Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Move Beyond Trajectories: Distribution Space Coupling for Crowd Navigation

Muchen Sun*, Francesca Baldini†‡ Peter Trautman†, and Todd Murphey*
*Department of Mechanical Engineering, Northwestern University, Evanston, IL 60208, USA
†Honda Research Institute, San Jose, CA 95134, USA
‡California Institute of Technology, Pasadena, CA 91125, USA

***Abstract*—Cooperatively avoiding collision is a critical functionality for robots navigating in dense human crowds, failure of which could lead to either overaggressive or overcautious behavior. A necessary condition for cooperative collision avoidance is to couple the prediction of the agents' trajectories with the planning of the robot's trajectory. However, it is unclear that *trajectory* based cooperative collision avoidance captures the correct agent attributes. In this work we migrate from trajectory based coupling to a formalism that couples agent preference *distributions*. In particular, we show that preference distributions (probability density functions representing agents' intentions) can capture higher order statistics of agent behaviors, such as willingness to cooperate. Thus, coupling in distribution space exploits more information about inter-agent cooperation than coupling in trajectory space. We thus introduce a general objective for coupled prediction and planning in distribution space, and propose an iterative best response optimization method based on variational analysis with guaranteed sufficient decrease. Based on this analysis, we develop a sampling-based motion planning framework called *DistNav*[1] that runs in real time on a laptop CPU. We evaluate our approach on challenging scenarios from both real world datasets and simulation environments, and benchmark against a wide variety of model based and machine learning based approaches. The safety and efficiency statistics of our approach outperform all other models. Finally, we find that DistNav is competitive with *human* safety and efficiency performance.**

I. Introduction

Collision avoidance in dense human crowds is a challenging problem. Whereas conventional motion planning algorithms work well with slowly moving obstacles and low obstacle density, they are designed to work with passive obstacles that will not react to the robot, and assume the robot has full knowledge about the obstacles' future states. Such assumptions, however, no longer hold in human crowds and could lead to either overcautious or overaggressive robot behaviors, which is known as the freezing robot problem [44]. Previous work on crowd navigation reveals that one necessary condition to avoid freezing robot behavior is to couple the prediction of the agent's future trajectories with robot planning (e.g., accounting for the robot's influence on agent behavior).

However, pure trajectory-based models make strong implicit assumptions about the distribution governing interactive agent behaviors (what we call the agent's "preference distribution"). For example, the *extent* to which an agent is willing to move out of the way of the robot is typically left unmodeled. Further, trajectory space approaches assume that the (implicitly modeled) preference distribution is a static quantity: *during* interaction, the preference distribution does not change. As an example, consider unimodal Gaussian agent models, where the covariance can be interpreted as the extent to which an agent will make room for another agent. Critically, as two agents proceed past each other, their covariance will change. If an agent moves far out of the way of the path of the robot, its covariance will become more peaked, indicating that the agent is only willing to make so much room. For non-Gaussian agent distributions, higher order factors such as skew (e.g., preference for left versus right passage) and multi-modality (e.g., ambiguity about which side an agent might pass on), are themselves coupled to the deformation of the agent trajectories. In short, assuming that interaction occurs *only* at the trajectory level ignores critical information about how

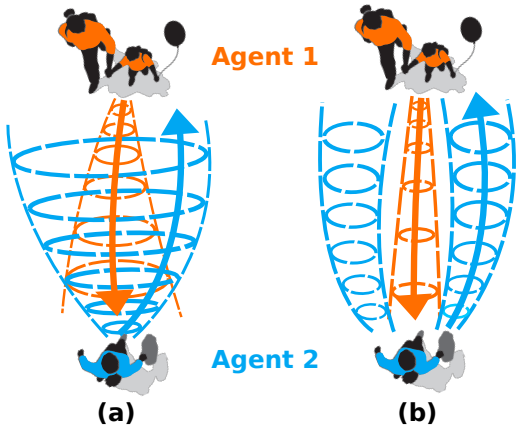

Fig. 1: Difference between modeling humans as specific trajectories and distributions: Agent 2 at the bottom (pictured as a human but also can be a robot) needs to plan a path while predicting how the other two pedestrians, who are a family and thus considered as a single agent 1, would react to agent 2's decision. The dashed circles represent each agent's probability distribution (preference) at each time step, here we assume agent 1 has a lower flexibility and thus agent 1 would expect agent 2 to cooperate more to make space for each other. **(a) By modeling humans as *trajectories*, agent 2 implicitly assumes the preferences of both agents are fixed in the presence of interaction, which is not necessarily true. (b) Modeling humans as *distributions* can overcome this issue, where agent 2's intention evolves to a bi-modal distribution because of the potential interaction, and the trajectory with maximum likelihood (go right) would eventually be chosen as the plan.** While both methods can recover a proper path for agent 2 to avoid splitting up the family, **explicitly modeling the evolution of preferences captures more information about the interaction.**

[1]For more details please visit https://sites.google.com/view/distnav/

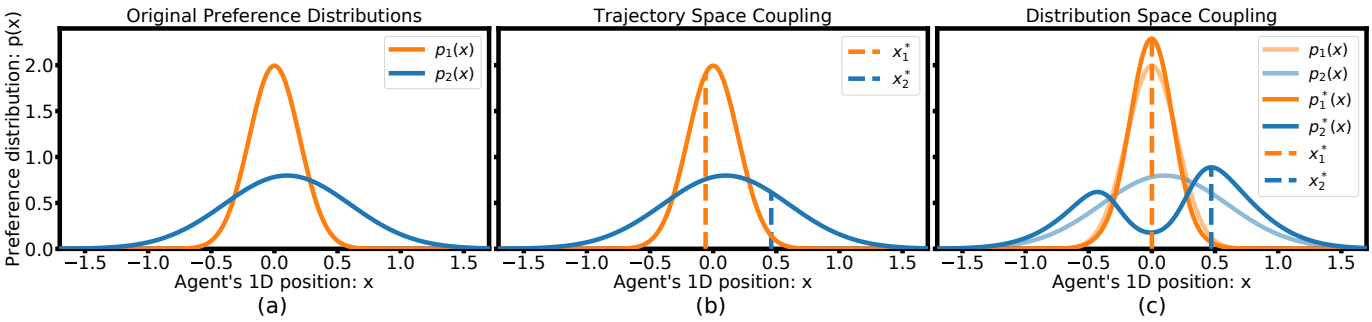

Fig. 2: Simplified 1D example showing two agents bypassing each other, in this example each agent only considers its 1D location at next one step. (a) Original preference distributions of the two agents, modeled as two Gaussians. (b) Coupled prediction and planning in trajectory space, where agent 2 finds its optimal plan (blue dashed line on the right) and optimal prediction for agent 1's plan (orange dashed line near the center) simultaneously. Again, agent 2 implicitly assumes two agents' preferences remain static in presence of interaction. (c) **Our method relaxes this assumption, and allows preference distributions to evolve in a non-parametric manner**. Agent 2's intention can be explicitly modeled as a bi-modal distribution, it can both go right or go left, but going right is more preferred. Optimal plan (blue dashed line) and prediction (orange dashed line) are then picked by agent 2 from its belief for updated preferences.

interaction deforms higher order factors of the agent model (e.g., how interaction deforms the preference distribution). Figure 1 illustrates the necessity of modeling agent preference evolution through a high-level example that is common in crowd navigation—two agents passing each other on the street.

We relax this assumption of a known and fixed preference distribution. Specifically, we do not require the agent's preference distribution to remain static during interaction, but instead allow the preference distribution to evolve (Figure 2 demonstrates the preference evolution in a 1D example). We accomplish this by:

1) Taking the atomic representational unit to be a probability density function over agent trajectories, in what we call *distribution space*.
2) Including both human-robot and human-human interaction.
3) Developing a method to jointly optimize over both robot and agent preference distributions.

In this work distributions represent agents' preferences over trajectories—including both what they want and their potentially multi-modal degree of flexibility. This use of distributions is distinct from that found in other computational methods that use distributions are their atomic unit, such as belief space planning, where the purpose of a distribution is to represent uncertainty (for example, [30] uses belief space planning for crowd navigation but decouples prediction and planning). Here we develop a formalism and algorithm to "shape" probability density functions for better cooperation, even for systems with no uncertainty present.

