Inverse Reinforcement Learning of Pedestrian–Robot Coordination

David Gonon and Aude Billard, *Fellow, IEEE*

***Abstract*—We apply inverse reinforcement learning (IRL) with a novel cost feature to the problem of robot navigation in human crowds. Consistent with prior empirical work on pedestrian behavior, the feature anticipates collisions between agents. We efficiently learn cost functions in continuous space from high-dimensional examples of public crowd motion data, assuming locally optimal examples. We evaluate the accuracy and predictive power of the learned models on test examples that we attempt to reproduce by optimizing the learned cost functions. We show that the predictions of our models outperform a recent related approach from the literature. The learned cost functions are incorporated into an optimal controller for a robotic wheelchair. We evaluate its performance in qualitative experiments where it autonomously travels between pedestrians, which it perceives through an onboard tracking system. The results show that our approach often generates appropriate motion plans that efficiently complement the pedestrians' motions.**

***Index Terms*—Human-aware motion planning, learning from demonstration, path planning for multiple mobile robots or agents.**

I. Introduction

AUTONOMOUS navigation in pedestrian areas has been an active area of research due to applications in smart wheelchairs or delivery robots. In order to navigate smoothly and safely in human crowds, robots need to consider how their own actions affect surrounding humans and reason about the individual costs and benefits of each involved agent. We propose to use a collective cost function that describes how desirable a particular combination of individual trajectories is from the collective point of view of all agents involved. We apply Inverse Reinforcement Learning (IRL) to capture important characteristics of human navigation from empirical data.

We assume a single objective function, whose features depend on each agent, similar to recent approaches [1], [2], [3] to IRL of multi-agent navigation. Similarly, techniques for crowd modeling and simulation [4], [5] optimize a single cost function over all agents' actions. Such formulations are appealing, as they allow to model agents' reciprocal actions, i.e. actions that complement each other.

Traditional IRL assumes that observations are globally optimal, causing their computational cost to scale exponentially with the number of state dimensions. However, a more recent approach [6] overcomes this curse of dimensionality (as shown in [6]) by assuming that examples are only locally optimal. We argue that this framework is particularly well suited to modeling human crowds, since they may evolve in multiple qualitatively different ways with similar probabilities. For example, a pedestrian may pass another pedestrian on the left or on the right side, while trajectories in between the two alternatives should be costly because they lead to a collision. Thus, the two alternatives can be described as local optima which are separated by trajectories with higher costs.

Despite the availability of promising algorithms, the design of appropriate features for the problem of multi-agent navigation IRL remains a challenging and critical step in achieving a cost function that generalizes in a meaningful way and proves useful in real-world situations. Common features to account for collisions include $f = d^{-2}$ [2] and $f = \exp(-d^2/\sigma^2/2)$ [3], where d denotes the distance between agents' centers. As both features are only position-dependent, they do not reflect how human walkers avoid collisions, namely by anticipating and regulating the minimum predicted distance d_{mp} [7] (cf. Fig. 1(a)).

Prior empirical work [4] derived an anticipatory interaction potential E, defined as

$$E := \eta\tau^{-2}\exp(\tau/\tau_o), \tag{1}$$

where τ denotes the time to collision of two given agents, $\tau_o = 3$ s is a time horizon, and $\eta = 1\ \text{s}^2$ may be set arbitrarily. Note that E drops discontinuously to zero when approaching agents change their velocities to avoid each other.

We propose a novel feature $\tilde{E}$ that smoothly approximates E (cf. Fig. 1(c)), as required by the framework for continuous IRL [6]. We train two models incorporating $\tilde{E}$ on real crowd data. Our quantitative evaluation demonstrates more accurate and smoother predictions of pedestrians' trajectories compared to a state-of-the-art alternative [2]. Further, we qualitatively evaluate our approach's performance at controlling a mobile robot navigating in a real crowd.

II. Related Work

Several previous works have used IRL to learn navigation in crowds. Some works [8], [9], [10], [11], [12] use discrete state and action spaces. These approaches operate on grids over

Manuscript received 14 December 2022; accepted 7 June 2023. Date of publication 27 June 2023; date of current version 3 July 2023. This letter was recommended for publication by Associate Editor M. R. Elara and Editor G. Venture upon evaluation of the reviewers' comments. This work was supported by Hasler Foundation, Switzerland. *(Corresponding author: David Gonon.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the Human Research Ethical Committee of EPFL under Application No. HREC-032-2019.

The authors are with the School of Engineering, EPFL, CH-1015 Lausanne, Switzerland (e-mail: david.gonon@epfl.ch; aude.billard@epfl.ch).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3289770, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3289770

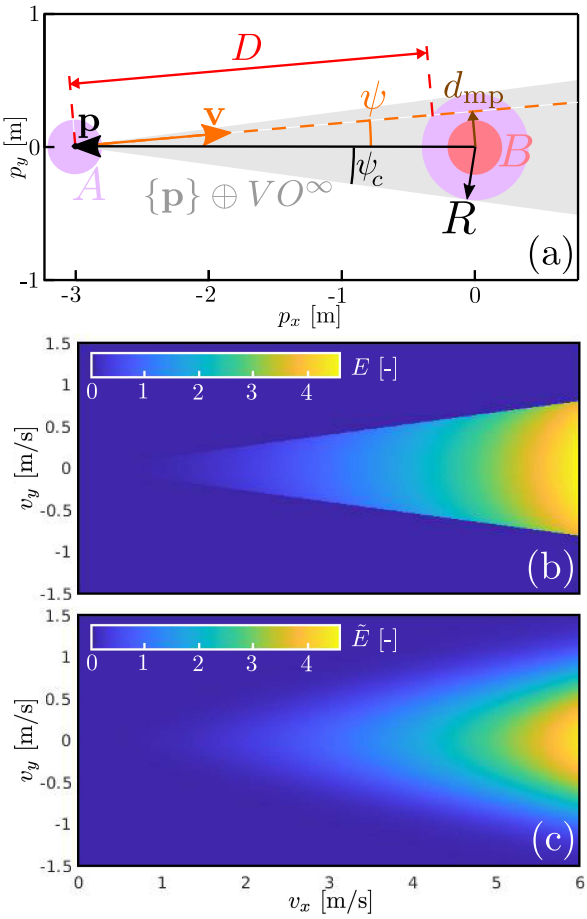

Fig. 1. (a) Two agents A and B are shown at exemplary positions. The cone of colliding relative velocities VO^{∞} is constructed as a function of their relative position and their combined radius R. The minimum predicted distance d_{mp} and the free relative path length D for an exemplary relative velocity $\mathbf{v}$ are also shown. (b) The agents' interaction energy E is shown as a function of their relative velocity $\mathbf{v}$. (c) The approximate interaction energy $\tilde{E}$ of the agents is smooth at the edge of VO^{∞}, in contrast to E.

the entire state space, and thus, they cannot easily incorporate multiple agents due to exponentially increasing computational cost. Instead, they typically consider pedestrians as exogenous inputs which affect the cost/reward features only. While such an approach may yield sensible behavior, it cannot produce a model which reasons about interactions between agents, since the behavior of other agents is treated as a given input to the model.

