whisker plots of robot navigation efficiency metrics (a) deviation from the optimal path to goal and (b) navigation time in scenarios with varying numbers of humans. Median values are illustrated by the line in the box-and-whisker plots and mean values are indicated by the $\times$. We observe that on average, using our method results in the most efficient robot navigation with respect to both metrics.

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. 3b). 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.

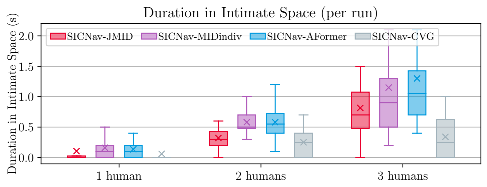
Navigation Methods: Each method uses a different approach to generate forecast samples of future human motion, then uses the Bilevel MPC method that we propose in Sec. 3 to simultaneously filter those predictions and optimize robot actions. The robot follows all methods in receding horizon fashion. The four navigation methods tested are as following, SICNav-JMID: our proposed method that uses the joint prediction JMID samples (i.e. each forecast sample contains future positions for all humans) with SICNav, SICNav-iMID: a baseline that uses individual agent samples from MID [4] with SICNav, SICNav-AF: joint prediction samples from the AgentFormer [3] model along with SICNav, and SICNav-CVG [21], a simple projection of the human's current velocity forward in time to infer an intended travel direction for the human to use with SICNav. This is a simple baseline method that does not use any learning-based trajectory forecasting. We present a validation of the open-loop prediction performance of the JMID, iMID, and AgentFormer models in App. B.

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 under analysis on an 24-core Lenovo Legion laptop using Intel$^{(R)}$ Core$^{(TM)}$ i9-14900HX @ 2.2 GHz CPUs and an NVIDIA$^{(R)}$ GeForce RTX 4080 GPU. The prediction models JMID, MID, and AgentFormer are implemented in PyTorch and wrapped with ROS to deploy inference on the robot at 10 Hz. We train the models using the same methodology as Sec. B except with the JRDB dataset [26], which also contains a robot in the scene. We also fine-tune the models using data collected from our robot. We implement a Python ROS node to run the controller in real-time at 10 Hz with CasADi as the modelling language and the Acados [27] solver for optimization. The solver generates C++ code from the CasADi models and uses Just-in-time Compilation to run in real-time. 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

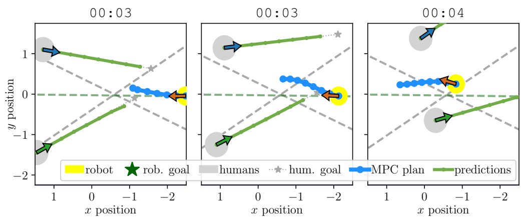


(a) Humans' deviation from their shortest paths to goal

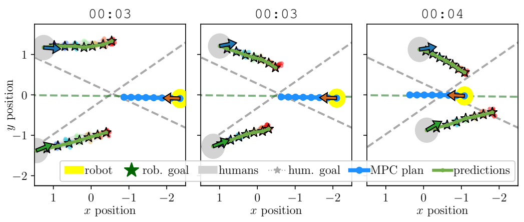


(b) Duration of time robot spends in personal space of a human

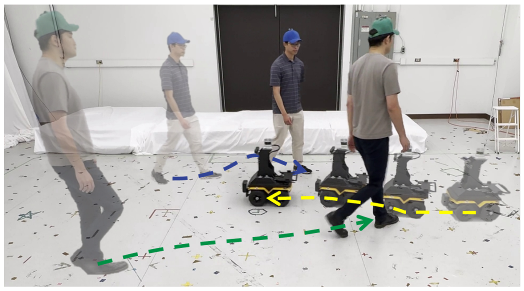
Figure 5: Box-and-whisker plots of (a) total duration of time robot spends in personal space of a human and (b) the humans' deviation from their shortest paths to goal scenarios with varying numbers of humans. Median values are illustrated by the line in the box-and-whisker plots and mean values are indicated by the $\times$. We observe that on average, using our method results in the most efficient robot navigation with respect to both metrics.