At last, we summarize our contributions as follow:

1) We formulate a general objective for coupled prediction and planning in distribution space (Section III).
2) We propose an iterative best response optimization algorithm which is guaranteed to decrease the objective in each iteration, and design a sampling-based navigation framework called DistNav (Section IV).
3) We conduct a comprehensive evaluation using both real world datasets and simulation environments, and benchmark DistNav against a variety of other crowd navigation methods. Results show that DistNav outperforms other methods in both safety and efficiency metrics, and is competitive with *human* safety and efficiency performance (Section V).

II. Related Work

Roboticists have been investigating navigation in human environments since the 1990s. Two landmark studies were the RHINO [2] and MINERVA [41] experiments. In [8] the authors observed that assuming agent independence leads to an uncertainty explosion that makes efficient navigation impossible. In [44], it was shown that bounding uncertainty (such as in [40, 8, 21, 33]) cannot prevent freezing robot behavior. Relatedly, *Human intention aware* planning is a popular crowd navigation approach [23]; examples include [17, 28, 38, 34, 45]. Although these approaches model human-robot interaction, they ignore human-robot *cooperation*.

Some approaches learn navigation strategies by observing examples, such as through inverse reinforcement learning [22, 50, 51] or deep reinforcement learning [6, 10, 39]. Typically, human relationship models are ignored; importantly, [4] models human-human interaction in a method called "socially aware reinforcement learning (SARL)." In [5], agent relational graphs are trained using deep reinforcement learning. In [27], a relational graph is paired with "negative examples" (e.g., collisions) to enforce safety constraints and social norms. These reinforcement learning methods rely on simulators for training, but as being pointed out in [12], current simulators make unrealistic but critical assumptions that will not hold in real world, which may affect the sim-to-real transfer.

Coupled prediction and planning approaches explicitly capture the mutual dependencies between human and robot. An important body of work is game theoretic planning[13, 7], these planners typically assume agent objectives are known, but recent works introduce online estimation of human agents' objectives [36, 25]. Further, [36] uses the social value orientation (SVO) to quantify the degree of agents' selfishness or altruism. Similar idea of modeling human willingness to coordinate can be found in other works. For example, [29] models mutual adaptation between human and robot in manipulation

tasks, and [43] models human flexibility as the covariance matrix of a Gaussian process. In addition to online estimation, [35] introduces an active information gathering algorithm that generates communicative behaviors for autonomous vehicles. However, in these works the planners still over-confidently predict human behavior as a single trajectory, and none of them capture how willingness to coordinate changes *during* interaction.

Lastly, while the deep learning approaches in [32, 1, 16, 48, 9] focus on *prediction*, they are an important contribution to crowd navigation. In [20], variational auto encoders capture multimodality. [3] includes human "intent uncertainty" and "control uncertainty" as part of the prediction, and model them with Gaussian mixture models. But none of these models explicitly measure or account for how flexibility changes during interaction.

III. Problem Formulation

A. Terminology

Consider there are $n+1$ agents including n pedestrians and one robot in the environment, we start by defining a set of unique indices $\mathcal{I} = \{R, 1, 2, \dots, n\}$ for all agents, where R is the index of the robot and is treated as *zero* when compared with other indices. The state of each agent is in the space $\mathcal{X} \subseteq \mathbb{R}^k$, for example if we only consider all agents' planar positions, then $\mathcal{X} \subseteq \mathbb{R}^2$. The trajectory of each agent $f^{(i)} : \mathbb{R}^+ \mapsto \mathcal{X}, i \in \mathcal{I}$ is a set function that maps a set of T time points to the agent state at that moment. In practice, the trajectory f would be evaluated as a vector or 2D matrix, thus dimension of the *trajectory space* is $f \in \mathcal{F} \subseteq \mathbb{R}^{k \times T}$. Agent states are measured through a measurement model $z_t^{(i)} = h(f^{(i)}(t))$, and we collect measurements of the agents' past states at time steps $\{1, 2, \dots, t\}$ as $z_{1:t}^{(i)} = [z_1^{(i)}, z_2^{(i)}, \dots, z_t^{(i)}]$, note that the observations could be noisy (in this paper we assume additive zero-mean Gaussian noises) and we don't assume $z_{1:t}^i$ to be complete: z_τ^i could be missing for some $\tau \in [1, t]$ and $i \in \mathcal{I}$.

Definition 1 (Distribution space)**.** *The distribution space $\mathcal{P}$ is a function space, where each element $p(f) : \mathcal{F} \mapsto \mathbb{R}_0^+$ is a probability density function that maps the trajectory space $\mathcal{F}$ to the non-negative real domain, and each element satisfies:*

$$\int_{\mathcal{F}} p(f) df = 1 \tag{III.1}$$

Definition 2 (Original preference distribution)**.** *The prior probability for agent i's trajectory conditioned on its measurements $z_{1:t}^{(i)}$ is defined as the agent's original preference distribution $p_i(f) = p(f^{(i)} = f | z_{1:t}^{(i)}) \in \mathcal{P}$. Since no measurement of other agents is used, original preference distribution doesn't include the interaction with other agents.*

In this paper we use Gaussian processes regression to compute the original preferences, so in the rest of the paper we assume the original preferences are GPs, but our results and algorithm apply to *arbitrary distributions*. We refer the readers to [33] for more details about GP regression.

Preference distribution represents the agent's intention, which contains information for agent's preferred trajectories (intents) and their willingness to give up the preferred trajectories in order to cooperate with other agents (flexibility). Below we define *intent* and *flexibility* for Gaussian and arbitrary preferences.

Definition 3 (Intent and flexibility for Gaussian preferences)**.** *When the agent's preference is a Gaussian process (GP) $p_i(f) = \mathcal{N}(f | \mu_i, \Sigma_i)$, the agent's intent and flexibility are defined as the GP mean μ_i and covariance Σ_i, respectively.*

Definition 4 (Intent and flexibility for arbitrary distributions)**.** *More generally, for any distribution $p(f)$, the intents are defined as the local maximums of $p(f)$ (so there could be multiple intents) and the flexibility is qualitatively measured through the covariance, skew and kurtosis of the distribution.*

Measuring flexibility for arbitrary distribution is tricky, in this paper we only consider qualitative analysis: Covariance represents the "spread" of the distribution, a larger covariance indicates a larger feasible action region. Skew measures the symmetry of the preference, for example whether the agent is more willing to go "left" or go "right". Kurtosis is the "tailedness" of the distribution, it can be interpreted as the agent's tolerance for large deviations from the intents.

At last, we also need to define a function to penalize the likelihood of collision between two trajectories.

Definition 5 (Collision penalty function)**.** *The collision penalty function $\psi(f^{(i)}, f^{(j)}) : \mathcal{F} \mapsto \mathbb{R}^+$ represents the likelihood of collision between two trajectories, and it needs to be symmetric such that $\psi(f^{(i)}, f^{(j)}) = \psi(f^{(j)}, f^{(i)})$.*

As an example, in our implementation we choose the collision penalty function to be:

$$\psi(f^{(i)}, f^{(j)}) = \max_t w \cdot \mathcal{N}(f^{(i)}(t) | f^{(j)}(t), \Sigma_\psi) \tag{III.2}$$

where $w \in \mathbb{R}$ is the penalty weight and $\Sigma_\psi \in \mathbb{R}^{k \times k}$ controls how close two agents can be.

B. Coupled Prediction and Planning in Distribution Space

Coupled prediction and planning (also called generative navigation) considers motion planning as a prediction problem; when applied to crowd navigation, the robot couples the prediction of pedestrian trajectories and planning for its own trajectory by optimizing the joint trajectories of all agents simultaneously. The navigation goal and waypoints for the robot can be included as (artificial) observations, so the predicted robot trajectory will pass through them. We refer the readers to [44] for more details about incorporating navigation task into prediction.

We start formulating coupled prediction and planning in distribution space by extending the collision penalty function to distribution space.