In contrast, the works [1], [2], [6], [13] have adapted the popular Maximum Entropy (MaxEnt) IRL [14] method to continuous state and action spaces. In the works on continuous spaces, trajectories are either parametrized by the control actions at each discrete time point [3] or as splines [1], [2], [13]. For dealing with the high dimensionality of the space of possible multi-agent trajectories, [3] resort to a local approximation [6] of the exponential policy for MaxEnt IRL, similar to our work. In contrast, [1], [2], [13] simplify the policy by discretizing it into topological variants (who passes on which side of who) and/or use computationally expensive Monte Carlo techniques.

III. Method

A. System Model

For a system comprising n agents, let $\mathbf{p}_i, \mathbf{v}_i, \mathbf{a}_i \in \mathbb{R}^2$ denote the position, velocity, and acceleration of the i-th agent, respectively, where $i \in \{1, 2, \dots n\}$. Let the system's *state*

$$\mathbf{x} := \begin{bmatrix} \mathbf{p}_1^{\mathrm{T}} & \mathbf{p}_2^{\mathrm{T}} & \dots & \mathbf{p}_n^{\mathrm{T}} & \mathbf{v}_1^{\mathrm{T}} & \mathbf{v}_2^{\mathrm{T}} & \dots & \mathbf{v}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2}$$

contain all positions and velocities. The system's *action*

$$\mathbf{u} := \begin{bmatrix} \mathbf{a}_1^{\mathrm{T}} & \mathbf{a}_2^{\mathrm{T}} & \dots & \mathbf{a}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{3}$$

is defined by all accelerations, on the other hand.

The system's transition from one state $\mathbf{x}^{(k)}$ to another state $\mathbf{x}^{(k+1)}$, under an action $\mathbf{u}^{(k+1)}$ over a time step of duration h, is described by the linear dynamic system

$$\mathbf{x}^{(k+1)} = \mathbf{A}\mathbf{x}^{(k)} + \mathbf{B}\mathbf{u}^{(k+1)},$$
$$\mathbf{A} := \begin{bmatrix} \mathbf{I} & h\mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \quad \mathbf{B} := \begin{bmatrix} (h^2/2)\mathbf{I} \\ h\mathbf{I} \end{bmatrix}, \tag{4}$$

where $\mathbf{I} \in \mathbb{R}^{2n \times 2n}$ denotes the identity matrix. The above dynamic model (4) corresponds to applying constant accelerations over a duration h.

A *state–action trajectory* $\mathcal{S} := (\mathcal{X}, \mathcal{U}, \mathcal{T})$ is defined by a state sequence $\mathcal{X} := \{\mathbf{x}^{(k)}\}_{k=0}^{K}$ and an action sequence $\mathcal{U} := \{\mathbf{u}^{(k)}\}_{k=1}^{K}$ satisfying the dynamic model (4) for $0 \leq k < K$ with a uniform time step h, and by a time sequence $\mathcal{T} := \{t^{(k)}\}_{k=0}^{K}$, where each $t^{(k)}$ denotes the time at which the state $\mathbf{x}^{(k)}$ is attained, and $t^{(k+1)} - t^{(k)} = h, \forall k$.

B. Framework for Inverse Reinforcement Learning

We assume that there is a scalar cost function $J(\mathcal{X}, \mathcal{U})$ which reflects the collective objectives of the multi-agent system under consideration, e.g. navigating efficiently and without any collision. For a given cost function J and an initial state $\mathbf{x}_o$, we model the system's behavior over $K+1$ uniform time steps as a local optimum of the optimal control problem

$$\begin{aligned} \min_{\mathcal{X},\mathcal{U}} \; & J(\mathcal{X}, \mathcal{U}) \\ \text{s.t. } & \mathbf{x}^{(k)} = \mathbf{A}\mathbf{x}^{(k-1)} + \mathbf{B}\mathbf{u}^{(k)}, \; k = 1, \dots K \\ & \mathbf{x}^{(0)} = \mathbf{x}_o. \end{aligned} \tag{5}$$

Given a set $\mathcal{E} := \{\mathcal{S}_l\}_{l=1}^{L}$ of L examples $\mathcal{S}_l$ (multi-agent trajectories) recorded from a human crowd, we aim at inferring a meaningful cost function J, such that the examples $\mathcal{S}_l$ are approximated by local optima of (5) with J.

We aim at linear estimation of J in the form

$$J(\mathcal{X}, \mathcal{U}; \mathbf{w}) := \sum_{k=1}^{K} \mathbf{w}^{\mathrm{T}} \mathbf{f}\left(\mathbf{x}^{(k)}, \mathbf{u}^{(k)}\right), \tag{6}$$

where the *feature vector* $\mathbf{f}(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^q$ is specified a priori to describe quantities of interest, and where $\mathbf{w} \in \mathbb{R}^q$ is the unknown vector of the features' weights. We use a linear technique as we hypothesize that our features describe non-interacting

cost components. In the original maximum entropy (MaxEnt) framework [14], the weights are learned by maximizing the examples' likelihood under a policy which assigns a probability density to an action sequence in proportion to the exponential of its reward (i.e. negative cost). We adopt the framework [6], where this policy is approximated as a Gaussian in $\mathcal{U}$ around the examples, thereby relaxing the original MaxEnt framework's assumption of their global optimality to only local optimality.

C. Cost Features for Cooperative Navigation

For simplicity, we use only four features, which we consider as the most essential ones for multi-agent navigation, describing individual effort and goal-directed motion, and pairwise distance and anticipatory collision avoidance, respectively. Individual features are of the form $f := (1/n)\sum_{i=1}^{n} f_i$, where the f_i denote agents' individual contributions, and pairwise features are of the form $f := (1/n)\sum_{i=1}^{n}\sum_{j=i+1}^{n} f_{ij}$, where the f_{ij} denote agents' pairwise contributions.

1) Individual Feature Contributions: An agent's effort can be described by the feature contribution

$$f_i^{(\mathrm{a2})} = |\mathbf{a}_i|^2/2 \tag{7}$$

where $\mathbf{a}_i$ denotes the agent's acceleration. Alternatively, let

$$f_i^{(\mathrm{a1})} = |\mathbf{a}_i| + \left(\log(1+\exp(-2\lambda|\mathbf{a}_i|)) - \log(2)\right)/\lambda \tag{8}$$

define a feature contribution smoothly approximating $|\mathbf{a}_i|$, where $\lambda > 0$ controls the function's sharpness at the origin.

