Learning to Play Trajectory Games Against Opponents with Unknown Objectives

Liu, Xinjie; Peters, Lasse; Alonso-Mora, Javier

DOI
10.1109/LRA.2023.3280809

Publication date
2023

Document Version
Final published version

Published in
IEEE Robotics and Automation Letters

Learning to Play Trajectory Games Against Opponents With Unknown Objectives

Xinjie Liu, *Graduate Student Member, IEEE*, Lasse Peters, *Graduate Student Member, IEEE*, and Javier Alonso-Mora, *Senior Member, IEEE*

***Abstract*—Many autonomous agents, such as intelligent vehicles, are inherently required to interact with one another. Game theory provides a natural mathematical tool for robot motion planning in such interactive settings. However, tractable algorithms for such problems usually rely on a strong assumption, namely that the objectives of all players in the scene are known. To make such tools applicable for ego-centric planning with only local information, we propose an adaptive model-predictive game solver, which jointly infers other players' objectives online and computes a corresponding generalized Nash equilibrium (GNE) strategy. The adaptivity of our approach is enabled by a differentiable trajectory game solver whose gradient signal is used for maximum likelihood estimation (MLE) of opponents' objectives. This differentiability of our pipeline facilitates direct integration with other differentiable elements, such as neural networks (NNs). Furthermore, in contrast to existing solvers for cost inference in games, our method handles not only partial state observations but also general inequality constraints. In two simulated traffic scenarios, we find superior performance of our approach over both existing game-theoretic methods and non-game-theoretic model-predictive control (MPC) approaches. We also demonstrate our approach's real-time planning capabilities and robustness in two-player hardware experiments.**

***Index Terms*—Trajectory games, multi-robot systems, integrated planning and learning, human-aware motion planning.**

I. Introduction

MANY robot planning problems, such as robot navigation in a crowded environment, involve rich interactions with other agents. Classic "predict-then-plan" frameworks neglect the fact that other agents in the scene are responsive to the ego-agent's actions. This simplification can result in inefficient or even unsafe behavior [1]. Dynamic game theory explicitly models the interactions as coupled trajectory optimization problems from a multi-agent perspective. A noncooperative equilibrium solution of this game-theoretic model then provides strategies for all players that account for the strategic coupling of plans. Beyond that, general constraints between players, such as collision avoidance, can also be handled explicitly. All of these features render game-theoretic reasoning an attractive approach to interactive motion planning.

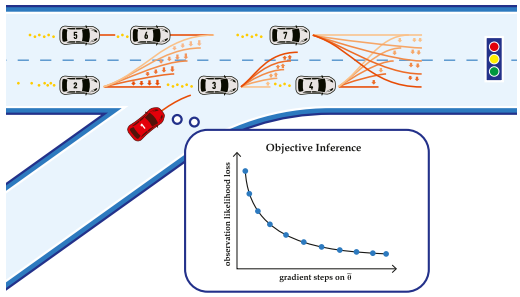


Fig. 1. An ego-agent (red) merging onto a busy road populated by six surrounding vehicles whose preferences for travel velocity and lane are initially unknown. Our approach adapts the ego agent's strategy by inferring opponents' intention parameters $\tilde{\theta}$ from partial state observations.

In order to apply game-theoretic methods for interactive motion planning from an *ego-centric* rather than *omniscient* perspective, such methods must be capable of operating only based on local information. For instance, in driving scenarios as shown in Fig. 1, the red ego-vehicle may only have partial-state observations of the surrounding vehicles and incomplete knowledge of their objectives due to unknown preferences for travel velocity, target lane, or driving style. Since vanilla game-theoretic methods require an objective model of *all* players [2], [3], this requirement constitutes a key obstacle in applying such techniques for autonomous strategic decision-making.

To address this challenge, we introduce our main contribution: a model-predictive game solver, which adapts to unknown opponents' objectives and solves for generalized Nash equilibrium (GNE) strategies. The adaptivity of our approach is enabled by a differentiable trajectory game solver whose gradient signal is used for MLE of opponents' objectives.

We perform thorough experiments in simulation and on hardware to support the following three key claims: our solver (i) outperforms both game-theoretic and non-game-theoretic baselines in highly interactive scenarios, (ii) can be combined with other differentiable components such as NNs, and (iii) is fast and robust enough for real-time planning on a hardware platform.

II. Related Work

To put our contribution into context, this section discusses four main bodies of related work. First, we discuss works on trajectory games which assume access to the objectives of all players in the scene. Then, we introduce works on inverse dynamic games that infer unknown objectives from data. Thereafter, we

Manuscript received 16 December 2022; accepted 16 April 2023. Date of publication 29 May 2023; date of current version 6 June 2023. This letter was recommended for publication by Associate Editor Yue Hu and Editor G. Venture upon evaluation of the reviewers' comments. This work was supported by European Union through ERC, INTERACT, under Grant 101041863. *(Xinjie Liu and Lasse Peters contributed equally to this work.) (Corresponding author: Xinjie Liu.)*

The authors are with the Department of Cognitive Robotics (CoR), Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: x.liu-47@student.tudelft.nl; l.peters@tudelft.nl; j.alonsomora@tudelft.nl).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3280809, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3280809

also relate our work to non-game-theoretic interaction-aware planning-techniques. Finally, we survey recent advances in differentiable optimization, which provide the underpinning for our proposed differentiable game solver.

A. N-Player General-Sum Dynamic Games

Dynamic games are well-studied in the literature [4]. In robotics, a particular focus is on multi-player general-sum games in which players may have differing yet non-adversarial objectives, and states and inputs are continuous.

Various equilibrium concepts exist in dynamic games. The Stackelberg equilibrium concept [5] assumes a "leader-follower" hierarchy, while the Nash equilibrium problem (NEP) [2], [5] does not presume such a hierarchy. Within the scope of NEP, there exist open-loop NEPs [3] and feedback NEPs [2], [6]. We refer the readers to [4] for more details about the difference between the concepts. When shared constraints exist between players, such as collision avoidance constraints, one player's feasible set may depend on other players' decisions. In that case, the problem becomes a generalized Nash equilibrium problem (GNEP) [7]. In this work, we focus on GNEPs under an open-loop information pattern which we solve by converting to an equivalent Mixed Complementarity Problem (MCP) [8].