Definition 6 (Expected collision penalty)**.** *The expected collision penalty $c(p_i, p_j) : \mathcal{P} \mapsto \mathbb{R}_0^+$ is defined as the joint*

expected value of the collision penalty function $\psi(f^{(i)}, f^{(j)})$ *with respect to two preference distributions:*

$$c(p_i, p_j) = \int_{\mathcal{F}} \int_{\mathcal{F}} \psi(f^{(i)}, f^{(j)}) p_i(f^{(i)}) p_j(f^{(j)}) df^{(i)} df^{(j)} \tag{III.3}$$

Definition 7 (Joint expected collision penalty)**.** *For* n *agents, the joint expected collision penalty is defined as:*

$$J_c(p_R, p_1, \ldots, p_n) = \sum_{i=R}^{n} \sum_{j=i+1}^{n} c(p_i, p_j) \tag{III.4}$$

Definition 8 (Coupled Prediction and Planning in Distribution Space)**.** *Given the original preference distributions for all agents* $p_R(f), p_1(f), \ldots, p_n(f)$*, and the expected collision penalty function* $c(p_i, p_j)$*, the target is to find optimal preference distributions* $(p_R, p_1, \ldots, p_n)^*$ *that minimize the joint expected collision penalty (III.4), and also prevent large deviations from the original preference distributions. Then the optimal joint* trajectories *are selected individually as the maximum of each agent's optimal preference distribution.*

One point worth emphasizing in the above definition is that we do not put strict constraints on the difference between current preference distributions and original preferences, but instead consider it as a "soft constraint" while giving the joint expected collision penalty (III.4) a higher priority as the optimization objective. This is because the original preference distribution can lead to dangerous conflicts since they don't include other agents' intentions, in which case strictly constraining the deviation from such original preferences limits the agents' cooperation for safer joint actions. As will be discussed in the next section, in our proposed algorithm, we only constrain the preference deviation between two consecutive iterations during the optimization, rather than constraining them with respect to the original preferences; this relaxation gives more freedom for conflict resolution.

We consider crowd navigation as a *receding horizon* planning problem, therefore solving for optimal preferences distributions and selecting optimal coupled joint trajectories over one time step. When new observations are obtained, new preference distributions are generated and the joint trajectories for coupled planning and prediction are updated.

IV. Variational Analysis for Coupled Prediction and Planning in Distribution Space

The formulation of coupled prediction and planning in distribution space raises several challenges:

1) The objective (III.4) is a functional, which is defined in the infinite-dimensional function space, so traditional optimization methods in vector space may not apply.
2) The objective is subject to subsidiary constraints (III.1).
3) The objective contains a combinatorial dependencies between the variables, in other words, all agents update their preferences in response to how others update. This structure makes the objective challenging to optimize.

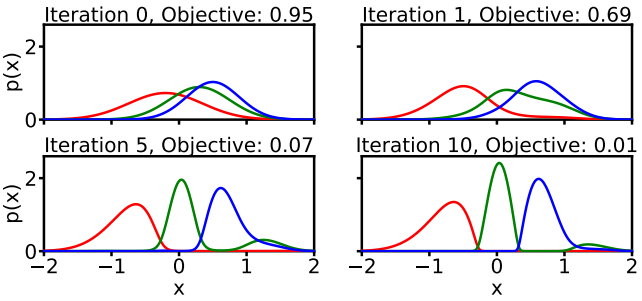

Fig. 3: Evolution of three 1D preference distributions in 10 iterations using the sequential variational update. All three distribution are initialized as Gaussian distributions, the collision penalty function is a Gaussian probability density function with small variance. We can clearly see the change of intents and flexibilities (variance, skew and kurtosis) during the iterations.

To address the above challenges, in this paper we combine the following techniques:

1) For the inter-agent dependencies, a strategy commonly used in multi-agent game solvers is the iterative best response (IBR) scheme [49][37]. In one iteration, each agent's strategy (in our case the preference distribution) is locally optimized by solving a subproblem; in the subproblem other agents' strategies are fixed.
2) The subproblem for each agent is constructed sequentially. Each agent updated their preference distribution in response to those who have already updated, and assuming those who haven't updated would follow.
3) Each subproblem can be solved analytically using Lagrange multipliers [14] as an isoperimetric problem with subsidiary conditions.

A. Sequential Iterative Variational Update

In k-th iteration, each agent updates its own preference sequentially by solving a subproblem. For agent i, the corresponding subproblem is:

$$p_i^{(k+1)}(f) = \arg\min_{p} \left\{ D_{KL}(p \| p_i^{(k)}) + \bar{c}_i^{(k)}(p) \right\} \tag{IV.1}$$

where

$$\bar{c}_i^{(k)}(p) = \sum_{j=R}^{i-1} c(p, p_j^{(k+1)}) + \sum_{j=i+1}^{n} c(p, p_j^{(k)}) \tag{IV.2}$$

$$= \int_{\mathcal{F}} p(f) \bar{\gamma}_i^{(k)}(f) df \tag{IV.3}$$

$$\bar{\gamma}_i^{(k)}(f) = \sum_{j=R}^{i-1} \int_{\mathcal{F}} \psi(f, f^{(j)}) p_j^{(k+1)}(f^{(j)}) df^{(j)} \tag{IV.4}$$

$$+ \sum_{j=i+1}^{n} \int_{\mathcal{F}} \psi(f, f^{(j)}) p_j^{(k)}(f^{(j)}) df^{(j)} \tag{IV.5}$$

The first term in the subproblem objective (IV.1) is the Kullback-Leibler divergence between the preferences in two iterations, which controls the change between preferences in two consecutive iterations, and therefore serves as a "soft constraint" on the deviation of current preference $p_i^{(k)}(f)$ from the original preference $p_i^{(0)}(f)$. The second term $\bar{c}_i^{(k)}(p)$ measures the summation of the expected collision penalties between

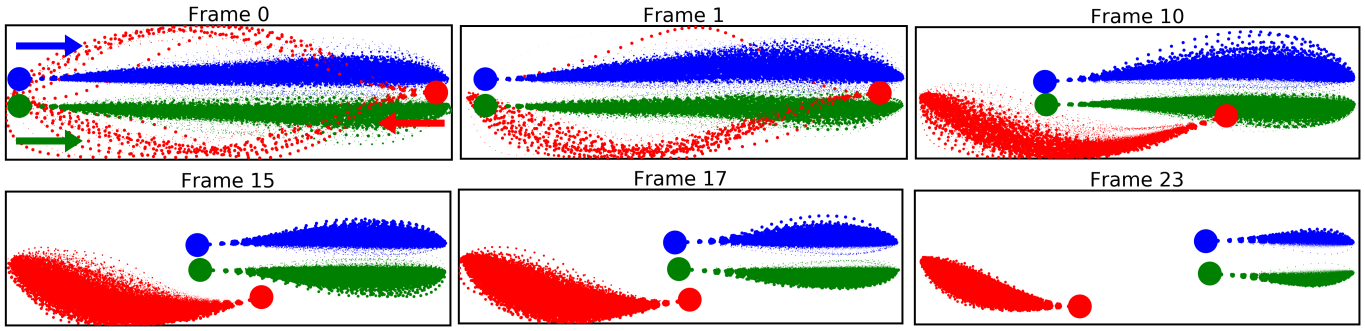

Fig. 4: Evolution of three agents' preferences while they are passing each other in a narrow hallway, each preference distribution is approximated by 1000 samples, each sample's size represents its log-scaled weight. Each agent's intent is approximated by the sample trajectory with largest weight, the sample distribution reflects the flexibility information, such as the covariance (spread of the samples), skew (symmetry of the sample weights) and kurtosis (number of outlier samples). **DistNav captures the change of both *intent* and *flexibility* of each agent during the interaction. This can be clearly seen in the first two frames where the red agent is simultaneously reasoning about going to the left and to the right of the blue and green agents—something impossible to represent with trajectory space coupling.**

agent i and rest of the agents, assuming their preferences are fixed, under this assumption optimizing the second term equals optimizing the overall joint expected collision penalty (III.4). Note that the construction of $\bar{c}_i^{(k)}(p)$ needs to follow a sequential order as shown in (IV.2): all agents with indices smaller than i would update their preferences for next iteration ahead of agent i, and their updated preferences are fixed in agent i's subproblem. For all agents with indices larger than i, their preferences are fixed as before being updated for next iteration.