An agent's deviation from its desired motion is described by the feature contribution

$$f_i^{(\mathrm{v})} = |\mathbf{v}_i - \breve{\mathbf{v}}_i|^2/2 \tag{9}$$

where $\mathbf{v}_i$ and $\breve{\mathbf{v}}_i$ denote the agent's actual and desired velocity, respectively.

2) Pairwise Feature Contributions: For a pair of agents with respective indices i and j, let $\mathbf{p}_{ij} := \mathbf{p}_i - \mathbf{p}_j$ and $\mathbf{v}_{ij} := \mathbf{v}_i - \mathbf{v}_j$ denote their relative position and velocity, respectively. We employ a simple pairwise feature contribution commonly found in related work, namely a gaussian function of both agents' distance [3]

$$f_{ij}^{(\mathrm{c})} = \exp\left(-|\mathbf{p}_{ij}|^2/(2\sigma^2)\right), \tag{10}$$

where $\sigma > 0$ denotes the activation radius.

Additionally, we define the pairwise feature contribution

$$f_{ij}^{(\mathrm{e})} := \tilde{E}\left(\mathbf{p}_{ij}, \mathbf{v}_{ij}\right) := \eta\, g(z\left(\mathbf{p}_{ij}, \mathbf{v}_{ij}\right))/\tilde{\tau}^2\left(\mathbf{p}_{ij}, \mathbf{v}_{ij}\right) \tag{11}$$

to smoothly approximate the interaction potential (1), with $\tilde{\tau}$ approximating τ. The sigmoid $g(z) := (1+\exp(-sz))^{-1}$ with $s > 0$ is activated for $\mathbf{v} \in VO^{\infty}$, since z is defined as

$$z := -\mathbf{p}^{\mathrm{T}}\mathbf{v} - \frac{|\mathbf{p}|^2|\mathbf{v}|}{\sqrt{|\mathbf{p}|^2+R^2}} = |\mathbf{p}|\,|\mathbf{v}|\left(\cos(\psi) - \cos(\psi_c)\right),$$

with $\psi := \angle(-\mathbf{p}, \mathbf{v})$, and ψ_c denoting half the opening angle of the cone VO^{∞} (cf. Fig. 1(a)), omitting the subscripts ij for brevity. Thus, $z = 0$ holds when the relative velocity is at the edge of VO^{∞}, which becomes the activation threshold.

Let $\tilde{\tau} := \tilde{D}/|\mathbf{v}|$, where $\tilde{D}$ approximates the distance D that the relative position $\mathbf{p}$ can travel along the relative velocity $\mathbf{v}$ before entering the disk of radius R centered at the origin (cf. Fig. 1(a)). We define $\tilde{D}^2$ as

$$\tilde{D}^2 := \varepsilon_1 + \left(|\mathbf{p}| - R\right)\left(|\mathbf{p}| - R + 2d_{\mathrm{mp}}^2/R\right) \tag{12}$$

$$d_{\mathrm{mp}}^2 := |\mathbf{p}|^2 - \frac{\left(\mathbf{p}^{\mathrm{T}}\mathbf{v}\right)^2}{|\mathbf{v}|^2+\varepsilon_2} \tag{13}$$

where $\varepsilon_2 > 0$ prevents division by zero, and $d_{\mathrm{mp}} \geq 0$ denotes the minimum predicted distance (cf. Fig. 1(a)). Disregarding ε_1, it holds $\tilde{D}^2 = D^2$, for $\psi = 0$ or $\psi = \psi_c$. We set $\varepsilon_1 := 0.22R^2$ to ensure that $\tilde{D}^2 > 0$ always holds. To see this for the only non-trivial case $|\mathbf{p}| < R$, we replace d_{mp}^2 in (12) by its upper bound $|\mathbf{p}|^2$ and analyze the resulting cubic function of $|\mathbf{p}|$, to find that its minimum on $0 < |\mathbf{p}| < \infty$ is positive. The Fig. 1(c) shows the resulting approximation for the same configuration as for Fig. 1(b), where we set $s = 10$ and $\varepsilon_2 = 0.01\ \mathrm{m}^2/\mathrm{s}^2$.

3) Feature Vectors and Normalization: We define

$$\mathbf{f}_{(a^2)} := \begin{bmatrix} f^{(\mathrm{a2})} & f^{(\mathrm{v})} & f^{(\mathrm{c})} & f^{(\mathrm{e})} \end{bmatrix}^{\mathrm{T}}, \tag{14}$$

$$\mathbf{f}_{(a^1)} := \begin{bmatrix} f^{(\mathrm{a1})} & f^{(\mathrm{v})} & f^{(\mathrm{c})} & f^{(\mathrm{e})} \end{bmatrix}^{\mathrm{T}}, \tag{15}$$

as two alternative feature vectors, which differ only in the first feature $f^{(\mathrm{a2})}$ or $f^{(\mathrm{a1})}$, which sum accelerations' squares or smooth magnitudes, respectively.

We normalize features and weights to reveal their relative importance. Let $\tilde{f}(\mathcal{E}) := P_{80}^{\mathcal{E}}\{f\}$ define the *normalizer* for any feature f as its 80-th percentile over a sample set $\mathcal{E}$. Then, for any feature f and the respective weight w, we define the corresponding *normalized feature* as $\phi := f/\tilde{f}$ and the corresponding *normalized weight* as $\theta := w\tilde{f}$, such that the corresponding term of the cost function can be written equally as $wf = \theta\phi$.

IV. Learning Experiments

We apply our IRL approach to real-world multi-agent trajectory data from the public datasets DIAMOR [15] and ETH [16]. Using $\mathbf{f}_{(a^2)}$ and $\mathbf{f}_{(a^1)}$ as feature vectors, we learn two respective cost functions $J_{(a^2)}$ and $J_{(a^1)}$, which model multiple agents' navigation choices, and we evaluate their accuracy on both datasets. The datasets DIAMOR and ETH have been recorded in a shopping mall and on a university campus, respectively, and include wheelchair users (DIAMOR) and pedestrians (both). Our code is available at https://github.com/epfl-lasa/navioc.

A. Data Pre-Processing

We obtain multi-agent examples of individual trajectory segments and associated desired velocities as follows.