B. Inverse Games

There are three main paradigms for solving inverse games: (i) Bayesian inference, (ii) minimization of Karush–Kuhn–Tucker (KKT) residuals, and (iii) equilibrium-constrained maximum-likelihood estimation. In type (i) methods, Le Cleac'h et al. [9] employ an Unscented Kalman Filter (UKF). This sigma-point sampling scheme drastically reduces the sampling complexity compared to vanilla particle filtering. However, a UKF is only applicable for uni-modal distributions, and extra care needs to be taken when uncertainty is multi-modal, e.g., due to multiple Nash equilibria. Type (ii) methods require *full* demonstration trajectories, i.e., including noise-free states and inputs, to cast the N-player inverse game as N independent unconstrained optimization problems [10], [11]. However, they assume full constraint satisfaction at the demonstration and have limited scalability with noisy data [12]. The type (iii) methods use KKT conditions of an open-loop Nash equilibrium (OLNE) as constraints to formulate a constrained optimization problem [12]. This type of method finds the same solution as type (ii) methods in the noise-free cases but can additionally handle partial and noisy state observations. However, encoding the equilibrium constraints is challenging, as it typically yields a non-convex problem, even in relatively simple linear-quadratic game settings. This challenge is even more pronounced when considering inequality constraints of the observed game, as this results in complementarity constraints in the inverse problem.

Our solution approach also matches the observed trajectory data in an MLE framework. In contrast to all methods above, we do so by making a GNE solver differentiable. This approach yields two important benefits over existing methods: (i) general (coupled) inequality constraints can be handled explicitly, and (ii) the entire pipeline supports direct integration with other differentiable elements, such as NNs. This latter benefit is a key motivation for our approach that is not enabled by the formulations in [9] and [12].

Note that Geiger et al. [13] explore a similar differentiable pipeline for inference of game parameters. In contrast to their work, however, our method is not limited to the special class of potential games and applies to general GNEPs.

C. Non-Game-Theoretic Interaction Models

Besides game-theoretic methods, two categories of interaction-aware decision-making techniques have been studied extensively in the context of collision avoidance and autonomous driving: (i) approaches that learn a navigation policy for the ego-agent directly without explicitly modeling the responses of others [14], [15], [16], and (ii) techniques that explicitly predict the opponents' actions to inform the ego-agent's decisions [17], [18], [19], [20], [21]. This latter category may be further split by the granularity of coupling between the ego-agent's decision-making process and the predictions of others. In the simplest case, prediction depends only upon the current physical state of other agents [22]. More advanced interaction models condition the behavior prediction on additional information such as the interaction history [17], the ego-agent's goal [19], [20], or even the ego-agent's future trajectory [18], [21].

Our approach is most closely related to this latter body of work: by solving a trajectory game, our method captures the interdependence of future decisions of all agents; and by additionally inferring the objectives of others, predictions are conditioned on the interaction history. However, a key difference of our method is that it explicitly models others as rational agents unilaterally optimizing their own cost. This assumption provides additional structure and offers a level of interpretability of the inferred behavior.

D. Differentiable Optimization

Our work is enabled by differentiating through a GNE solver. Several works have explored the idea of propagating gradient information through optimization algorithms [23], [24], [25], enabling more expressive neural architectures. However, these works focus on optimization problems and thus only apply to special cases of games, such as potential games studied by Geiger et al. [13]. By contrast, differentiating through a GNEP involves N *coupled* optimization problems. We address this challenge in Section IV-B.

III. Preliminaries

This section introduces two key concepts underpinning our work: forward and inverse dynamic games. In *forward* games, the objectives of players are known, and the task is to find players' strategies. By contrast, *inverse* games take (partial) observations of strategies as inputs to recover initially *unknown objectives*. In Section IV, we combine these two approaches into an adaptive solver that computes forward game solutions while estimating player objectives.

A. General-Sum Trajectory Games

Consider an N-player discrete-time general-sum trajectory game with horizon of T. In this setting, each player i has a control input $u_t^i \in \mathbb{R}^{m^i}$ which they may use to influence the their state $x_t^i \in \mathbb{R}^{n^i}$ at each discrete time $t \in [T]$. In this work, we assume that the evolution of each

player's state is characterized by an individual dynamical system $x^i_{t+1} = f^i(x^i_t, u^i_t)$. For brevity throughout the remainder of the letter, we shall use boldface to indicate aggregation over players and capitalization for aggregation over time, e.g., $\mathbf{x}_t := (x^1_t, \dots, x^N_t)$, $U^i := (u^i_1, \dots, u^i_T)$, $\mathbf{X} := (\mathbf{x}_1, \dots, \mathbf{x}_T)$. With a joint trajectory starting at a given initial state $\hat{\mathbf{x}}_1 := (\hat{x}^1_1, \dots, \hat{x}^N_1)$, each player seeks to find a control sequence U^i to minimize their own cost function $J^i(\mathbf{X}, U^i; \theta^i)$, which depends upon the joint state trajectory $\mathbf{X}$ as well as the player's control input sequence U^i and, additionally, takes in a parameter vector θ^i.[1] Each player must additionally consider private inequality constraints ${}^p g^i(X^i, U^i) \geq 0$ as well as shared constraints ${}^s g(\mathbf{X}, \mathbf{U}) \geq 0$. This latter type of constraint is characterized by the fact that all players have a shared responsibility to satisfy it, with a common example being collision avoidance constraints between players. In summary, this noncooperative trajectory game can be cast as a tuple of N coupled trajectory optimization problems:

$$\forall i \in [N] \begin{cases} \min\limits_{X^i, U^i} & J^i(\mathbf{X}, U^i; \theta^i) \\ \text{s.t.} & x^i_{t+1} = f^i(x^i_t, u^i_t), \forall t \in [T-1] \\ & x^i_1 = \hat{x}^i_1 \\ & {}^p g^i(X^i, \ U^i) \geq 0 \\ & {}^s g(\mathbf{X}, \ \mathbf{U}) \geq 0. \end{cases} \tag{1}$$

Note that each player's feasible set in this problem may depend upon the decision variables of others, which makes it a GNEP rather than a standard NEP [7].

