Learning Goal Conditioned Socially Compliant Navigation From Demonstration Using Risk-Based Features

Abhisek Konar[1], Bobak H. Baghi[1], and Gregory Dudek[1,2]

***Abstract*—One of the main challenges of operating mobile robots in social environments is the safe and fluid navigation therein, specifically the ability to share a space with other human inhabitants by complying with the explicit and implicit rules that we humans follow during navigation. While these rules come naturally to us, they resist simple and explicit definitions. In this paper, we present a learning-based solution to address the question of socially compliant navigation, which is to navigate while maintaining adherence to the navigational policies a person might use. We infer these policies by learning from human examples using inverse reinforcement learning techniques. In particular, this paper contributes an efficient sampling-based approximation to enable model-free deep inverse reinforcement learning, and a goal conditioned risk-based feature representation that adequately captures local information surrounding the agent. We validate our approach by comparing against a classical algorithm and a reinforcement learning agent and evaluate our feature representation against similar feature representations from the literature. We find that the combination of our proposed method and our feature representation produce higher quality trajectories and that our proposed feature representation plays a critical role in successful navigation.**

***Index Terms*—Robot Navigation, Social Navigation, Inverse Reinforcement Learning, Motion and Path Planning, Learning from Demonstration.**

I. Introduction

THIS work presents an inverse reinforcement learning-based navigation pipeline that can produce artificial agents that are capable of demonstrating socially compliant behavior. With recent strides in robotic sensing and artificial intelligence, it seems ever more likely that robots will come to inhabit human social spaces. Recent work has shown success when integrating robots into various social domains such as service robots for the elderly [1], [2], as shopping assistants [3], [4], guides at airports [5], and in the office environment [6], [7]. A significant challenge in integrating mobile robots into human social spaces is the need for socially compliant navigation. In this context, socially compliant navigation policies (sometimes known as "socially aware") are those that result in navigation that is consistent with normal socially acceptable behavior as exhibited by humans. In particular, this implies that in addition to obstacle avoidance and goal-seeking, such a policy must take into account subtle social cues, implicit socially-motivated intent, and behavioral signals from surrounding humans. Such policies seek to provide a degree of confidence and familiarity to the human inhabitants of said social space to go about their business with minimal modification to their behaviors. Thus, any socially compliant navigation algorithm must maintain a certain level of knowledge about human behavior as a foundation for its decision making.

Navigation and efficient motion planning comprise of one of the richest and well-understood parts of robotics (despite various open problems). For social navigation, however, the need to account for "appropriate" behavior is often more important than mere efficiency. Likewise, purely reactive strategies can lead to a range of problems. While reinforcement learning can lead to powerful solutions when the reward structure for the problem is well understood, social navigation depends on a range of poorly understood cues that may depend on the scenario. As such, inverse reinforcement learning is attractive since it allows us to infer suitable rewards from raw features based on observations. It, however, can be costly to compute and also demands a good performance metric that can be used to assess human-like behavior. Notably, however, some social cues are outside the scope of this work. For example, at a cocktail party, a person crying may influence the navigation behavior of other people nearby and our systems are currently oblivious to such facial and body-pose cues. In the outdoor datasets we use, such cues are probably less critical, but this does imply certain limitations on expected performance.

In this paper, we propose an efficient sampling-based approximation to enable model-free inverse reinforcement learning using deep networks to represent both the policy and the reward function. We also propose a goal-conditioned risk-based feature representation for the social navigation problem that captures local information surrounding the agent.

In the following sections, we review related work (section II) and background knowledge (section III), followed by the introduction of our IRL pipeline and risk-based feature representation (section IV), which constitute the main contributions of this work. We then proceed to evaluate our proposed

Manuscript received: July, 01, 2020; Revised October, 22, 2020; Accepted December, 10, 2020.

This paper was recommended for publication by Tamim Asfour upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Samsung Electronics, and also benefited from the infrastructure provided by the National Science and Engineering Council of Canada (NSERC).

[1]The authors are with School of Computer Science, McGill University, 3480 Rue University, Montreal, Quebec, Canada. H3A 2K6 akonar@cim.mcgill.ca

[2]Gregory Dudek is also with Samsung Electronics AI Center Montreal, 1250 René-Lévesque, Montreal, Quebec, Canada. H3B 4K4 greg.dudek@samsung.com

Digital Object Identifier (DOI): see top of this page.

method in a variety of settings (section V).

II. Related Work

Approaches to socially compliant navigation can be divided into two broad categories: classical methods and learning-based methods. Classical methods seek to explicitly model the "rules" for human social navigation while learning-based methods seek to explicitly or implicitly infer human social navigation attributes from some dataset of human trajectories. Among classical methods, pioneering work from Helbing and Molnar models human behavior using (carefully tuned) forces exerted between agents in an environment [8]. Some researchers have used camera footage to calibrate these parameters [9], [10], while others have made use of genetic algorithms to the same end [11].

In contrast, learning-based methods attempt to learn human social navigation behavior from examples, such as human motion datasets. Several techniques have been investigated including those that make use of LSTMs [12] in a reinforcement learning framework to produce socially-aware motion planners [13]. Others include a prediction model that works in tandem with a planning module to chalk out the best set of actions [14], [15]. A significant challenge of social navigation is that the criteria and associated rewards that characterize ideal performance are not readily available even though humans are proficient at the task. As a result, inverse reinforcement learning (IRL) methods that can infer the underlying reward structure are especially appealing [16]. A particularly popular approach is maximum entropy IRL (MaxEnt IRL) [17], which can then be used to train behavioral policies via reinforcement learning (RL) when combined with a local or global planning scheme. Pfeiffer et al. propose a method to select trajectories that minimize costs learned from MaxEnt IRL [18]. Kim et al. use a global path planner but learn to plan local paths using MaxEnt IRL with data from RGB-D cameras [19].