1) Fitting Individual Trajectories: The (noisy) trajectory data of an agent, as given originally in a dataset, is denoted as $\{\bar{t}^{(i)}, \bar{\mathbf{p}}^{(i)}\}_{i=1}^{M}$, consisting of M sequential time values $\bar{t}^{(i)}$ and corresponding positions $\bar{\mathbf{p}}^{(i)}$. A state–action trajectory $\mathcal{S}$, describing the single agent at hand ($n = 1$) according to Section III-A, is fit to the data by solving the following quadratic

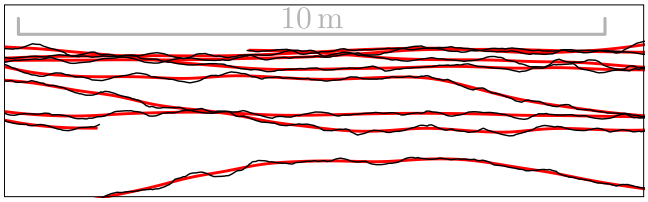

Fig. 2. Showing original tracks from the dataset DIAMOR (black) and the corresponding fit trajectories (red).

program. We jointly optimize $\mathcal{X}, \mathcal{U}$ and a sequence of interpolated positions $\hat{\mathcal{P}} := \{\hat{\mathbf{p}}^{(i)}\}_{i=1}^{M}$ in the problem

$$\begin{aligned} \min_{\mathcal{X},\mathcal{U},\hat{\mathcal{P}}} \; & \sum_{i=1}^{M} \left| \hat{\mathbf{p}}^{(i)} - \bar{\mathbf{p}}^{(i)} \right|^2 + \alpha \sum_{k=1}^{K} \left| \mathbf{u}^{(k)} \right|^2 \\ \text{s.t. } & \mathbf{x}^{(k+1)} = \mathbf{A}\mathbf{x}^{(k)} + \mathbf{B}\mathbf{u}^{(k+1)}, \; k = 0, \dots K-1 \\ & \hat{\mathbf{p}}^{(i)} = \begin{bmatrix} \mathbf{I} & h_i \mathbf{I} \end{bmatrix} \mathbf{x}^{(\kappa(i))} + \frac{h_i^2}{2} \mathbf{u}^{(\kappa(i)+1)}, \; i = 1, \dots M, \end{aligned} \tag{16}$$

where $\alpha > 0$ is a regularizer controlling smoothness, $\kappa(i) := \text{floor}((\bar{t}^{(i)} - t^{(0)})/h)$ indicates for each $\bar{t}^{(i)}$ the temporally closest foregoing state/action extrapolating $\hat{\mathbf{p}}^{(i)}$, and $h_i := \bar{t}^{(i)} - t^{(0)} - h\kappa(i)$ denotes the extrapolation time step.

Trajectories longer than 100 h are fit in an incremental fashion, in order to keep the dimensionality of the optimization problem (16) at an efficiently tractable level. Namely, we sequentially fit multiple temporally overlapping state–action trajectories of 100 time steps each to sub-sequences of the original trajectory's data, where additional constraints enforce continuity at stitching points. The Fig. 2 shows original tracks from DIAMOR, and the corresponding fit trajectories for $h = 0.05$ s and $\alpha = 0.01\text{s}^4$.

2) Estimating Desired Velocities: We assume each agent to have a constant desired speed $\breve{v}$, and we estimate $\breve{v}$ as the mode of the agent's speed histogram, disregarding speeds below a threshold $v_{\min} = 0.3$ m/s.

The desired orientation of any agent is assumed to be time-varying and denoted here as $\breve{\varphi}^{(k)}$. For DIAMOR, any agent's $\breve{\varphi}^{(k)}$ is estimated to be left or right dependent on the actual velocity $v_x^{(k)}$ being negative or positive, respectively. For ETH, we first identify any agent's goal among the dataset's four designated goals as the one which the agent is facing mostly during the later half of its trajectory. Secondly, $\breve{\varphi}^{(k)}$ is defined as the orientation of the vector from the agent to the goal at each instant k. Finally, for each agent, the desired velocity is defined as $\breve{\mathbf{v}}^{(k)} := \breve{v}[\cos\breve{\varphi}^{(k)} \sin\breve{\varphi}^{(k)}]^{\mathsf{T}}$, if $|\mathbf{v}^{(k)}| > v_{\min}$, or as zero, else. Thus, for each agent, a sequence $\breve{\mathcal{V}} := \{\breve{\mathbf{v}}^{(k)}\}_{k=0}^{K}$ of desired velocities is obtained.

3) Sampling Multi-Agent Trajectories: The time domain covered by individual trajectory fits (per dataset) is divided into adjacent intervals of uniform duration $T = 4.8$ s, by convention [2]. For each such interval, the parts of those trajectories which are defined on the entire interval are aggregated into a multi-agent state-action trajectory. The agents' desired velocities are taken as constant over each interval and equal to their values at the interval's initial time. For DIAMOR, trajectory parts that do not overlap with the area of interest in the corridor (shown partially in Fig. 2) are disregarded, whereas for ETH, all agents are considered.

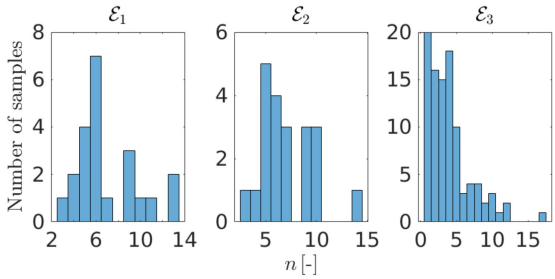

Fig. 3. For the three datasets $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$, histograms of the number of agents n in a given sample are shown.

TABLE I
FEATURES' NORMALIZERS $\tilde{F}$ AND LEARNED NORMALIZED WEIGHTS θ

f	$10\tilde{f}$	$\theta_{(a^2)}$	$\theta_{(a^1)}$	Parameters
$f^{(a2)}$	0.185 m^2/s^4	1.0000	n.a.	–
$f^{(a1)}$	1.039 m/s^2	n.a.	1.0000	λ=10 s^2/m
$f^{(v)}$	0.131 m^2/s^2	0.0385	7.5471	–
$f^{(c)}$	0.635	0.0007	0.0976	σ=0.5 m
$f^{(e)}$	0.021	0.0137	0.6593	η=1 s^2, s=25s^2/m^2 R=0.4 m, ε_2=0.01 m^2/s^2

4) Defining Training and Test Sets: We inspect each multi-agent sample obtained from DIAMOR and discard those which contain agents whose desired motion does not seem aligned with the x-direction, e.g. because they traverse the corridor along the y-direction or diagonally due to intrinsic motives and not to perform avoidance maneuvers. To group the remaining samples from DIAMOR in a training set $\mathcal{E}_1$ and a test set $\mathcal{E}_2$, they are sorted in their temporal order and then assigned in an alternating fashion to $\mathcal{E}_1$ or $\mathcal{E}_2$, to ensure that the two sets reflect similar crowd conditions. In contrast, all multi-agent samples obtained from ETH are grouped in the set $\mathcal{E}_3$, without performing any manual selection. The Fig. 3 shows the histogram of the number of agents n per sample for each of the three datasets. Their respective sizes are $|\mathcal{E}_1| = 22$, $|\mathcal{E}_2| = 21$, and $|\mathcal{E}_3| = 99$.