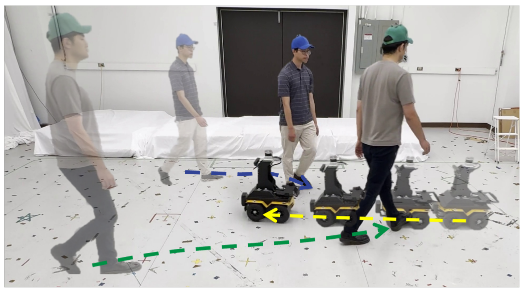
(a) Top view snapshots of SICNav-CVG robot



(b) Top view snapshots of SICNav-JMID robot



(c) SICNav-CVG robot turns towards blue-hat human. (vid in supp.)



(d) SICNav-JMID robot does not turn towards blue-hat human. (vid in supp.)

Figure 6: We illustrate two similar scenarios with SICNav-CVG (left) and SICNav-JMID (right) in the first 3-4 s of the scenarios. The top images (a, b) show the top-down snapshot graphs of the scenarios with dashed lines (gray for humans, green for robot) indicating each agent's optimal path. The bottom images (c, d) show the video of each scenario, with dashed arrows indicating paths traversed by each agent. Transparent images illustrate older frames of the video. The SICNav-CVG robot produces solutions that turn towards the blue-hat human, causing both the robot and the human to deviate from their optimal paths, while the JMID predictions allow SICNav-JMID to produce solutions where all agents stay closer to their shortest-paths-to-goal.

goal in each run). Fig. 4 shows the box-and-whisker plots of the robot navigation efficiency metrics for the different navigation methods. We observe that 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 differences in the interaction 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. 5a shows the box-and-whisker plots of the humans' deviation from their shortest paths to the goal for the different navigation methods. We observe that looking at all three scenarios, the learning-based methods SICNav-JMID, SICNav-IMID, and SICNav-AF perform similarly in terms of this metric. However, in the scenarios with $N \in \{2, 3\}$ we can see that SICNav-CVG is an outlier: causing a higher deviation of the humans from their shortest paths to the goal compared to the learning-based methods.

Second, to analyze proxemics, we look at the duration of time the robot spends within the *intimate space* [28], 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. In the scnarios with $N = 1$, we can see that all methods spend almost no time in the personal space of the human. In the scenarios with $N \in \{2, 3\}$, we observe that using SICNav-JMID spends less time in the personal 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. 5a, we can 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 to the goal.

We interpret this result as meaning that SICNav-CVG causes the humans to *run away* or *avoid* getting close to the robot. 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, 6c) and SICNav-JMID (6b, 6d). In the first snapshots for both methods, we observe the blue-hat agent's heading (blue arrow) deviates more from their optimal path than the green-hat agent's heading. With CVG predictions (6a, 6c), 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, 6d) predictions, at the second control step (middle graph in the snapshots), the JMID model predicts that the blue-hat agent intends to return toward their optimal path (gray dashed line), 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 agents to stay close to their optimal paths.

5 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 filter 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 that it outperforms state-of-the-art methods in terms of joint forecasting. We also conduct real-robot experiments to evaluate the performance of our method in a controlled environment. Our results show that our method outperforms state-of-the-art methods in terms of robot navigation efficiency and safety.

References

[1] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone. Trajectron++: Dynamically-Feasible Trajectory Forecasting With Heterogeneous Data. In *2020 European Conference on Computer Vision (ECCV)*, 2020. URL `http://arxiv.org/abs/2001.03093`.

[2] J. Yue, D. Manocha, and H. Wang. Human Trajectory Prediction via Neural Social Physics. In *Proceedings of the European Conference on Computer Vision (ECCV)*. arXiv, July 2022. URL `http://arxiv.org/abs/2207.10435`. arXiv:2207.10435 [cs].