The use of MaxEnt IRL, however, requires a representative sampling of the trajectory space which can be quite challenging in high dimensional and continuous spaces. For instance, Kuderer et al. divide the trajectory space topologically, based on the preference of passing obstacles by the left or right side [20]. This approximation allows them to significantly reduce the number of samples required to formulate a representative estimate by choosing only one representative sample per partition. Kretzschmar et al. [21], building on their previous work [22], make use of hybrid Monte Carlo [23] to efficiently guide the sampling of a continuous trajectory space. We modify MaxEnt IRL to reduce the amount of sampling required, as detailed in subsection IV-A.

An important choice in the MaxEnt IRL framework is the feature design used to learn the reward function and train the RL agent. Vasquez et al. [24] provide a comparative study of different feature representations, which are computed from the continuous positions and velocities of the agent and surrounding pedestrians. Similarly, Fahad et al. [25] make use of social affinity map features (SAM) [26], augmenting them with information from pedestrian velocities and employ maximum entropy deep IRL (MEDIRL) [27] which allows for richer representations of reward functions via the use of expressive neural networks. Our work builds on [25] and [24] by introducing a novel feature representation based on apparent collision risk, as well as a deterministic sampling method which greatly simplifies gradient computation for MEDIRL.

III. Inverse Reinforcement Learning

In this section, we briefly outline and provide the necessary background for our IRL method. Consider a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma, p_0)$ where $\mathcal{S}$ is the set of possible states, $\mathcal{A}$ is the set of possible actions, $\mathcal{T}$ are the transition probabilities $P(s'|s,a)$ for states $s, s' \in \mathcal{S}$ and action $a \in \mathcal{A}$, $R : \mathcal{S} \mapsto \mathbb{R}$ is the reward function, γ is the discounting factor, and p_0 is the probability distribution over initial states. The goal of IRL is to determine the optimal reward function R^* given demonstration trajectories $\mathcal{D} = \{\tau_1, \tau_2, \ldots, \tau_M\}$ from expert agents, where the trajectories $\tau_i = \{s_0, s_1, \ldots, s_T\}$ are comprised of states visited by the expert agents. Note that trajectories are comprised only of states due to the state-only formulation of the reward function. In practice, due to the continuous nature and complexity of the state space, hand-engineered features $\phi : \mathcal{S} \mapsto \mathbb{R}^N$ are computed for use instead of raw state information.

Early work on IRL by [16] shows that solving the Bellman equations for an analytic solution for the reward R is underspecified. To address this ambiguity, [17] impose a maximum entropy [28] constraint on the distribution of possible trajectories:

$$P(\tau) \propto \exp\left(R_{\boldsymbol{\theta}}(\tau)\right) = \exp\left(\sum_{s_i \in \tau} R_{\boldsymbol{\theta}}(s_i)\right) \tag{1}$$

where the reward function $R_{\boldsymbol{\theta}}(s_i)$ is represented a set of linear weights: $R_{\boldsymbol{\theta}}(s_i) = \boldsymbol{\theta}^\top \phi(s_i)$. Given a set of expert demonstrations, an optimal reward function can be learned by maximizing the log-likelihood L of the expert demonstrations:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{\tau \in \mathcal{D}} \log P(\tau|\boldsymbol{\theta}) \tag{2}$$

This linearly parameterized reward function suffers from poor modeling power in the face of complex tasks such as human navigation. Wulfmeier et al. [27] introduce maximum entropy deep inverse reinforcement learning (MEDIRL) by extending MaxEnt IRL to deep neural network reward functions, which perform better than linear parameters in modeling complex functions. The parameters $\boldsymbol{\theta}$ now become the parameters of the reward network and can be found using gradient descent methods. It is shown in [27] that the gradient of the loss function found in (2) is equal to the difference in state visitation frequencies (SVF) of expert demonstrations and a sampling policy $\pi_{\boldsymbol{\psi}} : \mathcal{S} \mapsto \mathcal{A}$ trained on the reward function:

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \sum_{s \in S} \left(\mu_{\mathcal{D}}(s) - \mu_{\pi_\psi}(s) \right) . \frac{\partial R_{\boldsymbol{\theta}}(\phi(s))}{\partial \boldsymbol{\theta}} \tag{3}$$

$$\mu_{\mathcal{D}}(s_i) = \sum_{\tau \in \mathcal{D}} \sum_{s_t \in \tau} \mathbf{1}_{s_i = s_t} \tag{4}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.3048657, IEEE Robotics and Automation Letters

where $\mu_{\mathcal{D}}(s)$ is the visitation frequency of state s as calculated from the expert demonstrations $\mathcal{D}$, $\mathbf{1}$ is an indicator function, and $\mu_{\pi_{\psi}}(s)$ is the visitation frequency of state s as obtained from trajectory samples $\tau_{\pi_{\psi}}$ drawn from policy π_{ψ}. $R_{\boldsymbol{\theta}}(\phi(s))$ is the reward obtained by state s according to the current reward network parameterized by $\boldsymbol{\theta}$.

Intuitively, both MaxEnt IRL and MEDIRL are feature matching algorithms which define the optimal reward function R^* as one which induces an optimal policy π^* to, on average, traverse trajectories with identical expected feature counts as the experts, i.e. $\mu_{\mathcal{D}}(s) = \mu_{\pi_{\psi}}(s), \forall s \in \mathcal{S}$. In practice, neither an optimal policy nor the optimal reward is available but a near optimal solution for both can be obtained by following the MEDIRL algorithm [27].