B. Training

We have adapted the publicly available software package[1] by [6] to learn from examples of varying dimensionality (due to varying numbers of agents) and complemented the package with our system and feature definitions. Training on the dataset $\mathcal{E}_1$ takes around 4 and 7 minutes for the feature vectors $\mathbf{f}_{(a^2)}$ and $\mathbf{f}_{(a^1)}$, respectively.

For any feature f, the Table I reports the normalizer $\tilde{f}(\mathcal{E}_1 \cup \mathcal{E}_2)$, which has been computed with respect to all the selected samples from DIAMOR, and the learned normalized weights $\theta_{(a^2)}$ and $\theta_{(a^1)}$ for the feature vectors $\mathbf{f}_{(a^2)}$ and $\mathbf{f}_{(a^1)}$, respectively. Weight vectors have been scaled such that the acceleration feature's normalized weight equals one.

[1] https://graphics.stanford.edu/projects/cioc/

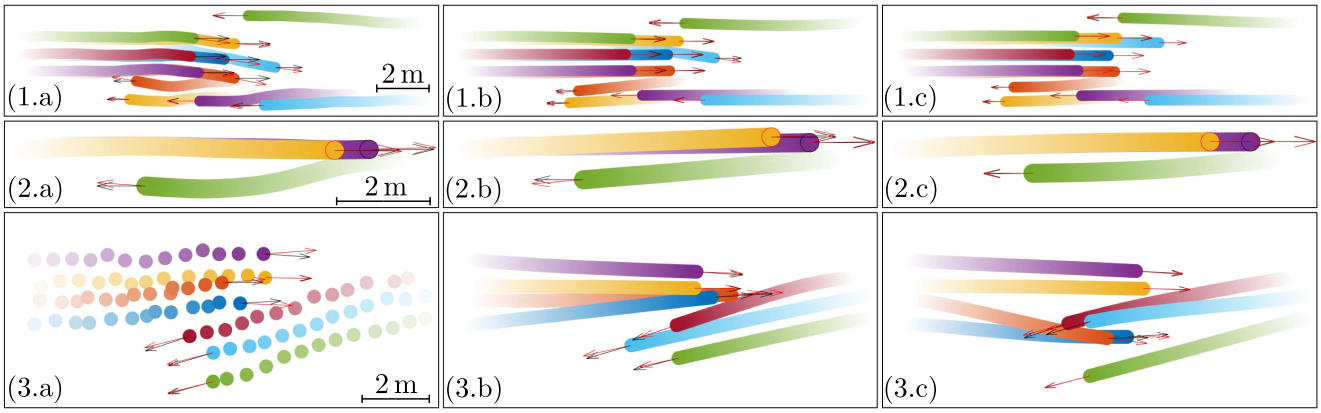

Fig. 4. Plots show multi-agent trajectories from the example sets (left column) and their optimized counter parts generated from our two models $J_{(a^2)}$ (middle column) and $J_{(a^1)}$ (right column). Agents are depicted by their bounding circles of diameter R centered at their positions over time, such that circles for later instants are drawn on top of earlier circles and in more saturated colors. Red arrows indicate the most recent velocity, whereas black arrows indicate the assumed desired velocity for each agent. The examples in (1.a, 2.a, 3.a) belong to the sets $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$, respectively.

1) Choice of Hyper Parameters: The weights θ are initialized as random noise with zero mean and standard deviation 0.01, except for the acceleration weight, which needs to be negative for convergence and thus is set to -1. We observe that for too small α, only a negligible weight is learned for the interaction feature $f^{(\mathrm{e})}$. The sharpness s for the feature $f_{(a^1)}$ strongly affects the learned behavior, as for both very small or large s, acceleration is penalized more strongly relative to the velocity error, such that agents slowly adopt their target velocity.

C. Evaluation

To evaluate the learned models on a given example, we optimize a multi-agent trajectory according to (5) under the learned cost functions. For examples from $\mathcal{E}_1$ and $\mathcal{E}_2$, each local optimization is initialized by the respective example's complete actual trajectory, and $\mathbf{x}_o$ is specified as the example's initial state obtained by non-causal fitting according to Section IV-A1. In contrast, for $\mathcal{E}_3$, each local optimization is initialized with $\mathcal{U} \equiv \mathbf{0}$, and $\mathbf{x}_o$ is specified as the causal estimate of each agent's state at time zero which is provided in the ETH dataset. We denote the euclidean distance between the optimized positions and ground truth as the modeling error e_{model}, for $\mathcal{E}_1$ and $\mathcal{E}_2$, and as the prediction error e_{predict}, for $\mathcal{E}_3$. As a baseline, we also evaluate on all example sets a model assuming constant velocity (CV) after starting from a given initial state $\mathbf{x}_o$.

Examples of actual and reproduced trajectories are shown in Fig. 4. For both our models and CV, Fig. 5 plots the modeling error e_{model} on the DIAMOR training and test sets $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively, as a function of time Δt after the initial time of the trajectories considered. The Fig. 6 compares the prediction error e_{predict} on the test set $\mathcal{E}_3$ from ETH for our models, CV, and the corresponding results reported in [2] for their IRL approach.

The Table II reports the number of collisions for all example sets, comparing ground truth with the learned models and CV. Here, any isolated period in which the distance between two agents is below their combined radius $R = 0.4$ m counts as a single collision. The Table II shows that our models generate very few ($J_{(a^2)}$) or zero ($J_{(a^1)}$) collisions.

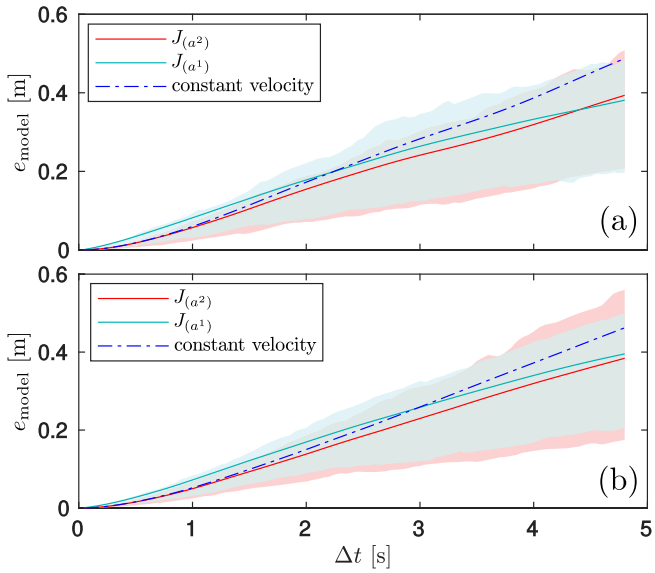

Fig. 5. Modeling error is calculated as the distance between agents' observed and re-optimized positions as a function of time Δt from the trajectory's initial time. The modeling error's sample mean and the interval between its lower and upper quartiles (shaded area) are shown for our two models $J_{(a^2)}$ and $J_{(a^1)}$, as well as the sample mean for a baseline model assuming constant velocity. They are evaluated on: (a) The training set $\mathcal{E}_1$. (b) On the test set $\mathcal{E}_2$, which are both sampled from the dataset DIAMOR [15].