[3] Y. Yuan, X. Weng, Y. Ou, and K. Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[4] T. Gu, G. Chen, J. Li, C. Lin, Y. Rao, J. Zhou, and J. Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *2022 IEEE/CVF Conference on Computer Vision and Pat-*

tern Recognition (CVPR), pages 17092–17101, 2022. doi:10.1109/CVPR52688.2022.01660. URL doi.org/10.1109/CVPR52688.2022.01660.

[5] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, Mar. 1997. ISSN 10709932. doi:10.1109/100.580977. URL doi.org/10.1109/100.580977.

[6] M. Khatib, H. Jaouni, R. Chatila, and J. Laumond. Dynamic path modification for car-like nonholonomic mobile robots. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 2920–2925, Albuquerque, NM, USA, 1997. IEEE. ISBN 978-0-7803-3612-4. doi:10.1109/ROBOT.1997.606730. URL http://ieeexplore.ieee.org/document/606730/.

[7] N. E. Du Toit and J. W. Burdick. Robot Motion Planning in Dynamic, Uncertain Environments. *IEEE Transactions on Robotics*, 28(1):101–115, Feb. 2012. ISSN 1552-3098, 1941-0468. doi:10.1109/TRO.2011.2166435. URL doi.org/10.1109/TRO.2011.2166435.

[8] C. Rösmann, F. Hoffmann, and T. Bertram. Integrated online trajectory planning and optimization in distinctive topologies. *Robotics and Autonomous Systems*, 88:142–153, Feb. 2017. ISSN 09218890. doi:10.1016/j.robot.2016.11.007. URL https://linkinghub.elsevier.com/retrieve/pii/S0921889016300495.

[9] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, June 2000. ISSN 00051098. doi:10.1016/S0005-1098(99)00214-9. URL https://linkinghub.elsevier.com/retrieve/pii/S0005109899002149.

[10] P. Trautman and A. Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 797–803. IEEE, 2010. ISBN 978-1-4244-6675-7. doi:10.1109/IROS.2010.5654369. URL https://doi.org/10.1109/IROS.2010.5654369.

[11] 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 (IROS)*, Detroit, MI, USA, Mar. 2023. URL http://arxiv.org/abs/2303.01424.

[12] 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 (ICRA)*, pages 6015–6022, 2019. ISBN 9781538660263. doi:10.1109/ICRA.2019.8794134. URL doi.org/10.1109/ICRA.2019.8794134.

[13] 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 (IROS)*, pages 10007–10013, Las Vegas, NV, USA, Oct. 2020. IEEE. ISBN 978-1-72816-212-6. doi:10.1109/IROS45743.2020.9340705. URL https://ieeexplore.ieee.org/document/9340705/.

[14] M. Everett, Y. F. Chen, and J. P. How. Collision avoidance in pedestrian-rich environments with deep reinforcement learning. *IEEE Access*, 9:10357–10377, 2021. ISSN 21693536. doi:10.1109/ACCESS.2021.3050338.

[15] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick. End-to-End Safe Reinforcement Learning through Barrier Functions for Safety-Critical Continuous Control Tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3387–3395, July 2019. ISSN 2374-3468, 2159-5399. doi:10.1609/aaai.v33i01.33013387. URL https://ojs.aaai.org/index.php/AAAI/article/view/4213.

[16] 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. *International Journal of Robotics Research*, 34(3):335–356, 2015. ISSN 17413176. doi:10.1177/0278364914557874. URL doi.org/10.1177/0278364914557874.

[17] M. Sun, F. Baldini, P. Trautman, and T. Murphey. Move Beyond Trajectories: Distribution Space Coupling for Crowd Navigation. In *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, July 2021. ISBN 978-0-9923747-7-8. doi:10.15607/RSS.2021.XVII.053. URL doi.org/10.15607/RSS.2021.XVII.053.