Theorem IV.1. *The global minimum for the subproblem (IV.1) is:*

$$p_i^{(k+1)}(f) = \frac{p_i^{(k)}(f)\exp(-\bar{\gamma}_i^{(k)}(f))}{\int_{\mathcal{F}} p_i^{(k)}(f)\exp(-\bar{\gamma}_i^{(k)}(f))df} \tag{IV.6}$$

Proof: See appendix. ■

Theorem IV.2. *After all n agents' preferences have been updated sequentially in one iteration following (IV.6), the inequality below holds, if $p_i^{(k+1)}(f) \neq p_i^{(k)}(f)$ for some $i \in \mathcal{I}$:*

$$J_c(p_R^{(k+1)}, p_1^{(k+1)}, \ldots, p_n^{(k+1)}) \tag{IV.7}$$
$$\leq J_c(p_R^{(k)}, p_1^{(k)}, \ldots, p_n^{(k)}) - \xi\ ,\ \xi > 0 \tag{IV.8}$$

Proof: See appendix. ■

Theorem IV.2 shows that in each iteration, the joint expected collision penalty (III.4) is guaranteed to be sufficiently decreased by updating the preference for each agent based on (IV.6). Since (III.4) is bounded below at zero, with the decrease of its value, the subproblem (IV.1) comes closer to optimizing the KL-divergence between the preference at two iterations; therefore the new preference at the next iteration comes closer to the one in last iteration. While we leave the question of whether the iterative update (IV.6) could lead to guaranteed or efficient convergence to future work, in practice we don't actually look for the minimum of (III.4), which could be infeasible for navigation (e.g., a set of Dirac delta functions infinitely far from each other), but instead look for sufficiently small (III.4) that is safe enough. To terminate the iteration, one can threshold either the objective (III.4) or the KL-divergence between two iterations. Figure 3 shows an example of updating three 1D distributions using the update rule.

B. DistNav: Sampling-Based Crowd Navigation Based On Sequential Iterative Variational Analysis

Unfortunately, computing the preference updates (IV.6) analytically in high dimensional space is intractable due to the integrals in (IV.6), which is also known as the *curse of dimensionality*. Therefore, we propose a sampling-based motion planner based on (IV.6) to approximate the evolution of preferences and select the reference trajectory for robot navigation, the key idea here is to approximate the integrals through Monte Carlo integration.

We start by generating m samples from the original preference distribution of each agent, here we denote the samples for agent i as

$$[\mathbf{f}_i]^{(k)} \sim p_i^{(k)}(f) \tag{IV.9}$$
$$[\mathbf{f}_i]^{(k)} = \{\mathbf{f}_{i,1}^{(k)}, \mathbf{f}_{i,2}^{(k)}, \ldots, \mathbf{f}_{i,m}^{(k)}\} \tag{IV.10}$$

where each sample $\mathbf{f}_{i,j}^{(k)}$ indicates the j-th sample of agent i at k-th iteration, and it consists of two parts, the trajectory and the weight, we will not update the sample trajectory but only the weight:

$$\mathbf{f}_{i,j}^{(k)} = (f_{i,j}, w_{i,j}^{(k)}), \quad f_{i,j} \in \mathcal{F}, \quad w_{i,j}^{(k)} \in \mathbb{R} \tag{IV.11}$$

Before the iterations begin, the weight of each sample will be initialized as $w_{i,j}^{(0)} = 1$. In the iteration, the weight of each sample will first be updated based on (IV.6), where the integral of $\bar{\gamma}_i^{(k)}(f_{i,y})$ is approximated through Monte Carlo integration as:

$$\bar{\gamma}_i^{(k)}(f_{i,y}) \approx \sum_{j=R}^{i-1}\left(\frac{1}{m}\sum_{z=1}^{m}\psi(f_{i,y}, f_{j,z})\cdot w_{j,z}^{(k+1)}\right) + \sum_{j=i+1}^{n}\left(\frac{1}{m}\sum_{z=1}^{m}\psi(f_{i,y}, f_{j,z})\cdot w_{j,z}^{(k)}\right) \tag{IV.12}$$

And the weight of the sample is then updated as:

$$w_{i,j}^{(k+1)} = w_{i,j}^{(k)} \exp\left(-\bar{\gamma}_i^{(k)}(f_{i,y})\right) \tag{IV.13}$$

After all agent i's sample weights have been updated, we normalize the weights such that the average weight remains 1, this step is the approximation to the denominator in (IV.6). For each sample $\mathbf{f}_{i,j}^{(k)} = (f_{i,j}, w_{i,j}^{(k)})$, the updated preference for the trajectory is approximated as:

$$p_i^{(k)}(f_{i,j}) \approx w_{i,j}^{(k)} p_i^{(0)}(f_{i,j}) \tag{IV.14}$$

After the iteration terminates, the optimal trajectory of each agent is selected as the sample with largest approximated preference. Pseudocode of the whole algorithm can be found in appendix. Figure 4 shows how DistNav uses samples to approximate the evolution of three agents' preferences, where they pass each other in a narrow hallway.

V. Evaluation

A. Rationale for Both Simulated and Real World Dataset Evaluation

For evaluation, we considered the crowd datasets ETH [31] and UCY [26] and crowd simulators based on the the social forces model (SFM, [18]; e.g., PEDSIM [15]) and optimal reciprocal collision avoidance agents [47, 46], such as implemented in [4]. Ultimately, both evaluation methodologies have complementary strengths. For example, simulated agents can be overly permissive with aggressive robots: our Monte Carlo IGP produced zero collisions in PEDSIM while quickly navigating to the goal, whereas in our ETH study it was unsafe in 34/181 of runs (row 9, Table I) and exhibited freezing robot behavior. However, simulation provides information about how the algorithm leverages agent cooperation. Alternatively, the humans in pre-recorded datasets are non-responsive, and so the robot cannot leverage cooperation. However, pre-recorded datasets provide *human* benchmarks on safety and efficiency performance, which can be useful in assessing an algorithm's real world viability. Thus, we used both validation techniques to gain insight about a) the algorithm's ability to leverage cooperation and b) the algorithm's safety and efficiency performance compared to human safety and efficiency performance.

However, *how* to evaluate a navigation algorithm against ETH and UCY is non-trivial. For instance, many trajectories in these datasets have no interaction; thus, the majority of the runs in these datasets are not sufficiently challenging for a navigation study. Further, pre-recorded crowd datasets do not immediately suggest a *navigation* testing protocol (ETH and UCY are typically used to benchmark *prediction* algorithms, where testing protocol is straightforward). However, a subsample of the ETH dataset (Figure 5; 100 frames, 150 pedestrians) collected for testing a deep network in [19] has many interactions and substantial congestion; indeed, every pedestrian interacts at least once and most pedestrians interact many times during the 100 frame sequence. To derive a *navigation* test protocol, we expand on an idea from the experimental section of [42]: 1) identify a pedestrian, 2) extract the start and end position of that pedestrian, 3) remove that pedestrian from the observation dataset of the navigation algorithm, and 4) provide the start and end positions of the removed pedestrian and the current and previous positions of the remaining agents to the navigation algorithm. Thus we assure that at least one path through the crowd exists (the one taken by the removed pedestrian). Additionally, by providing the navigation algorithm with start and end points that are joined by a path *through* the crowd, the navigation algorithm naturally confronts high crowd densities (the human agent confronts an average density of 0.22 $people/m^2$ within a $3m$ radius circle). Finally, this testing protocol provides us with a powerful performance benchmark: *actual human performance on the exact same situation* as encountered by the navigation algorithm (first row, Table I). To determine our safety threshold, we computed the shortest distance that any two humans in the ETH dataset ever came to each other; that distance was 0.21m. Additionally, two humans only came within 0.3m of each other 3 times. Thus, if the robot is within 0.21m of any human, we consider that a *collision*, while distances within 0.3m are considered *unsafe* or *uncomfortable*.

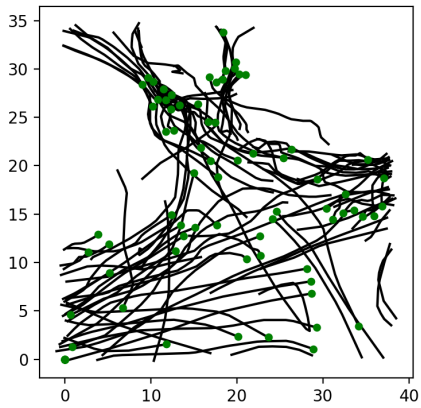

Fig. 5: First frame of the ETH data evaluated. Pedestrian current position in green; next 40 time steps plotted as black curves.

Furthermore, we partition this ETH dataset into what we call a "partial" trajectory dataset. In the partial trajectory dataset, we consider all (approximately) 10 meter long agent runs. For example, if agent 1's full trajectory was 30 meters long, we would have 3 partial trajectories. Partial trajectory experiments provide focused examination of an algorithm's ability to navigate through congestion in a safe and efficient manner. We identified 293 partial trajectories and tested 181 of them (the rest are discarded for calibration reason).