TABLE II
NUMBER OF COLLISIONS

Dataset	Ground truth	$J_{(a^2)}$	$J_{(a^1)}$	CV
$\mathcal{E}_1$	0	0	0	5
$\mathcal{E}_2$	0	0	0	5
$\mathcal{E}_3$	2	4	0	38

D. Discussion

It can be seen from Fig. 5 that both learned models provide a more accurate description of pedestrians' trajectories than the baseline model CV assuming constant velocity, and that good generalization from the training to the test set is achieved for

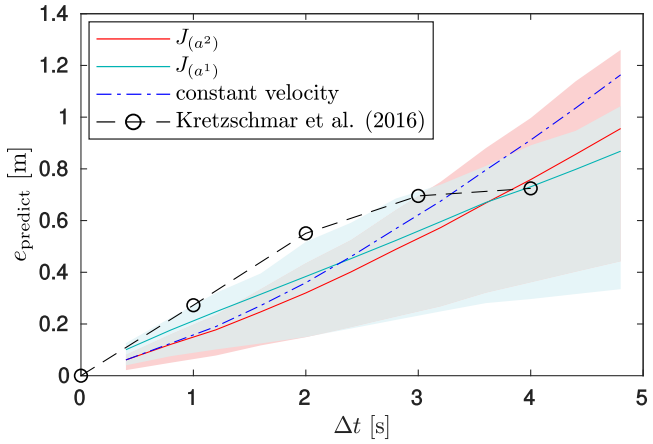

Fig. 6. Prediction error is calculated as the distance between agents' observed and predicted positions as a function of time Δt from the instant at which the prediction is issued. The plot shows the prediction error's sample mean and the interval between its lower and upper quartiles (shaded area) for our two models $J_{(a^2)}$ and $J_{(a^1)}$, as well as the sample mean reported in [2] for their IRL approach and the sample mean for a baseline assuming constant velocity, all for the test set $\mathcal{E}_3$ sampled from the dataset ETH [16].

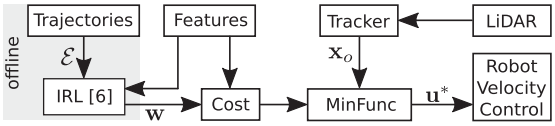

Fig. 7. Cost function, which is parametrized by weights $\mathbf{w}$, is learned offline via IRL from a set $\mathcal{E}$ of multi-pedestrian trajectories. The robot navigates on-line by seeking collective actions $\mathbf{u}^*$ which optimize the learned cost, given the robots' and tracked pedestrians' estimated state $\mathbf{x}_o$.

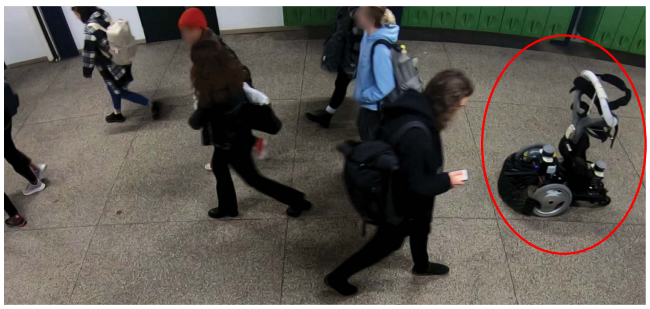

Fig. 8. In our experiments, the robot Qolo (encircled) drives in a busy corridor on EPFL campus by optimizing the learned objective $J_{(a^1)}$ on-line.

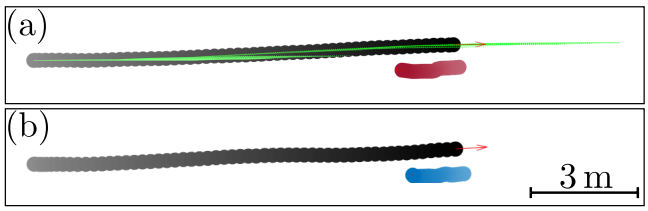

Fig. 9. Robot's estimated trajectory is shown (black circles) as it avoids a static obstacle (red/blue), which it detects and tracks using its on-board tracking system. Darker colors indicate later points in time. The static obstacle's apparent motion is due to errors in the robot's self-localization. (a) The robot follows a reference trajectory (green) planned on-line by our model $J_{(a^1)}$. (b) The robot uses a Velocity Obstacle approach based on [19] to determine its velocity command, where the time horizon is set to $\tau_{vo} = 7$ s.

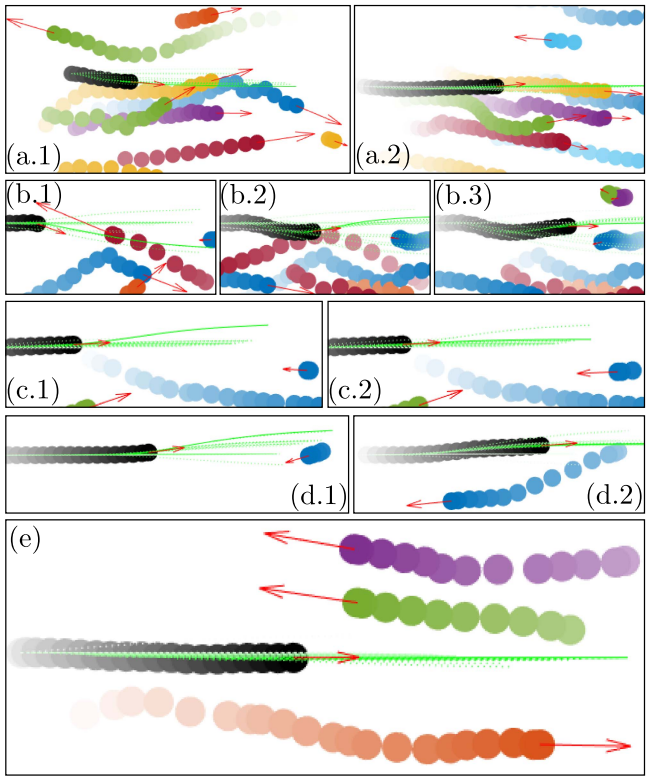

Fig. 10. Exemplary motions, recorded during our experiments with the robot Qolo on EPFL Campus, are visualized. The robot (black) and pedestrians (various colors) are depicted by circles of diameter 0.4 m, such that circles for later instants are drawn on top of earlier circles and in more saturated colors. Red arrows indicate the most recent velocity. Similarly, the robot's recently planned paths are drawn (green, dotted), where more saturated colors indicate later plans, and its current plan is drawn as well (green, solid). Each row of subfigures depicts a different situation, where time increases from left to right. (a.1) The robot starts to move and immediately avoids a pedestrian (green) heading the other way. (a.2) The robot has joined a lane of pedestrians, and one of them (green) steps away from the robot. (b.1) The robot attempts to avoid a pedestrian (red) on the less efficient side. (b.2) The pedestrian (red) has avoided the robot on the more efficient side, and the robot attempts to avoid a standing pedestrian (blue). (b.3) The robot attempts to pass in between the two standing pedestrians (blue and green/purple) and is about to be stopped by the supervisor. (c.1-2) At first, the robot plans to avoid a pedestrian (green), but the pedestrian changes course faster, such that the robot's plan changes back to a straight path. (d.1-2) The robot and a pedestrian (the supervisor) reciprocally avoid each other. (e) A pedestrian (red) overtakes the standing robot and then changes direction away from the robot as it starts moving; subsequently, the robot shows slight adaptation to approaching pedestrians (green, purple).