Algorithm 1: Deterministic Maximum Entropy IRL

Input : D, γ, p_0

$\boldsymbol{\theta}, \boldsymbol{\psi} \leftarrow \boldsymbol{\theta}_0, \boldsymbol{\psi}_0$ ▷ Initialize parameters

for $s \in \mathcal{D}$ **do**

  $\mu_{\mathcal{D}}(s) \leftarrow \sum_{\tau \in \mathcal{D}} \sum_{s_t \in \tau} \mathbf{1}_{s=s_t}$ ▷ by (4)

for $m \leftarrow 1$ **to** M **do**

  $\pi_{\boldsymbol{\psi}}^m \leftarrow \texttt{SolveMDP}(R_{\boldsymbol{\theta}}^m, \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma)$

  for $s \in \tau_{\pi_{\boldsymbol{\psi}}}$ **do**

    $\mu_{\pi_{\boldsymbol{\psi}}}(s) \leftarrow \sum_{\tau \in \tau_{\pi_{\boldsymbol{\psi}}}} \sum_{s_t \in \tau} \mathbf{1}_{s=s_t}$ ▷ by (5)

  $\frac{\partial L}{\partial \boldsymbol{\theta}^m} = \texttt{Backprop}\ (\boldsymbol{\theta}^m, \mu_{\mathcal{D}} - \mu_{\pi_{\boldsymbol{\psi}}})$

  $\boldsymbol{\theta}^{m+1} \leftarrow \texttt{UpdateWeight}\ (\frac{\partial L}{\partial \boldsymbol{\theta}^m}, \boldsymbol{\theta}^m)$

Output: optimal parameters θ, ψ

IV. Our Approach

A. Reward Learning

We use an approach based on MEDIRL, as described in section III, to learn policies capable of navigating in a densely crowded environment. Prior works [17], [27] compute the gradient term (3) by directly computing the policy SVF $\mu_{\pi_{\psi}}$ using dynamic programming, which is both expensive and requires a model of the world to accomplish. Recent work [29], [30], [31] derive sampling based approximations to the SVF term which allows for model-free learning of the reward function given a stochastic environment and policy.

In this work, we assume deterministic transition dynamics and a deterministic policy. While a deterministic setup (transition dynamics and policy) limits the extent of state-space exploration as compared to a its probabilistic counterpart, it allows us to greatly simplify the approximation of the policy SVF, which can now be calculated using deterministic policy rollouts from every starting position.

$$\mu_{\pi_{\psi}}(s_i) = \sum_{\tau \in \tau_{\pi_{\psi}}} \sum_{s_t \in \tau} \mathbf{1}_{s_i=s_t} \tag{5}$$

where $\tau_{\pi_{\psi}}$ are deterministic policy rollouts generated using policy π_{ψ}. Since only a single sample per pedestrian (initial scene configuration) is needed, this approximation requires significantly less samples than stochastic importance sampling based methods which require many stochastic samples from each starting position. While this approximation might seem limiting, we show in our experiments that it generalizes well to unknown datasets.

We refer to this approximate algorithm as deterministic MEDIRL and summarize our approach in algorithm 1.

Additionally, algorithm 1 differs from MEDIRL in that it does not necessarily use approximate value iteration to solve for the optimal policy. In this work, we solve the MDP using A2C [32], however, the use of any specific RL algorithm is not required.

B. Feature Extraction

In this section, we describe our proposed risk-based feature representation. We assume the following information is available: the agent position $\mathbf{p}_a$ and velocity $\mathbf{v}_a$, the goal position $\mathbf{p}_g$, and the positions and velocities of all pedestrians p within a certain radius r, denoted $\mathbf{p}_p, \mathbf{v}_p$ respectively.

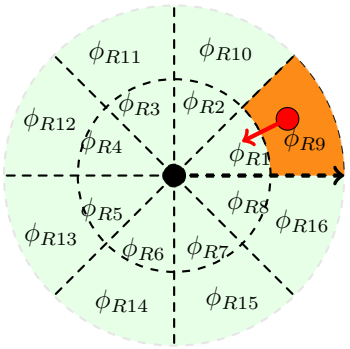


Fig. 1: Spatial bin layout for risk features. The black disk and arrow in the center represent the agent and its orientation respectively, while the red disk and arrow represent a pedestrian and its orientation in bin ϕ_{R9}. Since the pedestrian is on a collision course, the corresponding bin is set to high risk.

1) Risk Features: The agent requires information about the surrounding pedestrians and their velocities to adequately learn navigation strategies. Fahad et al. [25] augment the social affinity map (SAM) features [26] with average pedestrian speed and orientation information, segmenting the agent's surroundings into spatial bins from which the aforementioned attributes are computed. In a comprehensive comparison of various features, Vasquez et al. [24] make use of pedestrian speed and orientation to compute features.