A solution of this problem is a tuple of GNE strategies $\mathbf{U}^* := (U^{1*}, \dots, U^{N*})$ that satisfies the inequalities $J^i(\mathbf{X}^*, U^{i*}; \theta^i) \leq J^i((X^i, \mathbf{X}^{\neg i*}), U^i; \theta^i)$ for any feasible deviation (X^i, U^i) of any player i, with $\mathbf{X}^{\neg i}$ denoting all but player i's states. Since identifying a global GNE is generally intractable, we require these conditions only to hold locally. At a local GNE, then, no player has a unilateral incentive to deviate *locally* in feasible directions to reduce their cost.

Running example: We introduce a simple running example[2] which we shall use throughout the presentation to concretize the key concepts. Consider a tracking game played between $N = 2$ players. Let each agent's dynamics be characterized by those of a planar double-integrator, where states $x^i_t = (p^i_{x,t}, p^i_{y,t}, v^i_{x,t}, v^i_{y,t})$ are position and velocity, and control inputs $u^i_t = (a^i_{x,t}, a^i_{y,t})$ are acceleration in horizontal and vertical axes in a Cartesian frame. We define the game's state as the concatenation of the two players' individual states $\mathbf{x}_t := (x^1_t, x^2_t)$. Each player's objective is characterized by an individual cost

$$\begin{aligned} J^i = & \sum_{t=1}^{T-1} \|p^i_{t+1} - p^i_{\text{goal}}\|^2_2 + 0.1\|u^i_t\|^2_2 \\ & + 50 \max(0, d_{\min} - \|p^i_{t+1} - p^{-i}_{t+1}\|_2)^3, \end{aligned} \tag{2}$$

where we set $p^1_{\text{goal}} = p^2_t$ so that player 1, the tracking robot, is tasked to track player 2, the target robot. Player 2 has a fixed goal point p^2_{goal}. Both agents wish to get to their goal position efficiently while avoiding proximity beyond a minimal distance $d_{\min}$. Players also have shared collision avoidance constraints ${}^s g_{t+1}(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}) = \|p^1_{t+1} - p^2_{t+1}\|_2 - d_{\min} \geq 0, \forall t \in [T-1]$ and private bounds on state and controls ${}^p g^i(X^i, U^i)$. Agents need to negotiate and find an underlying equilibrium strategy in this noncooperative game, as no one wants to deviate from the direct path to their goal.

[1]The role of the parameters will become clear later in the letter when we move on to *inverse* dynamic games.

[2]Our final evaluation in Section V features denser interaction such as the 7-player ramp-merging scenario shown in Fig. 1.

B. Inverse Games

We now switch context to the *inverse* dynamic game setting. Let $\theta := (\hat{\mathbf{x}}_1, \theta^2, \dots, \theta^N)$ denote the aggregated tuple of parameters initially unknown to the ego-agent with index 1. Note that we explicitly infer the initial state of a game $\hat{\mathbf{x}}_1$ to account for the potential sensing noise and partial state observations. To model the inference task over these parameters, we assume that the ego-agent observes behavior originating from an unknown Nash game $\Gamma(\theta) := (\hat{\mathbf{x}}_1, {}^s g, \{f^i, {}^p g^i, J^i(\cdot; \theta^i)\}_{i \in [N]})$, with objective functions and constraints parameterized by initially unknown values θ^i and $\hat{\mathbf{x}}_1$, respectively.

Similar to the existing method [12], we employ an MLE formulation to allow observations to be *partial* and *noise-corrupted*. In contrast to that method, however, we also allow for inequality constraints in the hidden game. That is, we propose to solve

$$\begin{aligned} \max_{\theta, \mathbf{X}, \mathbf{U}} \quad & p(\mathbf{Y} \mid \mathbf{X}, \mathbf{U}) \\ \text{s.t.} \quad & (\mathbf{X}, \mathbf{U}) \text{ is a GNE of } \Gamma(\theta) \end{aligned} \tag{3}$$

where $p(\mathbf{Y} \mid \mathbf{X}, \mathbf{U})$ denotes the likelihood of observations $\mathbf{Y} := (\mathbf{y}_1, \dots, \mathbf{y}_T)$ given the estimated game trajectory $(\mathbf{X}, \mathbf{U})$ induced by parameters θ. This formulation yields an *mathematical program with equilibrium constraints (MPEC)* [26], where the outer problem is an estimation problem while the inner problem involves solving a dynamic game. When the observed game includes inequality constraints, the resulting inverse problem necessarily contains complementarity constraints and only few tools are available to solve the resulting problem. In the next section, we show how to transform (3) into an unconstrained problem by making the inner game differentiable, which also enables combination with other differentiable components.

Running example: We assign the tracker (player 1) to be the ego-agent and parameterize the game with the goal position of the target robot $\theta^2 = p^2_{\text{goal}}$. That is, the tracker does not know the target agent's goal and tries to infer this parameter from position observations. To ensure that (3) remains tractable, the ego-agent maintains only a fixed-length buffer of observed opponent's positions. Note that solving the inverse game requires solving games rather than optimal control problems at the inner level to account for the noncooperative nature of observed interactions, which is different from inverse optimal control (IOC) even in the 2-player case. We employ a Gaussian observation model, which we represent with an equivalent negative log-likelihood objective $\|\mathbf{Y} - r(\mathbf{X}, \mathbf{U})\|^2_2$ in (3), where $r(\mathbf{X}, \mathbf{U})$ maps $(\mathbf{X}, \mathbf{U})$ to the corresponding sequence of expected positions.

IV. Adaptive Model-Predictive Game Play

We wish to solve the problem of model-predictive game play (MPGP) from an ego-centric perspective, i.e., without prior knowledge of other players' objectives. To this end, we present an adaptive model-predictive game solver that combines the tools of Section III: first, we perform MLE of unknown objectives by solving an *inverse game* (Section III-B); then, we

solve a *forward game* using this estimate to recover a strategic motion plan (Section III-A).

A. Forward Games as MCPs

We first discuss the conversion of the GNEP in (1) to an equivalent MCP. There are three main advantages of taking this view. First, there exists a wide range of off-the-shelf solvers for this problem class [27]. Furthermore, MCP solvers directly recover strategies for all players *simultaneously*. Finally, this formulation makes it easier to reason about derivatives of the solution w.r.t. to problem data. As we shall discuss in Section IV-C, this derivative information can be leveraged to solve the inverse game problem of (3).