both models $J_{(a^2)}$ and $J_{(a^1)}$. In terms of mean modeling error, $J_{(a^2)}$ outperforms $J_{(a^1)}$ especially at low Δt. Similarly, Fig. 6 shows that our models generate more accurate predictions than the baseline. For $\Delta t < 4$ s, our models' predictions are also more accurate than those by [2]. Comparing $J_{(a^2)}$ and $J_{(a^1)}$, it is again visible that for low Δt, predictions by $J_{(a^2)}$ are more accurate on average than those by $J_{(a^1)}$, whereas for high Δt, the opposite can be said. For $\Delta t > 4$ s, it seems that the approach by [2] would yield the most accurate predictions, considering the trend in the error curve (which remains speculation, however, due to missing data in this regime).

Regarding Fig. 6, it is noteworthy that the error curves' sense of curvature differs across models, namely, the curves for $J_{(a^2)}$

and CV bend upwards, whereas the curve by [2] bends downwards, and the curve for $J_{(a^1)}$ is rather straight. This observation is related to how the different approaches take into account the agents' desired velocities or goals on the one hand, and their initial velocities on the other hand. In the framework of [2], trajectories' endpoints are specified a priori. Thus, when accurate goal estimates are available, predicted and actual trajectories will tend to converge again for later points in time with their method. In contrast, our models consider desired velocities and trade off following them versus keeping accelerations small. Since $J_{(a^2)}$ gives higher weight to accelerations than $J_{(a^1)}$ (cf. Table I), it leads to slower adoption of desired velocities in favor of maintaining initial velocities (similarly as CV). This difference is also visible in the reproduced trajectories, comparing e.g. the ones shown in Fig. 4(2.b), (2.c).

V. Robot Experiments

To investigate our IRL method's capability to guide a robot through human crowds, we implement an optimal control system on the smart wheelchair Qolo [17] and deploy the robot on EPFL campus, in a (usually) busy corridor (cf. Fig. 8). Our experiments were conducted under approval by the Human Research Ethical Committee of EPFL under review no. HREC-032-2019, renewed in January 2022.

1) Implementation: We use the package MinFunc[2] to optimize a cost over the actions of the robot and of tracked pedestrians on-line (cf. Fig. 7). The optimal controller's objective is defined by the learned model $J_{(a^1)}$, which allows to command the robot via its desired velocity, as the respective feature's weight $\theta^{(\mathrm{v})}_{(a^1)}$ is sufficiently high. When formulating the optimal control problem (5) to be solved on-line, we increase the time step to $h = 0.4$ s and use only $K = 12$ steps, in order to obtain the same planning time horizon of 4.8 s as in the learning phase at lower computational cost. To account for the robot's larger diameter, we set $R = 0.6$ m for the interaction energy feature. The robot's desired velocity is set equal to $\check{\mathbf{v}} = [1\text{ m/s}, 0]$, in a reference frame whose x-axis aligns with the corridor.

We implement an on-board tracking system[3] for pedestrians, using the Robot Operating System (ROS) and a LiDAR sensor mounted on the robot's front to detect legs. Firstly, planar laserscans are segmented by breakpoints [18]. Then, circles are fit to segments, and those with radius below 7 cm are retained as detections. Each detection is associated with the track with closest predicted position, if its distance is below 0.75 m, or initiates a new track otherwise. Tracks' states are estimated by α-β-filtering ($\alpha = 0.15$, $\beta = 0.02$).

To maintain the problem size tractable on-line, the state $\mathbf{x}_o$ describes the robot and, at most, 3 pedestrians, namely those maximizing a score $\zeta := -|y| - 0.25\max(x, 0) + 3\min(x, 0)$, where x and y denote a given pedestrian's position relative to the robot in its forward and lateral direction, respectively.

2) Static Obstacle: We perform a preliminary experiment, involving only the robot and a static obstacle of cylindrical shape to be detected as a leg by the robot's tracking system. In each trial, the robot is positioned around 12 m away from the obstacle and oriented such that it is facing the obstacle as precisely as possible. We perform two trials, using our method $J_{(a^1)}$ and a Velocity Obstacle approach based on [19], respectively.

3) Human Crowds: For experiments in human crowds, the robot uses our method to drive autonomously over a distance of around 15 m between uninformed pedestrians and a supervisor who is ready to remotely stop the robot at any moment. The supervisor usually walks behind the robot or by its side at a distance of around 2–5 m, except in one trial, where the supervisor deliberately interacts with the robot, shown in Fig. 10(d). In total, 5 trials are performed.

A. Results

The Fig. 9 shows the robot's motion in the preliminary experiments with the static obstacle. It successfully avoids and passes the obstacle on the left side with either method.

For the crowd experiments, Fig. 10 plots estimated trajectory segments for the five trials. In one case (cf. Fig. 10(b.3)), a distracted pedestrian stands in the robot's way, such that the robot needs to avoid a collision by itself. Here, the robot exhibits the correct tendency but does not achieve sufficient clearance, and thus, the supervisor needs to stop it for safety reasons. In the remaining 4 out of 5 trials, the robot travels the full distance without interruption. The robot's speed's average and standard deviation over all trials are 0.82 ± 0.16 m/s. For the distance traveled along the x-axis and the duration of a trial, the mean and standard deviation are 15.48 ± 4.83 m and 18.80 ± 5.17 s, respectively.

We generally observe that, in order to prevent imminent collisions, pedestrians adapt more to the robot than vice-versa. Nonetheless, the robot's contribution to collision avoidance is mostly constructive, i.e. it changes direction in such a way that the minimum predicted distance increases. This is exemplified by the situations depicted in Fig. 10(a.1), (b.3), (c.2), (d.2), (e). We also observe one case where the robot's response to an imminent collision does not match that of the involved pedestrian, as the robot plans to avoid the pedestrian on the right, whereas the pedestrian plans to avoid the robot on the left, which is depicted in Fig. 10(b.1), (b.2). There, at a time when the pedestrian has already changed direction such that their paths would not cross, the robot rotates, attempting to cross the pedestrian's future path. In two cases (cf. Fig. 10(a.2),–(e), a pedestrian actively increases clearance to the robot, which is driving behind the pedestrian.