In this paper, we similarly segment the agent's surrounding into spatial bins. Seeking to exploit the human tendency to focus on nearby entities especially ones that might come closer in the near future [33], we propose *risk features* which aims to capture the collision risk posed by pedestrians within each bin. To form the bins, two concentric circles of radii r_1 and r_2 around the agent are each split into eight equally divided spatial bins, as seen in Figure 1. The bins are oriented relative to the agent's orientation. Each spatial bin is then given a one-hot vector $\boldsymbol{\phi}_{Ri} = [\phi_{\text{high}}, \phi_{\text{medium}}, \phi_{\text{low}}]$ risk rating based on the highest risk posed by pedestrians inside the spatial bin based

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.3048657, IEEE Robotics and Automation Letters

on the computed quantity

$$\text{apparent safety} = \tan(\alpha_{\text{risk}}) \, \|\mathbf{p}_a - \mathbf{p}_p\| \tag{6}$$

where α_{risk} is the angle between the relative position vector $\mathbf{p}_{ap} = \mathbf{p}_a - \mathbf{p}_p$ and the relative orientation vector $\mathbf{o}_{ap} = \frac{\mathbf{v}_p}{\|\mathbf{v}_p\|} - \frac{\mathbf{v}_a}{\|\mathbf{v}_a\|}$. If apparent safety is lower than some predefined threshold and $\alpha_{\text{risk}} < 90°$, the bin is considered high risk while while $\alpha_{\text{risk}} > 90°$ is considered low risk as the pedestrian is moving away from the agent, otherwise the risk is considered to be medium. These conditions are summarized in Table I. The complete risk feature vector is obtained by concatenating the risk one-hot vector for each spatial bin resulting in $\boldsymbol{\phi}_R = [\boldsymbol{\phi}_{R1}, \boldsymbol{\phi}_{R2}, \ldots, \boldsymbol{\phi}_{R16}]$.

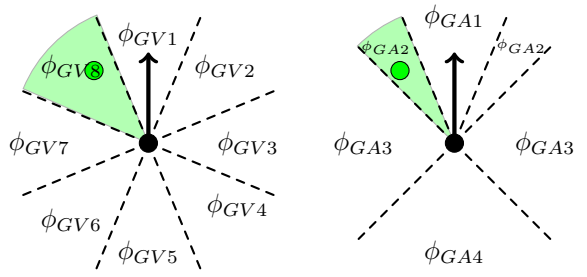
TABLE I: Bin conditions for risk features.

Feature	condition
ϕ_{high}	$\alpha_{\text{risk}} < 90°$ and apparent safety (6) < threshold
ϕ_{low}	$\alpha_{\text{risk}} > 90°$
ϕ_{medium}	otherwise

2) Goal Vector Features: These features, denoted $\boldsymbol{\phi}_{GV}$, encode the direction of the goal relative to the agent's current position. This is done by binning the angle α_{GV} between the agent's position and the goal position $(\mathbf{p}_g - \mathbf{p}_a)$ and the agent's velocity vector $\mathbf{v}_a$ in one of eight equal bins using angle thresholds found in Table II and depicted in Figure 2.

TABLE II: Bin thresholds for goal vector features.

Feature	Threshold
ϕ_{GV1}	$\alpha_{GV} \in \left[\frac{15\pi}{8}, 2\pi\right) \cup [0, 2\pi)$
ϕ_{GV2}	$\alpha_{GV} \in \left[\frac{\pi}{8}, \frac{3\pi}{8}\right)$
ϕ_{GV3}	$\alpha_{GV} \in \left[\frac{3\pi}{8}, \frac{5\pi}{8}\right)$
ϕ_{GV4}	$\alpha_{GV} \in \left[\frac{5\pi}{8}, \frac{7\pi}{8}\right)$
ϕ_{GV5}	$\alpha_{GV} \in \left[\frac{7\pi}{8}, \frac{9\pi}{8}\right)$
ϕ_{GV6}	$\alpha_{GV} \in \left[\frac{9\pi}{8}, \frac{11\pi}{8}\right)$
ϕ_{GV7}	$\alpha_{GV} \in \left[\frac{11\pi}{8}, \frac{13\pi}{8}\right)$
ϕ_{GV8}	$\alpha_{GV} \in \left[\frac{13\pi}{8}, \frac{15\pi}{8}\right)$



(a) Goal vector features. (b) Goal angle features.

Fig. 2: The black disk at the center of the diagram and the black arrow represents the agent's position and heading respectively. The green disk depicts the goal position and is consistent in both the figures. It lies on the ϕ_{GV8}, and ϕ_{GA2} for the goal vector and goal angle feature respectively activating the corresponding bins. Note that the features are relative to the agent's orientation and are thus turning with the agent.

3) Goal Angle Features: These features, denoted $\boldsymbol{\phi}_{GA}$, encode the heading error of the agent with respect to the goal location and are computed by binning the smallest angle α_{GA} between the vector to the goal from the agent's current position $(\mathbf{p}_g - \mathbf{p}_a)$ and the agent's velocity $\mathbf{v}_a$ according to Table III. The bins start narrow in the direction which the agent is facing and gradually widens as to help with agent heading precision. These features supplement the goal vector features (subsubsection IV-B2) with finer-grained goal orienting information which is useful when maneuvering close to the goal.

TABLE III: Bin thresholds for goal angle features.

Feature	Threshold
ϕ_{GA1}	$\alpha_{GA} \in \left[0, \frac{\pi}{8}\right)$
ϕ_{GA2}	$\alpha_{GA} \in \left[\frac{\pi}{8}, \frac{\pi}{4}\right)$
ϕ_{GA3}	$\alpha_{GA} \in \left[\frac{\pi}{4}, \frac{3\pi}{4}\right)$
ϕ_{GA4}	$\alpha_{GA} \in \left[\frac{3\pi}{4}, \pi\right]$

4) Orientation Change: These features, denoted by $\boldsymbol{\phi}_O$, encode the agent's change in orientation. The smallest angle α_{OC} between the agent's previous and current velocity vector is computed and binned according to the thresholds found in Table IV. The progressive widening of the bin is to help agents differentiate between different magnitudes of orientation change thus allowing for finer control when learning.

TABLE IV: Bin thresholds orientation change features.