In order to solve the GNEP presented in (1) we derive its first-order necessary conditions. We collect all equality constraints for player i in (1) into a vector-valued function $h^i(X^i, U^i; \hat{x}_1^i)$, introduce Lagrange multipliers μ^i, ${}^p\lambda^i$ and ${}^s\lambda$ for constraints $h^i(X^i, U^i; \hat{x}_1^i)$, ${}^pg^i(X^i, U^i)$, and ${}^sg(\mathbf{X}, \mathbf{U})$ and write the Lagrangian for player i as

$$\begin{aligned}\mathcal{L}^i(\mathbf{X}, \mathbf{U}, \mu^i, {}^p\lambda^i, {}^s\lambda; \theta) &= J^i(\mathbf{X}, \mathbf{U}; \theta^i) \\ &+ \mu^{i\top} h^i(X^i, U^i; \hat{x}_1^i) - {}^s\lambda^\top {}^sg(\mathbf{X}, \mathbf{U}) - {}^p\lambda^{i\top} {}^pg^i(X^i, U^i).\end{aligned} \tag{4}$$

Note that we share the multipliers associated with shared constraints between the players to encode equal constraint satisfaction responsibility [28]. Under mild regularity conditions, e.g., linear independence constraint qualification (LICQ), a solution of (1) must satisfy the following joint KKT conditions:

$$\begin{aligned}&\forall i \in [N] \begin{cases} \nabla_{(X^i, U^i)} \mathcal{L}^i(\mathbf{X}, \mathbf{U}, \mu^i, {}^p\lambda^i, {}^s\lambda; \theta) = 0 \\ 0 \leq {}^pg^i(X^i, U^i) \perp {}^p\lambda^i \geq 0 \end{cases} \\ &h(\mathbf{X}, \mathbf{U}; \hat{\mathbf{x}}_1) = 0 \\ &0 \leq {}^sg(\mathbf{X}, \mathbf{U}) \perp {}^s\lambda \geq 0,\end{aligned} \tag{5}$$

where, for brevity, we denote by $h(\mathbf{X}, \mathbf{U}; \hat{\mathbf{x}}_1)$ the aggregation of all equality constraints. If the second directional derivative of the Lagrangian is positive along all feasible directions at a solution of (5)—a condition that can be checked a posteriori—this point is also a solution of the original game. In this work, we solve trajectory games by viewing their KKT conditions through the lens of MCPs [8, Section 1.4.2].

Definition 1: A Mixed Complementarity Problem (MCP) is defined by the following problem data: a function $F(z) : \mathbb{R}^d \mapsto \mathbb{R}^d$, lower bounds $\ell_j \in \mathbb{R} \cup \{-\infty\}$ and upper bounds $u_j \in \mathbb{R} \cup \{\infty\}$, each for $j \in [d]$. The solution of an MCP is a vector $z^* \in \mathbb{R}^n$, such that for each element with index $j \in [d]$ one of the following equations holds:

$$z_j^* = \ell_j, F_j(z^*) \geq 0 \tag{6a}$$

$$\ell_j < z_j^* < u_j, F_j(z^*) = 0 \tag{6b}$$

$$z_j^* = u_j, F_j(z^*) \leq 0. \tag{6c}$$

The parameterized KKT system of (5) can be expressed as a *parameterized family* of MCPs with decision variables corresponding to the primal and dual variables of (5),

$$z = \left[\mathbf{X}^\top, \mathbf{U}^\top, \boldsymbol{\mu}^\top, {}^p\lambda^{1\top}, \ldots, {}^p\lambda^{N\top}, {}^s\lambda^\top\right]^\top,$$

and problem data

$$F(z; \theta) = \begin{bmatrix} \nabla_{(X^1, U^1)} \mathcal{L}^i \\ \vdots \\ \nabla_{(X^N, U^N)} \mathcal{L}^N \\ h \\ {}^pg^1 \\ \vdots \\ {}^pg^N \\ {}^sg \end{bmatrix}, \quad \ell = \begin{bmatrix} -\infty \\ \vdots \\ -\infty \\ -\infty \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}, \quad u = \begin{bmatrix} \infty \\ \vdots \\ \infty \\ \infty \\ \infty \\ \vdots \\ \infty \\ \infty \end{bmatrix}, \tag{7}$$

where, by slight abuse of notation, we overload F to be parametrized by θ via $\mathcal{L}^i$ and use ∞ to denote elements for which upper or lower bounds are dropped.

B. Differentiation of an MCP Solver

An MCP solver may be viewed as a function, mapping problem data to a solution vector. Taking this perspective, for a parameterized family of MCPs as in (7), we wish to compute the function's derivatives to answer the following question: How does the solution z^* respond to local changes of the problem parameters θ?

1) The Nominal Case: Let $\Psi(\theta) := (F(\cdot; \theta), \ell, u)$ denote an MCP parameterized by $\theta \in \mathbb{R}^p$ and let $z^* \in \mathbb{R}^n$ denote a solution of that MCP, which is implicitly a function of θ. For this nominal case, we consider only solutions at which *strict complementarity* holds. We shall relax this assumption later. If F is smooth, i.e., $F(\cdot; \theta), F(z^*; \cdot) \in C^1$, we can recover the Jacobian matrix $\nabla_\theta z^* = (\frac{\partial z_j^*}{\partial \theta_k}) \in \mathbb{R}^{n \times p}$ by distinguishing two possible cases. For brevity, below, gradients are understood to be evaluated at z^* and θ.

a) Active bounds: Consider first the elements z_j^* that are either at their lower or upper bound, i.e., z_j^* satisfies (6a) or (6c). Since strict complementarity holds at the solution, $F_j(z^*; \theta)$ must be bounded away from zero with a finite margin. Hence, the smoothness of F guarantees that a local perturbation of θ will retain the sign of $F_j(z^*; \theta)$. As a result, z_j^* remains at its bound and, locally, is identically zero. Let $\tilde{\mathcal{I}} := \{k \in [n] \mid z_k^* = \ell_k \vee z_k^* = u_k\}$ denote the index set of all elements matching this condition and $\tilde{z}^* := [z^*]_{\tilde{\mathcal{I}}}$ denote the solution vector reduced to that set. Trivially, then, the Jacobian of this vector vanishes, i.e., $\nabla_\theta \tilde{z}^* = 0$.