[18] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for Autonomous Cars that Leverage Effects on Human Actions. In *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation, 2016. ISBN 978-0-9923747-2-3. doi:10.15607/RSS.2016.XII.029.

[19] A. Lerner, Y. Chrysanthou, and D. Lischinski. Crowds by Example. *Computer Graphics Forum*, 26(3):655–664, Sept. 2007. ISSN 0167-7055, 1467-8659. doi:10.1111/j.1467-8659.2007.01089.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2007.01089.x.

[20] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268, Kyoto, Sept. 2009. IEEE. ISBN 978-1-4244-4420-5. doi:10.1109/ICCV.2009.5459260. URL http://ieeexplore.ieee.org/document/5459260/.

[21] S. Samavi, F. Shkurti, and A. P. Schoellig. Sicnav: Safe and interactive crowd navigation using model predictive control and bilevel optimization, 2023. URL https://arxiv.org/abs/2310.10982.

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

[23] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[24] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha. Reciprocal n-Body Collision Avoidance. In B. Siciliano, O. Khatib, F. Groen, C. Pradalier, R. Siegwart, and G. Hirzinger, editors, *Robotics Research*, volume 70, pages 3–19. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-19456-6 978-3-642-19457-3. doi:10.1007/978-3-642-19457-3_1. URL http://link.springer.com/10.1007/978-3-642-19457-3_1. Series Title: Springer Tracts in Advanced Robotics.

[25] 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*, 52(4):2289–2301, Apr. 2022. ISSN 2168-2216, 2168-2232. doi:10.1109/TSMC.2020.3048964. URL https://ieeexplore.ieee.org/document/9328889/.

[26] 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.

[27] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl. acados – a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation*, Oct 2021. ISSN 1867-2957. doi:10.1007/s12532-021-00208-8. URL https://doi.org/10.1007/s12532-021-00208-8.

[28] E. T. Hall. *The hidden dimension*. Anchor Books, New York, 1990. ISBN 978-0-385-08476-5.

Table 1: Trajectory Forecasting Results on the ETH/UCY Benchmark. Bold is best. Underline is second-best.

Method	ETH		Hotel		Univ		Zara1		Zara2		Average	
	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓	ADE (m) ↓ / FDE (m) ↓	SADE (m) ↓ / SFDE (m) ↓
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**

[29] C. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, D. Anguelov, et al. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9644–9653, 2023.

[30] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

[31] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun. Implicit latent variable model for scene-consistent motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 624–641. Springer, 2020.

[32] E. Weng, H. Hoshino, D. Ramanan, and K. Kitani. Joint metrics matter: A better standard for trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20315–20326, 2023.

A ORCA description

Solving ORCA finds the following set of optimal (i.e. minimum-cost and feasible) velocities for each agent j in a scene of N agents at each time step t,

Definition 1 (ORCA) *The* set of velocities solving the ORCA problem for agent j *is,*

$$\mathcal{O}(\mathbf{x}_t; \boldsymbol{\Upsilon}_{t+1}^{(j)}) := \underset{\mathbf{v},\zeta}{\arg\min} \left\| \mathbf{v} - \mathbf{v}_{\textit{pref}}(\mathbf{x}_t, \boldsymbol{\Upsilon}_{t+1}^{(j)}) \right\|_2^2 + M\zeta^2 \;\textbf{\textit{s.t.}}\; \begin{array}{c} \mathbf{n}^{(j|l)}(\mathbf{x}_t)^\top \mathbf{v} \geq \rho^{(j|l)}(\mathbf{x}_t) - \zeta \\ \mathbf{n}^{(j|\tilde{l})}(\mathbf{x}_t)^\top \mathbf{v} \geq \rho^{(j|\tilde{l})}(\mathbf{x}_t) \\ \left\|\mathbf{v}\right\|_2^2 \leq {v_{\max}}^2, \zeta \geq 0 \end{array} \tag{3}$$