Feature	Threshold
ϕ_{O1}	$\alpha_{OC} \in \left[0, \frac{\pi}{9}\right)$
ϕ_{O2}	$\alpha_{OC} \in \left[\frac{\pi}{9}, \frac{2\pi}{9}\right)$
ϕ_{O3}	$\alpha_{OC} \in \left[\frac{2\pi}{9}, \frac{3\pi}{9}\right)$
ϕ_{O4}	$\alpha_{OC} \in \left[\frac{2\pi}{9}, \frac{3\pi}{9}\right)$
ϕ_{O5}	$\alpha_{OC} \in \left[\frac{3\pi}{9}, \frac{4\pi}{9}\right)$
ϕ_{O6}	$\alpha_{OC} \in \left[\frac{4\pi}{9}, \pi\right)$

V. Experiments

A. Experimental Setup: The Environment

We train and evaluate our agents in a custom, continuous 2D top-down environment populated by pedestrians whose positions are informed by the expert dataset. A snapshot of our environment can be found in Figure 3.

The following sections detail the important components of our environment.

1) The Agent: The agent is defined by its position, velocity, and radius. The radius is predefined and shared with other pedestrians in the environment, and used in computing collision with any obstacle by checking overlap. At any given simulation time step, the agent accepts a discrete control signal from any external controller (such as a policy), selecting from a preset list of orientation and speed changes in a differential drive style.

2) The Goal: The goal location marks the destination for the agent and is determined according to the environment's initialization behavior detailed in subsubsection V-A5. Overlapping of the goal and the agent terminates an episode.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.3048657, IEEE Robotics and Automation Letters

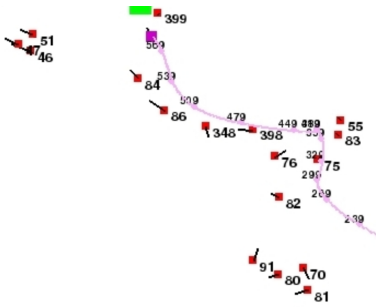


Fig. 3: A snapshot of the environment at a given time frame from the UCY students001 dataset. The black square represents the agent tracing a grey trajectory over time. The pink square and the light-pink trail represent the ground truth pedestrian position and trajectory respectively. The numbers along the traced trajectories denote timestamps. The red squares are pedestrians with their velocity represented by the protruding line, and ID denoted by the associated number.

3) The Pedestrians: Pedestrians are populated onto the environment using an expert dataset. The dictation of the pedestrian movements using a dataset lends to the deterministic nature of the environment. All the pedestrian share the same radius and are the only form of navigational obstacle in the environment. During the training of agents, collisions with pedestrians terminate the training episode while this is not necessarily the case during evaluation depending on the measured metric.

4) Input Space and State: The control scheme for our agent is that of a discrete differential drive, where any controller can choose between a discrete change in orientation of [-30, -20, -10, 0, +10, +20, +30] values in degrees, and a speed change of [-0.4, -0.2, 0, +.02, +0.4] pixels per second. The action space resulting from the Cartesian product of the aforementioned possibilities leads to a discrete action space with 35 distinct possibilities. While the environment returns the location and velocity information of all pedestrians in the scene, the feature representation limits the information of pedestrians to a disk of radius r around the agent for the purpose of decision-making.

5) Initialization Behavior: Whenever the environment is initialized or reset, a pedestrian is replaced by the agent at the same position. The selection of pedestrians is made according to a deterministic queue comprising of all pedestrians, making the agent cycle through all pedestrian initial positions before ever repeating. The final location of the replaced pedestrian determines the goal position of the environment.

B. Training Details

To solve the MDP in the inner loop of training the IRL agents, we make use of A2C [32]. Both our reward and policy networks comprise an input layer whose width is equal to the features being used, followed by a 256 unit wide hidden layer. For the reward network, the output layer uses $\tanh$ activation and returns a real-valued reward. The policy network bifurcates into two heads (outputs): a value head comprising a linear layer that outputs a real number and an action head comprising a linear layer followed by a softmax layer. We use ELU [34] as the activation function for both the input and the hidden layer of both the networks. We use the stochastic gradient descent (SGD) [35] optimizer with a learning rate of 0.0005 for the reward network and the Adam optimizer [36] with a learning rate of 0.0001 for the policy network. For training, we use the densely crowded 'students003' subset of the UCY pedestrian dataset [37] which is comprised of 430 individual pedestrians. The length of the trajectories varies from 53 to 2875 time steps with an average of 406 time steps per trajectory.

For each feature representation considered, we train 21 separate random seeds each for 72 hours measured by wall time. This random seeding, affecting the stochastic optimization components of MEDIRL training, is the only source of experiment randomness as the environment is completely deterministic. From each seed, we select the policy with the smallest SVF difference from the expert.

C. Experiments Conducted

We evaluate the performance of our risk-based feature extractor and deterministic MEDIRL approach by validating against reinforcement learning and potential field [38] baselines. Furthermore, we compare the performance of our risk-based features against similar feature representations found in literature when trained used the deterministic MEDIRL algorithm.

As it is difficult to directly measure what qualifies as socially compliant navigation, we employ a small set of objective metrics motivated by existing literature. While each of these metrics measures an aspect of navigational behavior, they are chosen to represent good socially compliant navigation when considered collectively.