b) Inactive bounds: The second case comprises elements that are strictly between the bounds, i.e., z_j^* satisfying (6b). In this case, under mild assumptions on F, for any local perturbation of θ there exists a perturbed solution such that F remains at its root. Therefore, the gradient $\nabla_\theta z_j^*$ for these elements is generally non-zero, and we can compute it via the implicit function theorem (IFT). Let $\bar{\mathcal{I}} := \{k \in [n] \mid F_k(z^*; \theta) = 0, \ell_k < z_k^* < u_k\}$ be the index set of all elements satisfying case (b) and let

$$\bar{z}^* := [z^*]_{\bar{\mathcal{I}}}, \quad \bar{F}(z^*, \theta) := [F(z^*; \theta)]_{\bar{\mathcal{I}}} \tag{8}$$

denote the solution vector and its complement reduced to said index set. By the IFT, the relationship between parameters θ and

solution $z^*(\theta)$ is characterized by the stationarity of $\bar{F}$:

$$\begin{aligned} 0 &= \nabla_\theta \left[\bar{F}(z^*(\theta), \theta)\right] \\ &= \nabla_\theta \bar{F} + (\nabla_{\bar{z}^*} \bar{F})(\nabla_\theta \bar{z}^*) + (\nabla_{\bar{z}^*} \bar{F}) \underbrace{(\nabla_\theta \tilde{z}^*)}_{\equiv 0} \end{aligned} \tag{9}$$

Note that, as per the discussion in case (a), the last term in this equation is identically zero. Hence, if the Jacobian $\nabla_{\bar{z}^*} \bar{F}$ is invertible, we recover the derivatives as the unique solution of the above system of equations,

$$\nabla_\theta \bar{z}^* = -\left(\nabla_{\bar{z}^*} \bar{F}\right)^{-1} (\nabla_\theta \bar{F}). \tag{10}$$

Note that (9) may not always have a unique solution, in which case (10) cannot be evaluated. We discuss practical considerations for this special case below.

2) Remarks on Special Cases and Practical Realization: The above derivation of gradients for the nominal case involves several assumptions on the structure of the problem. We discuss considerations to improve numerical robustness for practical realization of this approach below. We note that both special cases discussed hereafter are rare in practice. In fact, across 100 simulations of the running example with varying initial states and objectives, neither of them occurred.

a) Weak complementarity: The nominal case discussed above assumes strict complementarity at the solution. If this assumption does not hold, the derivative of the MCP is not defined. Nevertheless, we can still compute subderivatives at θ. Let the set of all indices for which this condition holds be denoted by $\hat{\mathcal{I}} := \{k \in [n] \mid F_k(z^*; \theta) = 0 \wedge z_k^* \in \{\ell_k, u_k\}\}$. Then by selecting a subset of $\hat{\mathcal{I}}$ and including it in $\bar{\mathcal{I}}$ for evaluation of (10), we recover a subderivative.

b) Invertibility: The evaluation (10) requires invertibility of $\nabla_{\bar{z}^*} \bar{F}$. To this end, we compute the least-squares solution of (9) rather than explicitly inverting $\nabla_{\bar{z}} \bar{F}$.

C. Model-Predictive Game Play With Gradient Descent

Finally, we present our pipeline for adaptive game-play against opponents with unknown objectives. Our adaptive MPGP scheme is summarized in Algorithm 1. At each time step, we first update our estimate of the parameters by approximating the inverse game in (3) via gradient descent. To obtain an unconstrained optimization problem, we substitute the constraints in (3) with our differentiable game solver. Following the discussion of (7), we denote by $z^*(\theta)$ the solution of the MCP formulation of the game parameterized by θ. Furthermore, by slight abuse of notation, we overload $\mathbf{X}(z^*), \mathbf{U}(z^*)$ to denote functions that extract the state and input vectors from z^*. Then, the inverse game of (3) can be written as unconstrained optimization,

$$\max_\theta \quad p(\mathbf{Y} \mid \mathbf{X}(z^*(\theta)), \mathbf{U}(z^*(\theta))). \tag{11}$$

Online, we approximate solutions of this problem by taking gradient descent steps on the negative logarithm of this objective, with gradients computed by chain rule,

$$\begin{aligned} &\nabla_\theta \left[p(\mathbf{Y} \mid \mathbf{X}(z^*(\theta)), \mathbf{U}(z^*(\theta))\right] \\ &\quad = (\nabla_{\mathbf{X}} p)(\nabla_{z^*} \mathbf{X})(\nabla_\theta z^*) + (\nabla_{\mathbf{U}} p)(\nabla_{z^*} \mathbf{U})(\nabla_\theta z^*). \end{aligned} \tag{12}$$

Here, the only non-trivial term is $\nabla_\theta z^*$, whose computation we discussed in Section IV-B. To reduce the computational cost, we warm-start using the estimate of the previous time step and terminate early if a maximum number of steps is reached. Then, we solve a forward game parametrized by the estimated $\tilde{\theta}$ to compute control commands. We execute the first control input for the ego agent and repeat the procedure.

Algorithm 1: Adaptive MPGP.

Hyper-parameters: stopping tolerance: stop_tol, learning rate: lr
Input: initial $\tilde{\theta}$, current observation buffer $\mathbf{Y}$, new observation $\mathbf{y}$
$\mathbf{Y} \leftarrow$ updateBuffer$(\mathbf{Y}, \mathbf{y})$
/* inverse game approximation */
while not stop_tol and not max_steps_reached **do**
  $(z^*, \nabla_\theta z^*) \leftarrow$ solveDiffMCP$(\tilde{\theta})$ ▷ sec. IV-B
  $\nabla_\theta p \leftarrow$ composeGradient$(z^*, \nabla_\theta z^*, \mathbf{Y})$ ▷ eq. (12)
  $\tilde{\theta} \leftarrow \tilde{\theta} - \nabla_\theta p \cdot$ lr
end
$z^* \leftarrow$ solveMCP$(\tilde{\theta})$ ▷ forward game, eq. (7)
applyFirstEgoInput(z^*)
return $\tilde{\theta}, \mathbf{Y}$

V. Experiments

To evaluate our method, we compare against two baselines in Monte Carlo studies of simulated interaction. Beyond these quantitative results, we showcase our method deployed on Jackal ground robots in two hardware experiments.