B. Rationale for Test Algorithms

We collected safety and path length data on humans, DistNav, "first order" interacting Gaussian processes (foIGP, [43]), "second order" IGP (soIGP, the extension of foIGP that considers both robot-agent and agent-agent interactions), soIGP using Monte Carlo optimization with 10^6 samples (so_MC_1e6, implementation details can be found in [43]), the "dynamic window approach" (DWA, [11]), and ORCA [47]. For our deep reinforcement learning baselines, we tested against "collision avoidance with deep reinforcement learning" (CADRL, [6]),

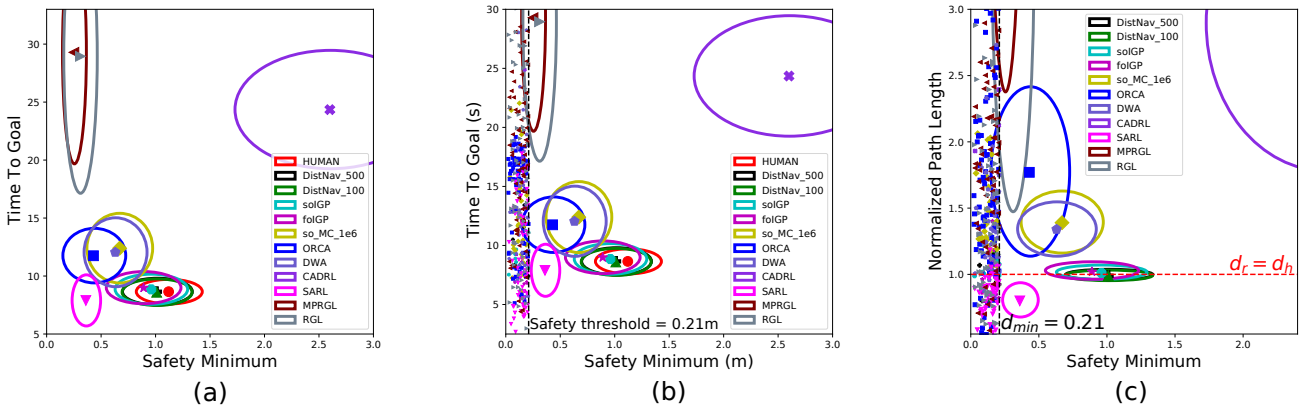

Fig. 6: **Partial trajectory statistics**. Total number of runs is 181; x axes of a) and b) in meters. All figures plot distance to nearest pedestrian on x-axis; (a) plots means $\pm$ 1 standard deviation of algorithm and the human; (b) appends plot (a) with the d_{min} threshold, the closest distance two humans passed in the dataset. Inspection of the region left of d_{min} shows numerous instances of DWA, so_MC_1e6, and SARL; (c) normalizes algorithm path length with human path length. E.g., values below $d_r = d_h$ mean that the robot moved to the goal more directly than the human.

"socially aware reinforcement learning" (SARL, [4]), "relational graph learning" and "model predictive relational graph learning" (RGL, MP-RGL, [5]).

Each algorithm was chosen to explore a certain aspect of the performance space. We collected data on humans to serve as an upper bound on performance. We tested soIGP and foIGP as state of the art trajectory space approaches. We tested DWA and ORCA because both are widely deployed; in particular, DWA is the default navigation algorithm in ROS (see ROS's base local planner). We tested against the 4 highest performing deep reinforcement learning variants to understand how model based and learning based algorithms fare against each other.

Finally, we trained SARL in 7 different environments using the toolbox implemented in [4] [2]. We trained in a 3m by 10m corridor (which mimics the ETH test conditions) with 15 and 5 people (0.5 and 0.16 people/m^2 densities), with mostly cross human cross traffic. The high density environment produced freezing robot behavior (freezing behavior $= 71\%$, $\max(d_r/d_h) = 23.8$, where d_r and d_h are the path lengths of the robot and human, respectively), while the low density training produced a policy that was unsafe (Collisions $= 34\%$). We thus attempted training in the high density corridor, but with random start and goal positions of the people; this again resulted in freezing robot behavior at test (freezing behavior $= 18\%$, $\max(d_r/d_h) = 14.7$). We also trained in a 4m radius circular environment with 10 people, so the density was ≈ 0.2 people/m^2 (the average density in the ETH data was ≈ 0.2 people/m^2). This policy also showed freezing robot behavior (freezing behavior $= 10\%$, $\max(d_r/d_h) = 4.32$). Additionally, we trained the other DRL variants (CADRL, RGL, and MP-RGL) in both ORCA and SFM training environments, with 5, 7, 14, 21, and 28 people in a 4m radius circular environment. Ultimately, the training regimen detailed in [4]—a 4m radius circle with 5 agents—produced the best performing policy for all DRL variants. We used these top performing policies for testing.

[2]The toolbox is available at https://github.com/vita-epfl/CrowdNav

C. Safety and Efficiency Evaluation on ETH

	Discomfort	Collisions	Freezing Behavior	$\max(d_r/d_h)$
Human	**1.7%**	**0**	**0%**	**1**
DistNav_500‡	**6.0%**	**3.0%**	**0%**	**1.18**
DistNav_100‡	**3.0%**	**1.0%**	**0%**	**1.18**
soIGP	13.3%	5%	1%	1.6
foIGP	16.7%	10.5%	3%	1.8
so_MC_1e6†	30%	18.8%	51%	5.3
ORCA	63%	48.6%	58%	9.2
DWA	35%	23.8%	48%	4.1
CADRL*	12%	6.6%	80%	14.7
SARL*	50%	31.5%	0.5%	3.3
RGL*	67.4%	48%	73%	28.1
MP-RGL*	62%	39%	88%	22.5

‡ DistNav_500 and DistNav_100 use 500 and 100 samples for each agent, respectively. The differences in the metrics between the two are partially from the sampling nature of the algorithm. A more comprehensive future evaluation would take multiple trials to eliminate such effects.

* SARL, CADRL, RGL, and MP-RGL were trained in 7 different environments; we report the best performing policy.

TABLE I: **ETH partial trajectory metrics**. "Discomfort" and "Collisions" are the percent of runs such that safety minimum distance $s < 0.3m, 0.21m$; "Freezing Behavior" is the percent of robot path lengths 1.25 times longer than the corresponding human path length; $\max(d_r/d_h)$ is the maximal ratio between the path lengths of the robot and human, it measures how *inefficient* the algorithm is compared with human.

The results of 181 partial trajectory runs are reported in Table I and Figure 6 (a more comprehensive evaluation table can be found in the appendix). For safety and efficiency, only DistNav and soIGP are competitive with human performance, with DistNav outperforming soIGP. We tested DistNav with 100 samples per agent, in which case the algorithm can run in real time (average replanning time 0.23s). We also tested with 500 samples per agent, which resulted in longer computation time but no significant improvement in performance. In practice we think 100 samples per agent is a good balance between performance and computation efficiency.

Additionally, DWA and ORCA both exhibit freezing robot behavior (large value in "freezing behavior" column) and high rates of collision (large values in "collision" column). We note that all four DRL variants (last four rows of Table I)

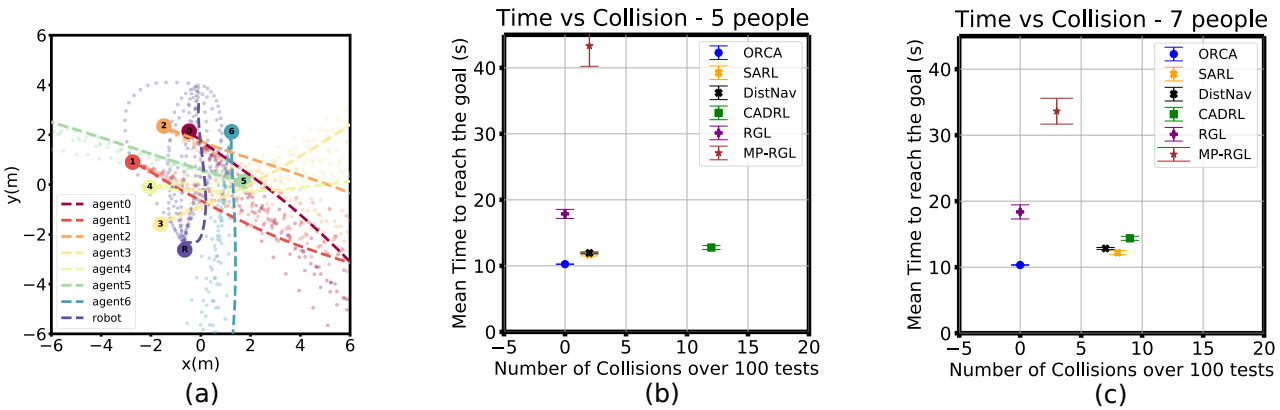

Fig. 7: **ORCA evaluation statistics.** (a) shows one frame of the 7 agents test in ORCA simulation, the dashed lines are the robot's planned trajectory and optimal prediction for pedestrian trajectories, the colored dots are parts of the samples used to approximate preference distributions. (b) and (c) plot each algorithm's number of collisions vs. mean time to reach the goal over 100 tests with 5 and 7 pedestrians, respectively.

display extremely long paths (CADRL, RGL, MP-RGL), high rates of collision (SARL, RGL, MP-RGL) or both (RGL, MP-RGL). Given the wide variety of testing protocols, these results indicate potential limitations of purely simulation-based training.