1) Goal reaching rate: This metric measures a fundamental aspect of any navigating agent: the fraction of runs in which the agent succeeds in reaching the goal from its initial position without collisions.
2) Normalized change in orientation: This metric measures the change in the agent's heading angle as a proxy to trajectory smoothness. A low change in orientation is favorable as a highly erratic agent is both disturbing to the surrounding pedestrians and practically challenging to deploy from a control standpoint. This metric is computed for each trajectory by summing the angle changes in each time step and normalizing by trajectory length.
3) Average drift analysis: This metric is similar to the average displacement error (ADE) metric from prior work [39], [40]. It measures the accumulating deviation of the agent's temporal and spatial position with respect to the ground truth trajectories traced by the pedestrians. Unlike previous works, which calculate the divergence over an entire trajectory, we test the divergence over segments of various lengths. This is done by dividing each trajectory into disjoint segments of a given size and calculating the divergence over all the segments. The drift of an agent A, for a given pedestrian p with a trajectory length of T_p for a segment length of S_T, is given by:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.3048657, IEEE Robotics and Automation Letters

$$\epsilon_{A,p,S_T} = \sum_{S_g \in P_{S_T}} \sum_{t=0}^{S_T} \left\| \mathbf{p}^t_{A,S_g} - \mathbf{p}^t_{gt,S_g} \right\| \quad (7)$$

where, P_{s_T} is the set of disjoint segments of length S_T obtained from pedestrian p, and $\mathbf{p}^t_{A,S_g}$ and $\mathbf{p}^t_{gt,S_g}$ representing agent position and the ground truth position at time frame t for segment S_g respectively.

Evaluation is done on two separate datasets: the university 'students001', and the 'Zara' subsets of the UCY pedestrian dataset. The university 'students001' comprises of 410 unique pedestrians, where the length of the trajectories vary from 61 to 5394 with an average of 801 frames per trajectory. The Zara dataset includes a total of 487 unique pedestrians with the length of the trajectories varying from 52 to 5827 and an average of 396 frames per trajectory. In terms of crowd density, university students001 has the highest crowd density followed by university students003 and the Zara. Error bars in the following figures indicate a 95% confidence interval unless stated otherwise.

D. Baseline Evaluation

We compare our proposed method against a potential field implementation [38] and an RL agent trained using the proposed risk features and A2C [32] as baselines with the goal of assessing our proposed agent's ability to successfully reach the destination without collision.

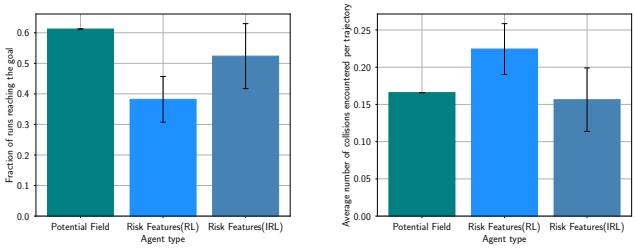
The RL agent is trained with handcrafted rewards and uses the same policy network as described in subsection V-B. The agent receives a reward of $+1$ for reaching the destination, a penalty of -1 for collision with an obstacle, and $0.001 \times l_{\text{step}}$ where l_{step} is the length of the step taken by the agent and is negative if the agent moves away from the goal.

Figure 4 shows that our agent is competitive with the baseline potential field implementation while producing much smoother trajectories as seen in Figure 5. It is interesting to note that while the potential field controller enjoys almost 9% greater success at reaching the goal, the IRL agent surpasses the potential field controller, albeit by a small margin, at collision avoidance. This can be explained by the differing objectives of each method: IRL tries to closely match demonstrated trajectories which do not always take optimal paths towards the goal and thus learns to behave in a more socially compliant manner, while the potential fields agent experiences a direct attracting force towards the goal and is more likely to reach it. The results from the Zara dataset show similar trends.

E. Comparing Different Feature Representations

This section showcases the performance of our risk-based features against existing feature representations in the literature. For comparison against the existing literature, we select feature representations proposed in [25] and [24] with some minor modifications.

From [24], we pick the feature representations $\mathcal{F}_1$ and $\mathcal{F}_3$. While these feature representations were found to be promising, they lack information about the goal state as they



(a) Baseline comparison of the rate of reaching destination on the UCY students001 dataset.

(b) Baseline comparison of number of collisions encountered on the UCY students001 dataset.

Fig. 4: Baseline comparison of on different objective metrics of navigation.

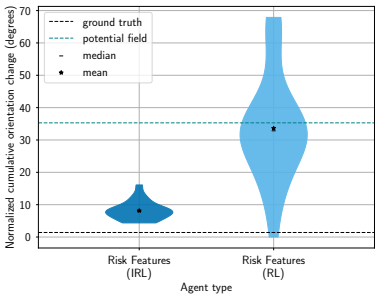
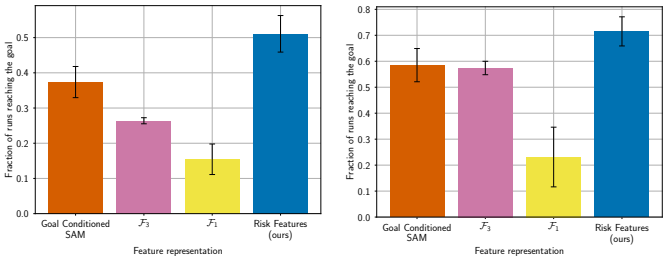


Fig. 5: Baseline comparison of trajectory smoothness as produced by baseline agents. lower normalized cumulative orientation change signifies a smoother trajectory.

were originally used in conjunction with a long-term planning algorithm. To enable a comparison with our method, we append to them the goal-conditioning features used in our proposed feature representation. Additionally, we make the same appendage to the velocity augmented SAM features from [25] as the originally proposed features failed to train a policy due to insufficient goal conditioning. We call this modified version the 'Goal conditioned SAM'.