where the optimization variable $\mathbf{v} \in \mathbb{R}^2$ is the velocity of agent j to be executed for one timestep, and the cost is based on a preferred velocity $\mathbf{v}_{\text{pref}} : \mathcal{X} \to \mathbb{R}^2$, a vector toward the agent's intended position at the next time step (to be obtained from the JMID samples). The maximum velocity of the agent is constrained by $v_{\max}$, and the vectors $\mathbf{n}^{(j|l)} : \mathcal{X} \to \mathbb{R}^2$, which are piece-wise linear functions of $\mathbf{x}_t$, and scalar $\rho^{(j|l)} : \mathcal{X} \to \mathbb{R}^2$ which are nonlinear functions of $\mathbf{x}_t$, are derived using a velocity obstacle approach (see Sec. 4 of [24]) and define linear collision avoidance constraints between agents j and agent $l, \forall l \in \{r, 1, \ldots, N\} \setminus \{j\}$, as well as between agent j and each static obstacle, indexed $\forall \tilde{l} \in \{N+1, \ldots, N+M\}$.

The cost function and constraint functions in (3) depend on the positions and velocities of all agents in the system at each time step meaning that the state of the system at time t, $\mathbf{x}_t$, *parameterizes* the problem. The cost function is additionally parameterized by the set of predictions obtained from the JMID model for agent j, $\boldsymbol{\Upsilon}_t^{(j)}$, as well as the importance weights $\mathbf{w}_t$ which are part of the state.

B Open-loop Prediction Experiments

Before using the prediction model on a robot in closed-loop, we validate open-loop performance on the popular ETH/UCY benchmark. Our evaluation follows the standard protocol of leave-one-scene-out cross-validation, with the standard 3.2s history and 4.8s forecast.

Models: We compare the following methods, JMID: our proposed method that generates joint predictions (i.e. each forecast sample contains future positions for all humans); IMID [4]: the original MID model that JMID extends. This baseline provides an ablation study on whether jointly modeling trajectory forecasts as with JMID actually results in better scene-level performance; AgentFormer [3]: a state-of-the-art method for joint human trajectory forecasting using an autoregressive Transformer architecture in contrast to the diffusion-based architecture that we use in JMID; T++ [1]: a Conditional Variational Autoencoder (CVAE)-based method on which the encoders of Individual Motion Indeterminacy Diffusion (IMID) and our JMID methods is based; To do inference with the diffusion models, IMID and JMID, we use a Denoising Diffusion Probabilistic Model (DDPM) version of the models, where denoising is conducted in 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), where denoising can be conducted with a variable number of steps, 2 in our case.

Metrics: Following existing literature [1, 3, 29, 30, 31, 32], we evaluate predictive performance using Best-of-20 Average Displacement Error (ADE): the lowest Euclidean distance between the ground truth of each agent and 20 samples obtained from the model, for the respective agent; Best-of-20 Final Displacement Error (FDE): the lowest Euclidean distance between positions at the final prediction step, T, of the ground truth of each agent and 20 samples obtained from the model, for the respective agent; Best-of-20 Scene-level Average Displacement Error (SADE): the lowest average ADE across all agents in the scene for 20 joint-prediction samples; Best-of-20 Scene-level Final Displacement Error (SFDE): 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' future). 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 [4]. We trained our model using the Adam optimizer with a learning rate of 10^{-4} and a batch size of 64.

Results and Discussion: Table 1 summarizes the results of our experiments. On average our method, JMID outperforms the baselines in the scene-level metrics on average and on most test-splits. However, AgentFormer performs the best in the individual-level metrics across all of the test splits. As discussed previously, the scene-level metrics are more meaningful than the individual-level metrics because they evaluate how well a model has learned interactions between pedestrians. Therefore, we believe our proposed method performs the best in the most meaningful metrics.