The experiments below are designed to support the key claims that our method (i) outperforms both game-theoretic and non-game-theoretic baselines in highly interactive scenarios, (ii) can be combined with other differentiable components such as NNs, and (iii) is sufficiently fast and robust for real-time planning on a hardware platform. A supplementary video of qualitative results can be found at https://xinjie-liu.github.io/projects/gamehttps://xinjie-liu.github.io/projects/game. Upon publication of this manuscript, the code for our method and experiments will be available at the same link.

A. Experiment Setup

1) Scenarios: We evaluate our method in two scenarios.

a) 2-player running example: To test the inference accuracy and convergence of our method in an intuitive setting, we first consider the 2-player running example. For evaluation in simulation, we sample the opponent's intent—i.e., their unknown goal position in (2)— uniformly from the environment. Partial observations comprise the position of each agent.

b) Ramp merging: To demonstrate the scalability of our approach and support the claim that our solver outperforms the baselines in highly interactive settings, we also test our method on a ramp merging scenario with varying numbers of players. This experiment is inspired by the setup used in [3] and is schematically visualized in Fig. 1. We model each player's dynamics by a discrete-time kinematic bicycle with the state comprising position, velocity and orientation, i.e., $x_t^i = (p_{x,t}^i, p_{y,t}^i, v_t^i, \psi_t^i)$, and controls comprising acceleration and steering angle, i.e., $u_t^i = (a_t^i, \phi)$. We capture their individual behavior by a cost function that penalizes deviation from a reference travel velocity and target lane; i.e., $\theta^i = (v_{\text{ref}}^i, p_{\text{y,lane}}^i)$. We add constraints for lane boundaries, for limits on speed, steering,

and acceleration, for the traffic light, and for collision avoidance. To encourage rich interaction in simulation, we sample each agent's initial state by sampling their speed and longitudinal positions uniformly at random from the intervals from zero to maximum velocity $v_{\max}$ and four times the vehicle length l_{car}, respectively. The ego-agent always starts on the ramp and all agents are initially aligned with their current lane. Finally, we sample each opponent's intent from the uniform distribution over the two lane centers and the target speed interval $[0.4v_{\max}, v_{\max}]$. Partial observations comprise the position and orientation of each agent.

2) Baselines: We consider the following three baselines.

a) KKT-constrained solver: In contrast to our method, the solver by Peters et al. [12] has no support for either private or shared inequality constraints. Consequently, this baseline can be viewed as solving a simplified version of the problem in (3) where the inequality constraints associated with the inner-level GNEP are dropped. Nonetheless, we still use a cubic penalty term as in (2) to encode soft collision avoidance. Furthermore, for fair comparison, we only use the baseline to *estimate* the objectives but compute control commands from a GNEP considering all constraints.

b) MPC with constant-velocity predictions: This baseline assumes that opponents move with constant velocity as observed at the latest time step. We use this baseline as a representative method for predictive planning approaches that do not explicitly model interaction.

c) Heuristic estimation MPGP: To highlight the importance of online intent inference, for the ramp merging evaluation, we also compare against a game-theoretic baseline that assumes a fixed intent for all opponents. This fixed intent is recovered by taking each agent's initial lane and velocity as a heuristic preference estimate.

To ensure a fair comparison, we use the same MCP backend [29] to solve all GNEPs and optimization problems with a default convergence tolerance of $1e^{-6}$. Furthermore, all planners utilize the same planning horizon and history buffer size of 10 time steps with a time-discretization of 0.1s. For the iterative MLE solve procedure in the 2-player running example and the ramp merging scenario, we employ a learning rate of $2e^{-2}$ for objective parameters and $1e^{-3}$ for initial states. We terminate maximum likelihood estimation iteration when the norm of the parameter update step is smaller than $1e^{-4}$, or after a maximum of 30 steps. Finally, opponent behavior is generated by solving a separate ground-truth game whose parameters are hidden from the ego-agent.

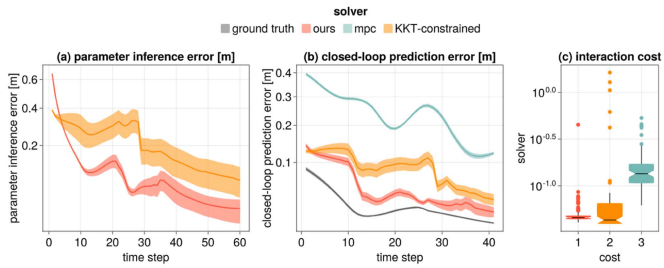


Fig. 2. Monte Carlo study for the 2-player tracking game for 100 trials. Solid lines and ribbons in (a) and (b) indicate the mean and standard error of the mean. Cost distributions in (c) are normalized by subtracting ground truth costs.

TABLE I
MONTE CARLO STUDY FOR THE RAMP MERGING SCENARIO DEPICTED IN FIG. 1 WITH 100 TRIALS FOR SETTINGS WITH 3, 5, AND 7 PLAYERS. EXCEPT FOR COLLISION AND INFEASIBLE SOLVE TIMES, ALL METRICS ARE REPORTED BY MEAN AND STANDARD ERROR OF THE MEAN

Set.	Method	Ego cost	Opp. cost	Coll.	Inf.	Traj. err. [m]	Param. err.	Time [s]
3 player	Ours	0.64 ± 0.36	0.06 ± 0.03	0	0	1.29 ± 0.05	0.41 ± 0.03	0.081 ± 0.002
	KKT-con	1.85 ± 1.21	0.05 ± 0.02	0	1	1.32 ± 0.06	2.39 ± 0.11	0.060 ± 0.002
	Heuristic	6.73 ± 2.40	0.09 ± 0.07	0	11	7.89 ± 0.26	3.96 ± 0.13	0.008 ± 0.001
	MPC	1.50 ± 0.45	0.33 ± 0.07	28	218	2.40 ± 0.11	n/a	0.009 ± 0.002
5 player	Ours	0.56 ± 0.43	0.16 ± 0.06	0	2	1.66 ± 0.07	0.47 ± 0.03	0.29 ± 0.02
	KKT-con	0.07 ± 0.32	0.06 ± 0.02	1	4	1.70 ± 0.06	2.15 ± 0.06	0.28 ± 0.02
	Heuristic	2.06 ± 0.44	0.35 ± 0.10	5	25	8.05 ± 0.19	2.91 ± 0.07	0.015 ± 0.001
	MPC	5.73 ± 2.91	0.42 ± 0.13	44	552	2.87 ± 0.13	n/a	0.014 ± 0.002
7 player	Ours	1.60 ± 1.19	0.06 ± 0.02	1	1	1.89 ± 0.05	0.46 ± 0.02	0.68 ± 0.02
	KKT-con	3.11 ± 1.72	0.09 ± 0.04	7	22	2.01 ± 0.06	1.93 ± 0.03	0.63 ± 0.06
	Heuristic	6.60 ± 1.67	0.27 ± 0.06	8	8	8.18 ± 0.15	2.44 ± 0.05	0.031 ± 0.002
	MPC	8.41 ± 1.45	0.59 ± 0.09	43	848	3.07 ± 0.08	n/a	0.0274 ± 0.004

B. Simulation Results

To compare the performance of our method to the baselines described in Section V-A2, we conduct a Monte Carlo study for the two scenarios described in Section V-A1.