We emphasize that although ETH provides useful information, it does not *validate* that an algorithm will perform well in the real world (this requires real world experiments). Nevertheless, we believe that the ETH benchmark can provide evidence of *invalidation*; that is, poor performance on ETH possibly indicates that an algorithm is not suitable for real world deployment.

D. Safety and Efficiency Evaluation in Simulation

We complement our ETH study with two simulation studies: one with 5 ORCA agents (the standard testing environment for DRL studies [4, 6, 5]) and one with 7 agents. Our ORCA simulators are closely related to those developed in [4], with a few important differences:

- The robot is visible in both the 5 and 7 agent test because we wish to understand how our algorithm performs in the presence of responsive agents.
- Once an ORCA agent reaches a goal, a new goal is provided to the agent. In this way, the agents circulate in the work space and the crowd density remains consistent throughout the run. The simulator in [4] provides a single goal to each agent; the agent stops once it arrives. Thus, our simulator has a higher average crowd density than the simulator in [4].

We point out that the ORCA *robot* outperforms all the algorithms in both the 5 and 7 person simulation; this is to be expected, since a group of ORCA agents have guarantees on collision performance and locally optimal efficiency. In short, an ORCA robot is perfectly tuned to an ORCA simulation. However, as seen in Table I, ORCA is unsuitable for deployment in scenarios with agents not obeying the ORCA protocol.

For the five person simulation, we see that only ORCA and RGL outperform DistNav in terms of safety; however, RGL exhibits substantially longer mean time to goal, indicating freezing robot like behavior (e.g., RGL often chooses to go around the crowd). SARL shows nearly identical performance to DistNav, but its critical to recall that SARL (and the other DRL variants) were specifically trained in a 5 person ORCA simulator. Thus, all the DRL variants have a large advantage over DistNav: they have been precisely tuned to this simulation, whereas DistNav has not. In combination with the ETH results in Figure 6 and Table I, DistNav displays substantially stronger performance, both in terms of safety and efficiency, and, more importantly, in the ability to generalize to novel scenarios. The 7 person case shows nearly identical qualitative results (although the exact number of collisions or time to goal changes slightly, the ordering of the algorithms remains the same.)

Finally, we attempted tests with the number of ORCA agents higher than 7, but this led to hard-to-interpret results because of simulator failures. For example, at higher densities, agents can be so close together that collisions are often *caused* by the ORCA agents themselves (e.g., an ORCA agent runs into the robot). While testing in simulation at higher densities is important, fixing the simulator is out of scope of this paper.

VI. Conclusion

We studied the crowd navigation problem by modeling both human and robotic actions as probability density functions (called *preference distributions*). This formulation, together with an optimization algorithm, captures the evolution of agents' preferences in the presence of interaction, something not modeled using trajectory space coupling. Further, we designed a sampling-based crowd navigation method, called DistNav, and benchmarked against a variety of methods in both a real world dataset and in simulation. In both the dataset and simulation evaluation, Distnav outperformed all other algorithms and was competitive with human safety and efficiency performance.

Acknowledgments

This material is supported by the NSF Grant CNS 1837515. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the aforementioned institutions.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social-LSTM: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[2] Wolfram Burgard, Armin B Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. The interactive museum tour-guide robot. In *Proceedings of the conference on Artificial intelligence/Innovative applications of artificial intelligence (AAAI)*, 1998.

[3] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*, 2019.

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

[5] Changan Chen, Sha Hu, Payam Nikdel, Greg Mori, and Manolis Savva. Relational graph learning for crowd navigation. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

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

[7] Simon Le Cleac'h, Mac Schwager, and Zachary Manchester. Algames: A fast solver for constrained dynamic games. *arXiv preprint arXiv:1910.09713*, 2019.

[8] N. Du Toit and J. Burdick. Robot motion planning in dynamic, uncertain environments. *IEEE Transactions on Robotics*, 2012.

[9] T. Fan, P. Long, W. Liu, and J. Pan. Fully distributed multi-robot collision avoidance via deep reinforcement learning for safe and efficient navigation in complex scenarios. *The International Journal of Robotics Research*, 2018.

[10] T. Fan, X. Cheng, J. Pan, P. Long, W. Liu, R. Yang, and D. Manocha. Getting robots unfrozen and unlost in dense pedestrian crowds. *IEEE Robotics and Automation Letters (RA-L)*, 2019.

[11] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 1997.

[12] Thierry Fraichard and Valentin Levesy. From crowd simulation to robot navigation in crowds. *IEEE Robotics and Automation Letters*, 2020.

[13] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C.J Tomlin. Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games. In *IEEE international conference on robotics and automation (ICRA)*, 2020.

[14] Izrail Moiseevitch Gelfand and Richard A Silverman. *Calculus of variations*. Courier Corporation, 2000.

[15] C. Gloor. Pedsim: pedestrian crowd simulation, 2016. URL http://pedsim.silmaril.org.

[16] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[17] E. Hall. *The Hidden Dimension*. Doubleday, 1966.

[18] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 1995.

[19] B. Ivanovich and M. Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[20] B. Ivanovich, E. Schmerling, K. Leung, and M. Pavone. Generative modeling of multimodal multi-human behavior. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[21] J. Joseph, F. Doshi-Velez, and N. Roy. A bayesian non-parametric approach to modeling mobility patterns. *Autonomous Robots*, 2011.

[22] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard. Socially compliant mobile robot navigation via inverse reinforcement learning. In *The International Journal of Robotics Research*, 2016.

[23] T. Kruse, R. Alami, A.K. Pandey, and A. Kirsch. Human-aware robot navigation: a survey. In *Robotics and Autonomous Systems*, 2013.

[24] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 2015.

[25] Simon Lecleac'h, Mac Schwager, and Zachary Manchester. Lucidgames: Online unscented inverse dynamic games for adaptive trajectory prediction and planning. *IEEE Robotics and Automation Letters*, 2021.

[26] A. Lerner, Y. Chrysanthou, and D. Lischinski. Crowds by example. In *Computer graphics forum*, 2007.

[27] Yuejiang Liu, Qi Yan, and Alexandre Alahi. Social nce: Contrastive learning of socially-aware motion representations. *arXiv preprint arXiv:2012.11717*, 2020.

[28] R. Mead, A. Atrash, and M. Mataric. Proxemic feature recognition for interactive robots: automating social science metrics. In *International conference on social robotics*, 2011.

[29] S. Nikolaidis, P. Lasota, R. Ramakrishnan, and J. Shah. Human robot mutual adaptation in collaborative tasks: models and experiments. *The International Journal of Robotics Research*, 2017.

[30] H. Nishimura, B. Ivanovich, A. Gaidon, M. Pavone, and M. Schwager. Risk-sensitive sequential action control with multi-modal human trajectory forecasting for safe crowd-robot interaction. *International Conference on Intelligent Robots and Systems*, 2020.

[31] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll never walk alone: modeling social behavior for multi-target tracking. In *International Conference on Computer Vision*, 2009.

[32] M. Pfeiffer, G. Paolo, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena. A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments. In *IEEE international conference on robotics and automation (ICRA)*, 2018.

[33] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. URL http://www.gaussianprocess.org/gpml/.