First, we evaluate the selected feature representations on the 'students001' subset of the UCY dataset. As shown in Figure 6a, we find that our proposed feature representation outperforms other feature representations in successfully reaching the goal. Based on manual inspection of the generated trajec-

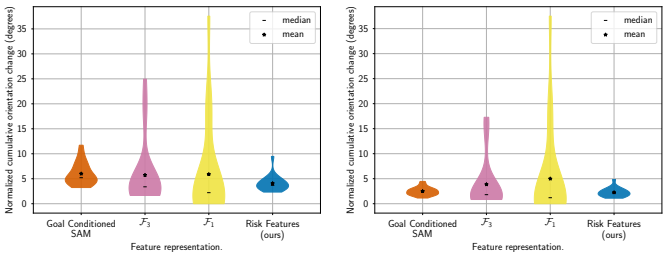


(a) 'students001' subset.

(b) Zara subset.

Fig. 6: Rate of successfully reaching goal for agents trained using deterministic MEDIRL on the UCY 'students001' and Zara subsets [37].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2020.3048657, IEEE Robotics and Automation Letters



(a) 'students001' subset. (b) Zara subset.

Fig. 7: Smoothness of trajectories produced by all agents trained using deterministic MEDIRL. Displayed results are from evaluations on the 'students001' and Zara subsets of the UCY dataset [37].

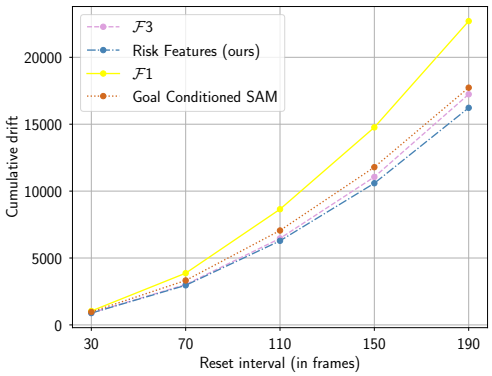


Fig. 8: Drift analysis for agents trained using deterministic MEDIRL on the 'students001' subset of the UCY dataset [37].

tories, we attribute the success of the risk features in large part to good obstacle avoidance. Additionally, our method performs slightly better in the the trajectory smoothness (Figure 7a) and drift metrics (Figure 8). Note that the drift and smoothness metrics, unlike the 'goal reached' metric, do not consider pedestrian collisions, which explains the similar performance of the feature representations that lack the risk-based collision avoidance features allowing them to largely ignore pedestrians, which is an undesirable trait.

Evaluating on the Zara dataset, the results reveal similar performance in successfully reaching the goal (Figure 6b) and in trajectory smoothness (Figure 7b), and an increased drift (Figure 9), which suggests a degree of generalization in the trained agents. The increase in drift can be attributed to the sparsity of the environment, which encourages the agent to opt for straighter paths than what the crowded 'students001' subset would allow for, thus opting for straight paths regardless of what path pedestrians in the same situation would opt to take.

F. Limitations and Future Work

A major drawback of MEDIRL is the expensive retraining of the policy after every reward network update. Adapting MEDIRL to bypass this computationally expensive step by interleaving the training of the reward network and the policy network similar to [29] is a promising avenue of investigation.

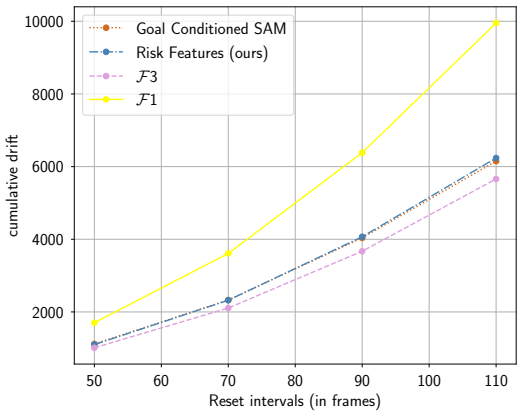


Fig. 9: Drift analysis for agents trained using deterministic MEDIRL. Displayed results are from evaluations on the Zara subset of the UCY dataset [37]. Note that the plots from Risk Features and Goal Conditioned SAM have significant overlap.

Additionally, the task of navigation can be divided into sub-tasks like long-term planning, short-term planning, and collision avoidance. The use of dedicated controllers to handle different sub-tasks in a hierarchical fashion [19] is another area worth exploring.

VI. Conclusion

In this work, we improved on the existing inverse reinforcement learning-based navigation pipelines by introducing a new 'risk-based' feature representation and an inexpensive sampling technique for maximum entropy deep inverse reinforcement learning which allows us to operate in a model-free environment. We demonstrated the efficacy of our algorithm and feature representation by training an agent on a crowded pedestrian dataset and showed it to produce smoother trajectories without sacrificing navigational success rate compared to potential field and reinforcement learning agents. We comprehensively compared our proposed risk features to several other feature representations found in literature and demonstrated an improvement in collision avoidance and increase in conformity to socially-compliant navigation. Furthermore, we demonstrated that the performance of our approach remains consistent when deployed in unknown scenarios.

References

[1] M. Kim, S. Kim, S. Park, M.-T. Choi, M. Kim, and H. Gomaa, "Service robot for the elderly," *IEEE Robotics Automation Magazine*, vol. 16, no. 1, pp. 34–45, Mar. 2009.

[2] C. Jayawardena, I. H. Kuo, U. Unger, A. Igic, R. Wong, C. I. Watson, R. Q. Stafford, E. Broadbent, P. Tiwari, J. Warren, J. Sohn, and B. A. MacDonald, "Deployment of a service robot to help older people," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 5990–5995.

[3] H.-M. Gross, H. Boehme, C. Schroeter, S. Mueller, A. Koenig, E. Einhorn, C. Martin, M. Merten, and A. Bley, "TOOMAS: Interactive Shopping Guide robots in everyday use - final implementation and experiences from long-term field trials," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 2005–2012.