1) 2-Player Running Example: Fig. 2 summarizes the results for the 2-player running example. For this evaluation, we filter out any runs for which a solver resulted in a collision. For our solver, the KKT-constrained baseline, and the MPC baseline this amounts to 2, 2 and 13 out of 100 episodes, respectively.

Figs. 2(a)–(b) show the prediction error of the goal position and opponent's trajectory, each of which is measured by ℓ^2-norm. Since the MPC baseline does not explicitly reason about costs of others, we do not report parameter inference error for it in Fig. 2(a). As evident from this visualization, both game-theoretic methods give relatively accurate parameter estimates and trajectory predictions. Among these methods, our solver converges more quickly and consistently yields a lower error. By contrast, MPC gives inferior prediction performance with reduced errors only in trivial cases, when the target robot is already at the goal. Fig. 2(c) shows the distribution of costs incurred by the ego-agent for the same set of experiments. Again, game-theoretic methods yield better performance and our method outperforms the baselines with more consistent and robust behaviors, indicated by fewer outliers and lower variance in performance.

2) Ramp Merging: Table I summarizes the results of for the simulated ramp-merging scenario for 3, 5, and 7 players.

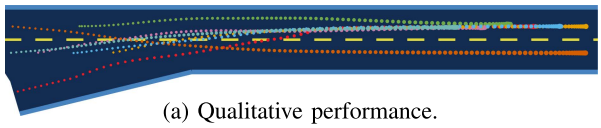
a) Task performance: To quantify the task performance, we report costs as an indicator for interaction efficiency, the number of collisions as a measure of safety, number of infeasible solves as an indicator of robustness, and trajectory and parameter error as a measure of inference accuracy. On a high level, we observe that the game-theoretic methods generally outperform the other baselines; especially for the settings with higher traffic density. While MPC achieves high efficiency (ego-cost) in the 3-player case, it collides significantly more often than the other methods across all settings. Among the game-theoretic approaches, we observe that online inference of opponent intents—as performed by our method and the KKT-constrained baseline—yields better performance than a game that uses a heuristic estimate of the intents. Within the inference-based game solvers, a Manning-Whitney U-test reveals that, across all settings, both methods achieve an ego-cost that is significantly lower than all other baselines but not significantly higher than solving the game with ground truth opponent intents. Despite this tie in terms of interaction *efficiency*, we observe a statistically significant improvement of our method over the KKT-constrained baseline in terms of *safety:* in the highly interactive 7-player case, the KKT-constrained baseline collides seven times more often than our method. This advantage is enabled by our method's ability to model inequality constraints within the inverse game.

b) Computation time: We also measure the computation time of each approach. The inference-based game solvers have generally a higher runtime than the remaining methods due to the added complexity. Within the inference methods, our method is only marginally slower than the KKT-constrained baseline, despite solving a more complex problem that includes inequality constraints. The average number of MLE updates for our method was 11.0, 19.2, and 22.7 for the 3, 5, and 7-player setting, respectively. While our current implementation achieves real-time planning rates only for up to three players, we note that additional optimizations may further reduce the runtime of our approach. Among such optimizations are low-level changes such as sharing memory between MLE updates as well as algorithmic changes to perform intent inference asynchronously at an update rate lower than the control rate. We briefly explore another algorithmic optimization in the next section.

3) Combination With an NN: To support the claim that our method can be combined with other differentiable modules, we demonstrate the integration with an NN. For this proof of concept, we use a two-layer feed-forward NN, which takes the buffer of recent partial state observations as input and predicts other players' objectives. Training of this module is enabled by propagating the gradient of the observation likelihood loss of (11) through the differentiable game solver to the parameters of the NN. Online, we use the network's prediction as an initial guess to reduce the number gradient steps. As summarized in Fig. 3 , this combination reduces the computation time by more than 60% while incurring only a marginal loss in performance.

C. Hardware Experiments

To support the claim that our method is sufficiently fast and robust for hardware deployment, we demonstrate the tracking game in the running example in Section III-A with a Jackal ground robot tracking (i) another Jackal robot (Fig. 4(a)) and (ii) a human player (Fig. 4(b)), each with initially unknown goals. Plans are computed online on a mobile i7 CPU. We generate plans using the point mass dynamics with a velocity constraint of $0.8\ \mathrm{ms}^{-1}$ and realize low-level control via the feedback controller of [30]. A video of these hardware demonstrations is included in the supplementary material. In both experiments, we observe that our adaptive MPGP planner enables the robot to infer the unknown goal position to track the target while avoiding collisions. The average computation time in both experiments was 0.035 s.

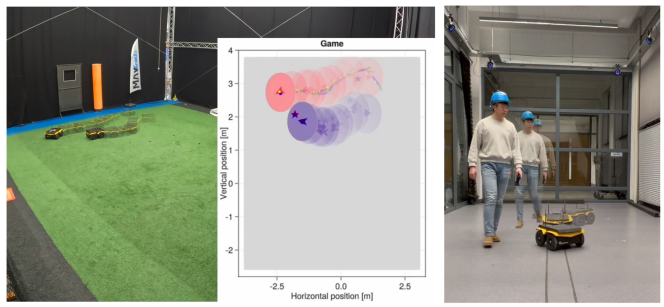


(a) Qualitative performance.

Ego cost	Opp. cost	Coll.	Inf.	Traj. err. [m]	Param. err.	Time [s]
2.19 ± 1.21	0.17 ± 0.07	3	5	2.34 ± 0.08	0.91 ± 0.08	0.274 ± 0.01

(b) Quantitative performance.