[34] Jorge Rios-Martinez, Anne Spalanzani, and Christian Laugier. Understanding human interaction for probabilistic autonomous navigation using risk-RRT. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[35] D. Sadigh, N. Landolfi, S.S. Sastry, S.A. Seshia, and A. D. Dragan. Planning for cars that coordinate with people: Leveraging effects on human actions for planning and active information gathering over human internal state. *Autonomous Robots*, 2018.

[36] Wilko Schwarting, Alyssa Pierson, Javier Alonso-Mora, Sertac Karaman, and Daniela Rus. Social behavior for autonomous vehicles. *Proceedings of the National Academy of Sciences*, 2019.

[37] Riccardo Spica, Eric Cristofalo, Zijian Wang, Eduardo Montijano, and Mac Schwager. A real-time game theoretic planner for autonomous two-player drone racing. *IEEE Transactions on Robotics*, 2020.

[38] M. Svenstrup, T. Bak, and J. Andersen. Trajectory planning for robots in dynamic human environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[39] L. Tai, J. Zhang, X. Lin, and W. Burgard. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. *IEEE international conference on robotics and automation (ICRA)*, 2018.

[40] Simon Thompson, Takehiro Horiuchi, and Satoshi Kagami. A probabilistic model of human motion and navigation intent for mobile robot path planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009.

[41] Sebastian Thrun, Michael Beetz, Maren Bennewitz, Wolfram Burgard, Armin B Cremers, Frank Dellaert, Dieter Fox, Dirk Haehnel, Chuck Rosenberg, Nicholas Roy, et al. Probabilistic algorithms and the interactive museum tour-guide robot minerva. *The International Journal of Robotics Research*, 2000.

[42] P. Trautman and A. Krause. Unfreezing the robot: Navigation in dense interacting crowds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[43] P. Trautman and K. Patel. Real time crowd navigation from first principles of probability theory. *Proceedings of the international conference on automated planning and scheduling*, 2020.

[44] P. Trautman, J. Ma, R. M. Murray, and A. Krause. Robot navigation in dense human crowds: Statistical models and experimental studies of human robot cooperation. *The International Journal of Robotics Research*, 2015.

[45] Vaibhav V Unhelkar, Claudia Pérez-D'Arpino, Leia Stirling, and Julie A Shah. Human-robot co-navigation using anticipatory indicators of human walking motion. In *IEEE international conference on robotics and automation (ICRA)*, 2015.

[46] J. van den Berg, M. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *IEEE international conference on robotics and automation (ICRA)*, 2008.

[47] J. van den Berg, S. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. In *International Symposium of Robotic Research (ISRR)*, 2009.

[48] A. Vemula, K. Muelling, and J. Oh. Social attention: Modeling attention in human crowds. In *IEEE international conference on robotics and automation (ICRA)*, 2018.

[49] Mingyu Wang, Zijian Wang, John Talbot, J Christian Gerdes, and Mac Schwager. Game theoretic planning for self-driving cars in competitive scenarios. In *Robotics: Science and Systems*, 2019.

[50] Brian D Ziebart, Andrew L Maas, Anind K Dey, and J Andrew Bagnell. Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. In *Proceedings of the international conference on Ubiquitous computing*, 2008.

[51] Brian D Ziebart, Nathan Ratliff, Garratt Gallagher, Christoph Mertz, Kevin Peterson, J Andrew Bagnell, Martial Hebert, Anind K Dey, and Siddhartha Srinivasa. Planning-based prediction for pedestrians. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009.

APPENDIX

A. Evaluation on ETH dataset with Supplementary Metrics

See Table II.

B. Pseudocode of DistNav

See Algorithm 1.

C. Proof

Proof for Theorem IV.1

Proof: The subproblem can be considered as an isoperimetric problem with a subsidiary condition (III.1), therefore we first formulate the Lagrangian as

$$\mathcal{L}(p,\lambda) = D_{KL}(p\|p_i^{(k)}) + \bar{c}_i^{(k)}(p) - \lambda(\int_{\mathcal{F}} p(f)df - 1) \tag{A.1}$$

$$\bar{c}_i^{(k)}(p) = \int_{\mathcal{F}} p(f)\bar{\gamma}_i^{(k)}(f)df \tag{A.2}$$

	Discomfort	Collisions	Freezing Behavior	$\max(d_r/d_h)$	$\mu(s)$	$\mu(d_r)$	$\mu(t)$
Human	1.7%	0	0%	1	$1.1 \pm .2m$	$8.7 \pm 1.0m$	NA
DistNav_500‡	6.0%	3.0%	0%	1.18	$1.01 \pm .32m$	$8.65 \pm 1.15m$	$1.07 \pm 0.83s$
DistNav_100‡	3.0%	1.0%	0%	1.18	$1.01 \pm .32m$	$8.6 \pm 1.14m$	$0.26 \pm 0.07s$
soIGP	13.3%	5%	1%	1.6	$.96 \pm .3m$	$8.9 \pm 1.3m$	$4.3 \pm .2s$
foIGP	16.7%	10.5%	3%	1.8	$.9 \pm .3m$	$9 \pm 1.5m$	$4.2 \pm .15s$
so_MC_1e6†	30%	18.8%	51%	5.3	$.67 \pm .3m$	$12.4 \pm 3m$	$6.1 \pm 0.8s$†
ORCA	63%	48.6%	58%	9.2	$.43 \pm .3m$	$11.8 \pm 2.4m$	$0.03 \pm .001s$
DWA	35%	23.8%	48%	4.1	$.6 \pm .4m$	$12.1 \pm 2.5m$	$.1 \pm .03s$
CADRL*	12%	6.6%	80%	14.7	$2.6 \pm .9m$	$24.35 \pm 5.1m$	$2.9 \pm .4s$
SARL*	50%	31.5%	0.5%	3.3	$.4 \pm .15m$	$7.9 \pm 2.2m$	$4.4 \pm 1.1s$
RGL*	67.4%	48%	73%	28.1	$0.3 \pm .15m$	$28.95 \pm 12.6m$	$2.1 \pm .7s$
MP-RGL*	62%	39%	88%	22.5	$0.33 \pm .16m$	$38.1 \pm 12.5m$	$0.3 \pm .01s$

‡ DistNav_500 and DistNav_100 use 500 and 100 samples for each agent, respectively. They were both run on a 12 core CPU, parallelized and accelerated by the Numba Python package [24], but no GPU is used.
† so_MC_1e6 was run on fully parallelized code on a 64 core CPU to achieve these times.
* SARL, CADRL, RGL, and MP-RGL were trained in 7 different environments; we report the best performing policy.

TABLE II: **ETH supplementary partial trajectory metrics**. Distance to nearest pedestrian is s and $\mu(s)$ is the mean; "Discomfort" and "Collisions" are the percent of runs such that $s < 0.3m, 0.21m$; "freezing behavior" is the percent of robot path lengths 1.25 times longer than the corresponding human path length; $\max(d_r/d_h)$ is the largest value of robot path length divided by corresponding human path length; $\mu(d_r)$ is mean robot path length d_r over all runs; $\mu(t)$ is mean time of all replanning steps.

where $\lambda \in \mathbb{R}$ is Lagrange multiplier. The necessary condition for $p^*(f)$ to be an extremum for the subproblem can be written as (Theorem 1, Page 43 [14]):

$$\frac{\partial \mathcal{L}}{\partial p}(p^*, \lambda) = \log p^*(f) + 1 - \log p_i^{(k)}(f) + \bar{\gamma}_i^{(k)}(f) - \lambda = 0 \tag{A.3}$$

$$p^*(f) = p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f) + \lambda - 1) \tag{A.4}$$

By substituting (A.4) into the equality constraint (III.1), we can solve for λ:

$$\int_{\mathcal{F}} p^*(f)df = \int_{\mathcal{F}} p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f) + \lambda - 1)df \tag{A.5}$$

$$= \exp(\lambda - 1) \int_{\mathcal{F}} p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f))df = 1 \tag{A.6}$$

$$\exp(\lambda - 1) = \frac{1}{\int_{\mathcal{F}} p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f))df} \tag{A.7}$$

Substituting (A.7) into (A.4) gives us:

$$p^*(f) = \frac{p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f))}{\int_{\mathcal{F}} p_i^{(k)}(f) \exp(-\bar{\gamma}_i^{(k)}(f))df} \tag{A.8}$$