B. Discussion

The robot's contribution to collision avoidance is often constructive, as shown by the examples in Fig. 10, but relatively small, in comparison to pedestrians' contributions. On the one hand, the robot's ability to execute planned trajectories is limited by the robot's acceleration bounds and the absence of position-feedback for tracking the trajectory (since our system's position estimates were found to be too unreliable for feedback control). On the other hand, in the example depicted in Fig. 10(b.1), it is visible that planned trajectories may lack consistency over

[2]https://www.cs.ubc.ca/schmidtm/Software/minFunc.html
[3]https://go.epfl.ch/leg-tracker

time, i.e. they may fluctuate between alternative plans, such as avoiding someone on the left or on the right. It is clear that the robot will not execute either of the alternative plans properly as long as it keeps switching between them. A remedy could be found in restricting solutions to some admissible set, or in performing some global analysis to guarantee that a local minimum is chosen in a unique fashion.

In the situation depicted in Fig. 10(b.3), the robot plans its path between two pedestrians, but they do not stand far enough apart and one of them does not see the robot. The robot's failure to avoid a collision can be explained by (i.) a non-zero velocity estimate for the actually standing pedestrian due to the robot's inaccurate estimation of its own angular velocity, (ii.) poor approximation of E by $\tilde{E}$ at low distances due to ε_1, (iii.) the aforementioned limited trajectory tracking performance, and (iv.) the wrong assumption that the pedestrian will cooperate. Our approach generates motion plans in which all agents contribute to minimizing a collective cost by adapting to some extent to each other. Thus, such plans cannot account properly for non-interacting agents. Still, by updating its plan at a high frequency, the robot may successfully avoid completely passive agents or static objects (cf. Fig. 9).

Another limitation stems from assuming that the desired velocities of agents equal their current velocities, which could be overcome by a more elaborate probabilistic estimation technique. Arguably, this would help to predict the crossing order in situations as in Fig. 10(b.1), (b.2).

VI. Conclusion

We have applied IRL in a continuous framework [6] to multi-agent navigation. We employ a novel cost feature which smoothly approximates an interaction energy proposed by an empirical study of crowd motion [4]. Two alternative feature vectors are constructed, quantifying acceleration either by its squared or smoothed magnitude, respectively, and their weights are trained on examples from the public dataset DIAMOR [15]. While both cost functions avoid collisions, the one based on squared accelerations favors smooth motions, whereas the alternative one lets agents adopt their target velocities quickly. We evaluate the models quantitatively on DIAMOR and the public dataset ETH [16]. In comparison with a prior approach to linear IRL of multi-agent navigation [2], our models yield better predictions for ETH on average, particularly in the short term, because they penalize for sharp motions.

Experiments with the robot Qolo have demonstrated that our approach can plan reasonable trajectories on-line in real-world interactions with pedestrians. However, to avoid collisions reliably, our approach requires further developments in order to 1. stick to a decision on which side to avoid a given pedestrian, 2. penalize more strongly for contact at low relative velocities, and 3. take into account the robot's non-holonomic kinematics and dynamic constraints.

VI. Acknowledgment

This letter was recommended for publication by Editor Gentiane Venture upon evaluation of the Associate Editor and Reviewers' comments.

References

[1] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-based prediction of trajectories for socially compliant navigation," in *Proc. Robot. Sci. Syst.*, Sydney, Australia, 2012, pp. 193–200.

[2] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *Int. J. Robot. Res.*, vol. 35, no. 11, pp. 1289–1307, 2016. [Online]. Available: https://doi.org/10.1177/0278364915619772

[3] Y. Che, A. M. Okamura, and D. Sadigh, "Efficient and trustworthy social navigation via explicit and implicit robot–human communication," *IEEE Trans. Robot.*, vol. 36, no. 3, pp. 692–707, Jun. 2020.

[4] I. Karamouzas, B. Skinner, and S. J. Guy, "Universal power law governing pedestrian interactions," *Phys. Rev. Lett.*, vol. 113, Dec. 2014, Art. no. 238701. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.113.238701

[5] I. Karamouzas, N. Sohre, R. Narain, and S. J. Guy, "Implicit crowds: Optimization integrator for robust crowd simulation," *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017, Art. no. 136. [Online]. Available: https://doi.org/10.1145/3072959.3073705

[6] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 475–482.

[7] A.-H. Olivier, A. Marin, A. Crétual, and J. Pettré, "Minimal predicted distance: A common metric for collision avoidance during pairwise interactions between walkers," *Gait Posture*, vol. 36, no. 3, pp. 399–404, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0966636212001117

[8] B. Kim and J. Pineau, "Socially adaptive path planning in human environments using inverse reinforcement learning," *Int. J. Social Robot.*, vol. 8, no. 1, pp. 51–66, 2016.

[9] B. D. Ziebart et al., "Planning-based prediction for pedestrians," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 3931–3936.

[10] R. Alsaleh and T. Sayed, "Modeling pedestrian-cyclist interactions in shared space using inverse reinforcement learning," *Transp. Res. Part F: Traffic Psychol. Behav.*, vol. 70, pp. 37–57, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1369847819306552

[11] M. Fahad, Z. Chen, and Y. Guo, "Learning how pedestrians navigate: A deep inverse reinforcement learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 819–826.

[12] D. Vasquez, B. Okal, and K. O. Arras, "Inverse reinforcement learning algorithms and features for robot navigation in crowds: An experimental comparison," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, USA, 2014, pp. 1341–1346.

[13] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 2096–2101.

[14] B. D. Ziebart et al., "Maximum entropy inverse reinforcement learning," in *Proc. 23rd Nat. Conf. Artif. Intell.*, Chicago, IL, USA, 2008, pp. 1433–1438.

[15] F. Zanlungo, T. Ikeda, and T. Kanda, "Potential for the dynamics of pedestrians in a socially interacting group," *Phys. Rev. E*, vol. 89, Jan. 2014, Art. no. 012811. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.89.012811

[16] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 261–268.

[17] D. Paez Granados, H. Kadone, and K. Suzuki, "Unpowered lower-body exoskeleton with torso lifting mechanism for supporting sit-to-stand transitions," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Madrid, Spain, 2018, pp. 2755–2761.

[18] G. A. Borges and M.-J. Aldon, "Line extraction in 2D range images for mobile robotics," *J. Intell. Robotic Syst.*, vol. 40, no. 3, pp. 267–297, 2004.

[19] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. Robot. Res.*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds. Berlin, Heidelberg: Springer, 2011, pp. 3–19.