[4] C. P. Gharpure and V. A. Kulyukin, "Robot-assisted shopping for the blind: Issues in spatial cognition and product selection," *Intel Serv Robotics*, vol. 1, no. 3, pp. 237–251, Jul. 2008. [Online]. Available: https://doi.org/10.1007/s11370-008-0020-9

[5] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore, H. Hung, O. A. I. Ramírez, M. Joosse, H. Khambhaita, T. Kucner, B. Leibe, A. J. Lilienthal, T. Linder, M. Lohse, M. Magnusson, B. Okal, L. Palmieri, U. Rafi, M. van Rooij, and L. Zhang, "SPENCER: A Socially Aware Service Robot for Passenger Guidance and Help in Busy Airports," in *Field and Service Robotics: Results of the 10th International Conference*, ser. Springer Tracts in Advanced Robotics, D. S. Wettergreen and T. D. Barfoot, Eds. Cham: Springer International Publishing, 2016, pp. 607–622. [Online]. Available: https://doi.org/10.1007/978-3-319-27702-8_40

[6] N. Mitsunaga, T. Miyashita, H. Ishiguro, K. Kogure, and N. Hagita, "Robovie-IV: A Communication Robot Interacting with People Daily in an Office," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 5066–5072.

[7] H. Huttenrauch and K. Eklundh, "Fetch-and-carry with CERO: Observations from a long-term user study with a service robot," in *11th IEEE International Workshop on Robot and Human Interactive Communication Proceedings*, Sep. 2002, pp. 158–163.

[8] D. Helbing and P. Molnar, "Social Force Model for Pedestrian Dynamics," *Physical Review E*, vol. 51, May 1998.

[9] D. Helbing and A. Johansson, "Pedestrian, Crowd and Evacuation Dynamics," in *Encyclopedia of Complexity and Systems Science*, Apr. 2010, vol. 16, pp. 697–716.

[10] A. Johansson, D. Helbing, and P. K. Shukla, "Specification of the Social Force Pedestrian Model by Evolutionary Adjustment to Video Tracking Data." *Advances in Complex Systems*, vol. 10, pp. 271–288, Dec. 2007.

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

[12] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

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

[14] A. Rudenko, L. Palmieri, and K. O. Arras, "Joint long-term prediction of human motion using a planning-based social force approach," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4571–4577.

[15] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 3931–3936.

[16] A. Y. Ng and S. J. Russell, "Algorithms for Inverse Reinforcement Learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., Jun. 2000, pp. 663–670.

[17] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[18] M. Pfeiffer, U. Schwesinger, H. Sommer, E. Galceran, and R. Siegwart, "Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 2096–2101.

[19] B. Kim and J. Pineau, "Socially Adaptive Path Planning in Human Environments Using Inverse Reinforcement Learning," *Int J of Soc Robotics*, vol. 8, no. 1, pp. 51–66, Jan. 2016. [Online]. Available: https://doi.org/10.1007/s12369-015-0310-2

[20] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-Based Prediction of Trajectories for Socially Compliant Navigation," Jul. 2012.

[21] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, Sep. 2016. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364915619772

[22] H. Kretzschmar, M. Kuderer, and W. Burgard, "Learning to predict trajectories of cooperatively navigating agents," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong, China: IEEE, May 2014, pp. 4015–4020. [Online]. Available: http://ieeexplore.ieee.org/document/6907442/

[23] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid Monte Carlo," *Physics Letters B*, vol. 195, no. 2, pp. 216–222, Sep. 1987. [Online]. Available: http://www.sciencedirect.com/science/article/pii/037026938791197X

[24] D. Vasquez, B. Okal, and K. O. Arras, "Inverse Reinforcement Learning algorithms and features for robot navigation in crowds: An experimental comparison," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014, pp. 1341–1346.

[25] M. Fahad, Z. Chen, and Y. Guo, "Learning How Pedestrians Navigate: A Deep Inverse Reinforcement Learning Approach," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 819–826.

[26] A. Alahi, V. Ramanathan, and L. Fei-Fei, "Socially-Aware Large-Scale Crowd Forecasting," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014, pp. 2211–2218.

[27] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum Entropy Deep Inverse Reinforcement Learning," *arXiv:1507.04888 [cs]*, Mar. 2016. [Online]. Available: http://arxiv.org/abs/1507.04888

[28] E. T. Jaynes, "Information Theory and Statistical Mechanics," *Phys. Rev.*, vol. 106, no. 4, pp. 620–630, May 1957. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.106.620

[29] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*, 2016, pp. 49–58.

[30] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 182–189.

[31] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1331–1336.

[32] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937. [Online]. Available: http://proceedings.mlr.press/v48/mniha16.html

[33] J. Jovancevic-Misic and M. Hayhoe, "Adaptive gaze control in natural environments," *Journal of Neuroscience*, vol. 29, no. 19, pp. 6234–6238, 2009. [Online]. Available: https://www.jneurosci.org/content/29/19/6234

[34] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1511.07289

[35] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. JMLR.org, 2013, p. III–1139–III–1147.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[37] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by Example," *Computer Graphics Forum*, vol. 26, no. 3, pp. 655–664, 2007. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2007.01089.x

[38] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986. [Online]. Available: https://doi.org/10.1177/027836498600500106

[39] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 961–971. [Online]. Available: http://ieeexplore.ieee.org/document/7780479/

[40] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. K. Chandraker, "DESIRE: distant future prediction in dynamic scenes with interacting agents," *CoRR*, vol. abs/1704.04394, 2017. [Online]. Available: http://arxiv.org/abs/1704.04394