Since the subproblem objective is unbounded from above, the solution $p^*(f)$ is a global minimum, which completes the proof. ■

Lemma A.1. *The subproblem solution (IV.6) can sufficiently decrease the second term $\bar{c}_i^{(k)}(p)$ in the subproblem objective (IV.1), if $p_i^{(k+1)}(f) \neq p_i^{(k)}(f)$:*

$$\bar{c}_i^{(k)}(p_i^{(k+1)}) \leq \bar{c}_i^{(k)}(p_i^{(k)}) - \xi \tag{A.9}$$

$$\xi > 0 \tag{A.10}$$

Proof: Since p_i^{k+1} is the global minimum of the subproblem (IV.1), we have:

$$D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) + \bar{c}_i^{(k)}(p_i^{(k+1)}) \tag{A.11}$$

$$\leq D_{KL}(p_i^{(k)} \| p_i^{(k)}) + \bar{c}_i^{(k)}(p_i^{(k)}) = \bar{c}_i^{(k)}(p_i^{(k)}) \tag{A.12}$$

$$\bar{c}_i^{(k)}(p_i^{(k+1)}) \leq \bar{c}_i^{(k)}(p_i^{(k)}) - D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) \tag{A.13}$$

If $p_i^{(k+1)}(f) \neq p_i^{(k)}(f)$, then $D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) > 0$, which completes the proof. ■

Proof for Theorem IV.2

Proof: Based on Lemma A.1, we have for each $i \in \mathcal{I}$, the following inequality holds:

$$\bar{c}_i^{(k)}(p_i^{(k+1)}) \leq \bar{c}_i^{(k)}(p_i^{(k)}) - D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) \tag{A.14}$$

Summing up left hand side of the inequality for all $i \in \mathcal{I}$ gives us:

$$\sum_{i=R}^{n} \bar{c}_i^{(k)}(p_i^{(k+1)})$$

$$= \sum_{i=R}^{n} \sum_{j=R}^{i-1} c(p_i^{(k+1)}, p_j^{(k+1)}) + \sum_{i=R}^{n} \sum_{j=i+1}^{n} c(p_i^{(k+1)}, p_j^{(k)}) \tag{A.15}$$

$$= \sum_{i=R}^{n} \sum_{j=i+1}^{n} c(p_i^{(k+1)}, p_j^{(k+1)}) + \sum_{i=R}^{n} \sum_{j=R}^{i-1} c(p_i^{(k)}, p_j^{(k+1)}) \tag{A.16}$$

The last equality above is based on the structure of the combinatorial summation and the fact that $c(p_i, p_j) = c(p_j, p_i)$. Meanwhile summing up the right hand side of the inequality for all $i \in \mathcal{I}$ gives us:

$$\sum_{i=R}^{n} \bar{c}_i^{(k)}(p_i^{(k)}) - \sum_{i=R}^{n} D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) \tag{A.17}$$

$$= \sum_{i=R}^{n} \bar{c}_i^{(k)}(p_i^{(k)}) - \xi \tag{A.18}$$

$$= \sum_{i=R}^{n} \sum_{j=R}^{i-1} c(p_i^{(k)}, p_j^{(k+1)}) + \sum_{i=R}^{n} \sum_{j=i+1}^{n} c(p_i^{(k)}, p_j^{(k)}) - \xi \tag{A.19}$$

Now by the combining summation of both sides of the inequality, we would have:

$$\sum_{i=R}^{n}\sum_{j=i+1}^{n} c(p_i^{(k+1)}, p_j^{(k+1)}) + \sum_{i=R}^{n}\sum_{j=R}^{i-1} c(p_i^{(k)}, p_j^{(k+1)}) \tag{A.20}$$

$$\leq \sum_{i=R}^{n}\sum_{j=R}^{i-1} c(p_i^{(k)}, p_j^{(k+1)}) + \sum_{i=R}^{n}\sum_{j=i+1}^{n} c(p_i^{(k)}, p_j^{(k)}) - \xi \tag{A.21}$$

and therefore

$$\sum_{i=R}^{n}\sum_{j=i+1}^{n} c(p_i^{(k+1)}, p_j^{(k+1)}) \leq \sum_{i=R}^{n}\sum_{j=i+1}^{n} c(p_i^{(k)}, p_j^{(k)}) - \xi \tag{A.22}$$

If $p_i^{(k+1)}(f) \neq p_i^{(k)}(f)$ for some $i \in \mathcal{I}$, then $\xi = \sum_{i=R}^{n} D_{KL}(p_i^{(k+1)} \| p_i^{(k)}) > 0$, which completes the proof. ■

D. Implementation Details for DistNav

1) Choice of Collision Penalty Function: The collision penalty for two trajectories should be evaluated on time-aligned elements (poses at each time step), in our implementation we use a Gaussian penalty function and select the maximal collision penalty among all time steps. Even though theoretically a Dirac delta function should work as collision penalty since preference distribution already contains information about "comfort distance", for the sampling-based method an explicit collision penalty is still necessary.

2) Selection of Critical Agents: After generating the initial samples for all agents, we compute the weights of all other agents' GP preferences on the robot's intent (GP mean) based on (IV.12) as the "interaction score". Agents with scores higher than a user-defined threshold are considered as critical agents for interaction, and thus are included for coupled prediction and planning. This process could drastically reduce the computation time.

Algorithm 1: Sampling-Based Crowd Navigation Based On Sequential Iterative Variational Analysis (DistNav)

Input : $[[\mathbf{f}_0], [\mathbf{f}_1], \ldots, [\mathbf{f}_n]]$: Samples representing initial preferences of $n+1$ agents, each agent has m samples. The weight of each sample is initialized as 1. Index 0 indicates the robot.
$\psi(x_1, x_2)$: collision penalty function.
$[p_1(f), p_2(f), \ldots, p_n(f)]$: Original preference distributions of $n+1$ agents.
ϵ: Termination condition.

Output: $[f_0^*, f_1^*, \ldots, f_n^*]$: Optimal trajectories of n agents selected from samples. Index 0 indicates the robot.

$i \leftarrow 0$
while $i \leq n$ **do**
  $[\mathbf{f}_i]^{(0)} \leftarrow [\mathbf{f}_i]$
  $i \leftarrow i + 1$
end while
$k \leftarrow 0$
$objective \leftarrow \frac{1}{m}\sum_{i=0}^{n}\sum_{j=i+1}^{n}\sum_{l=0}^{m}\left(\psi(f_{i,m}, f_{j,m}) \cdot w_{i,m}^{(k)} \cdot w_{j,m}^{(k)}\right)$
while $objective \geq \epsilon$ **do**
  $i \leftarrow 0$
  while $i \leq n$ **do**
    $j \leftarrow 1$
    while $j \leq m$ **do**
      $v \leftarrow \sum_{l=0}^{l=i-1}\sum_{h=1}^{h=m}\left(\psi(f_{i,j}, f_{l,h}) \cdot w_{l,h}^{(k+1)}\right) + \sum_{l=i+1}^{l=n}\sum_{h=1}^{h=m}\left(\psi(f_{i,j}, f_{l,h}) \cdot w_{l,h}^{(k)}\right)$
      $w_{i,j}^{(k+1)} \leftarrow w_{i,j}^{(k)} \cdot \exp(-\frac{v}{m})$
      $j \leftarrow j + 1$
    end while
    $j \leftarrow 1$
    while $j \leq m$ **do**
      $w_{i,j}^{(k+1)} \leftarrow w_{i,j}^{(k+1)} / \left(\frac{1}{m}\sum_{l=0}^{m} w_{i,j}^{(k+1)}\right)$
      $j \leftarrow j + 1$
    end while
    $i \leftarrow i + 1$
  end while
  $objective \leftarrow \frac{1}{m}\sum_{i=0}^{n}\sum_{j=i+1}^{n}\sum_{l=0}^{m}\left(\psi(f_{i,m}, f_{j,m}) \cdot w_{i,m}^{(k)} \cdot w_{j,m}^{(k)}\right)$
  $k \leftarrow k + 1$
end while
$i \leftarrow 0$
while $i \leq n$ **do**
  $f_i^* \leftarrow \arg\max_{f_{i,j}^{(k)}} p_i(f_{i,j}^{(k)}) w_{i,j}^{(k)}$
  $i \leftarrow i + 1$
end while
return $[f_0^*, f_1^*, \ldots, f_n^*]$