Fig. 3. Performance of our solver in combination with an NN for 100 trials of the 7-player ramp merging scenario.



(a) (b)

Fig. 4. Time lapse of the running-example in which a Jackal tracks (a) another Jackal and (b) a human. Overlaid in (a) are the position of target robot (red) its true goal (red star), the tracker (blue), and its goal estimate (blue star).

VI. Conclusion

In this letter, we presented a model-predictive game solver that adapts strategic motion plans to initially unknown opponents' objectives. The adaptivity of our approach is enabled by a differentiable trajectory game solver whose gradient signal is used for MLE of unknown game parameters. As a result, our adaptive MPGP planner allows for safe and efficient interaction with other strategic agents without assuming prior knowledge of their objectives or observations of full states. We evaluated our method in two simulated interaction scenarios and demonstrated superior performance over a state-of-the-art game-theoretic planner and a non-interactive MPC baseline. Beyond that, we demonstrated the real-time planning capability and robustness of our approach in two hardware experiments.

In this work, we have limited inference to parameters that appear in the objectives of other players. Since the derivation of the gradient in Section IV-B can also handle other parameterizations of F—so long as they are smooth—future work may extend this framework to infer additional parameters of

constraints or aspects of the observation model. Furthermore, encouraged by the improved scalability when combining our method with learning modules such as NNs, we seek to extend this learning pipeline in the future. One such extension would be to operate directly on raw sensor data, such as images, to exploit additional visual cues for intent inference. Another extension is to move beyond MLE-based point estimates to inference of potentially multi-modal distributions over opponent intents, which may be achieved by embedding our differentiable method within a variational autoencoder. Finally, our framework could be tested on large-scale datasets of real autonomous-driving behavior.

ACKNOWLEDGMENT

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

REFERENCES

[1] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 797–803.

[2] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin, "Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 1475–1481.

[3] L. Cleac'h, M. Schwager, and Z. Manchester, "ALGAMES: A fast augmented Lagrangian solver for constrained dynamic games," *Auton. Robots*, vol. 46, no. 1, pp. 201–215, 2022.

[4] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, 2nd ed. Philadelphia, PA, USA: SIAM, 1999.

[5] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 3, pp. 884–897, May 2020.

[6] F. Laine, D. Fridovich-Keil, C.-Y. Chiu, and C. Tomlin, "The computation of approximate generalized feedback Nash equilibria," *SIAM J. Optim.*, vol. 33, no. 1, pp. 294–318, 2023.

[7] F. Facchinei and C. Kanzow, "Generalized Nash equilibrium problems," *Ann. Operations Res.*, vol. 175, no. 1, pp. 177–211, 2010.

[8] F. Facchinei and J.-S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Berlin, Germany: Springer, 2003.

[9] L. Cleac'h, M. Schwager, and Z. Manchester, "LUCIDGames: Online unscented inverse dynamic games for adaptive trajectory prediction and planning," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5485–5492, Jul. 2021.

[10] C. Awasthi and A. Lamperski, "Inverse differential games with mixed inequality constraints," in *Proc. IEEE Amer. Control Conf.*, 2020, pp. 2182–2187.

[11] S. Rothfuß, J. Inga, F. Köpf, M. Flad, and S. Hohmann, "Inverse optimal control for identification in non-cooperative differential games," *IFAC-LettersOnLine*, vol. 50, no. 1, pp. 14909–14915, 2017.

[12] L. Peters, D. Fridovich-Keil, V. R. Royo, C. J. Tomlin, and C. Stachniss, "Inferring objectives in continuous dynamic games from noise-corrupted partial state observations," in *Proc. Robot.: Sci. Syst.*, 2021. [Online]. Available: https://www.roboticsproceedings.org/rss18/p051.html

[13] P. Geiger and C.-N. Straehle, "Learning game-theoretic models of multiagent trajectories using implicit layers," in *Proc. Conf. Advancements Artif. Intell.*, 2021, pp. 4950–4958.

[14] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst.*, 2018, pp. 3052–3059.

[15] B. Brito, M. Everett, J. P. How, and J. Alonso-Mora, "Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4616–4623, Jul. 2021.

[16] V. Tolani, S. Bansal, A. Faust, and C. Tomlin, "Visual navigation among humans with optimal control as a supervisor," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2288–2295, Apr. 2021.

[17] 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.

[18] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone, "Multimodal probabilistic model-based planning for human-robot interaction," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1–9.

[19] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction conditioned on goals in visual multi-agent settings," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 2821–2830.

[20] J. Roh, C. Mavrogiannis, R. Madan, D. Fox, and S. Srinivasa, "Multimodal trajectory prediction via topological invariance for navigation at uncontrolled intersections," in *Proc. Conf. Robot Learn.*, 2021, pp. 2216–2227.

[21] M. Sun, F. Baldini, P. Trautman, and T. Murphey, "Move beyond trajectories: Distribution space coupling for crowd navigation," in *Proc. Robot.: Sci. Syst.*, 2021. [Online]. Available: https://www.roboticsproceedings.org/rss17/p053.html

[22] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, "What the constant velocity model can teach us about pedestrian motion prediction," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1696–1703, Apr. 2020.

[23] D. Ralph and S. Dempe, "Directional derivatives of the solution of a parametric nonlinear program," *Math. Program.*, vol. 70, no. 1, pp. 159–172, 1995.

[24] B. Amos and J. Z. Kolter, "Optnet: Differentiable optimization as a layer in neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 136–145.

[25] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 9558–9570.

[26] Z.-Q. Luo, J.-S. Pang, and D. Ralph, *Mathematical Programs With Equilibrium Constraints*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

[27] S. C. Billups, S. P. Dirkse, and M. C. Ferris, "A comparison of large scale mixed complementarity problem solvers," *Comput. Optim. Appl.*, vol. 7, no. 1, pp. 3–25, 1997.

[28] A. A. Kulkarni and U. V. Shanbhag, "On the variational equilibrium as a refinement of the generalized Nash equilibrium," *Automatica*, vol. 48, no. 1, pp. 45–55, 2012.

[29] S. P. Dirkse and M. C. Ferris, "The PATH solver: A nommonotone stabilization scheme for mixed complementarity problems," *Optim. Methods Softw.*, vol. 5, no. 2, pp. 123–156, 1995.

[30] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1990, pp. 384–389.