Journal Title
XX(X):1–16
©The Author(s) 2018
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Learning Players' Objectives in Continuous Dynamic Games from Partial State Observations

Lasse Peters[1,3], Vicenç Rubies-Royo[2], Claire J. Tomlin[2], Laura Ferranti[1], Javier Alonso-Mora[1], Cyrill Stachniss[3], and David Fridovich-Keil[4]

[1]Delft University of Technology
[2]University of California, Berkeley
[3]University of Bonn
[4]The University of Texas at Austin

Abstract

Robots deployed to the real world must be able to interact with other agents in their environment. Dynamic game theory provides a powerful mathematical framework for modeling scenarios in which agents have individual objectives and interactions evolve over time. However, a key limitation of such techniques is that they require a-priori knowledge of all players' objectives. In this work, we address this issue by proposing a novel method for learning players' objectives in continuous dynamic games from noise-corrupted, partial state observations. Our approach learns objectives by coupling the estimation of unknown cost parameters of each player with inference of unobserved states and inputs through Nash equilibrium constraints. By coupling past state estimates with future state predictions, our approach is amenable to simultaneous online learning and prediction in receding horizon fashion. We demonstrate our method in several simulated traffic scenarios in which we recover players' preferences for, e.g., desired travel speed and collision-avoidance behavior. Results show that our method reliably estimates game-theoretic models from noise-corrupted data that closely matches ground-truth objectives, consistently outperforming state-of-the-art approaches.

Keywords

inverse dynamic games, inverse optimal control, multi-agent prediction

1 Introduction

To operate safely and efficiently in environments shared with other agents, robots must be able to predict the effects of their actions on the decisions of others. In many such settings, agents do not form a single team that shares a joint objective. Instead, each agent may have an individual objective, encoded by a cost function which they optimize unilaterally. Unless the objectives of all agents are perfectly aligned, agents must therefore compete to minimize their own cost, while accounting for the strategic behavior of others. For example, consider the highway navigation scenario in Figure 1. Here, each driver travels along the highway with an individual objective that encodes their preferences for speed, acceleration, and proximity to other cars. In heavy traffic, the objectives of drivers may conflict. For instance, if car 1 (blue) wishes to maintain its speed, it must overtake the slower vehicles in front. At the same time, however, the faster car 2 (orange) may wish maintain its speed and but would be forced to decelerate if the driver of car 1 changes lanes.

Mathematically, such interactions of multiple agents with individual, potentially conflicting objectives are characterized by a *noncooperative dynamic game*. The theory underpinning dynamic games is well established (Isaacs 1954-1955; Başar and Olsder 1999) and recent work has put forth efficient algorithms to determine equilibrium solutions to these problems, given players' objectives (Fridovich-Keil et al. 2020; Di and Lamperski 2019). The *forward* game problem is depicted in Figure 1 (left to right) for the highway driving scenario: given the cost functions of all players (left), a forward game solver computes their rational strategies and corresponding future trajectories (right).

Unfortunately, the objectives of agents in a scene are often not known a priori. Therefore, in order for game-theoretic methods to find practical application in fields such as robotics, it is imperative to recover these objectives from data. This *inverse* dynamic game problem is illustrated in Figure 1 (right to left) for the highway driving scenario: given observations of player's strategies (right), an inverse game solver recovers objectives (left) which explain the observed behavior. This inverse dynamic game problem is the focus of this work.

The challenge of recovering objectives from observed behavior has been extensively studied in the literature on inverse optimal control (IOC) (Kalman 1964; Mombaur et al. 2010; Albrecht et al. 2011) and inverse reinforcement

Corresponding author:
Lasse Peters, Delft University of Technology, Delft, The Netherlands.
Email: l.peters@tudelft.nl

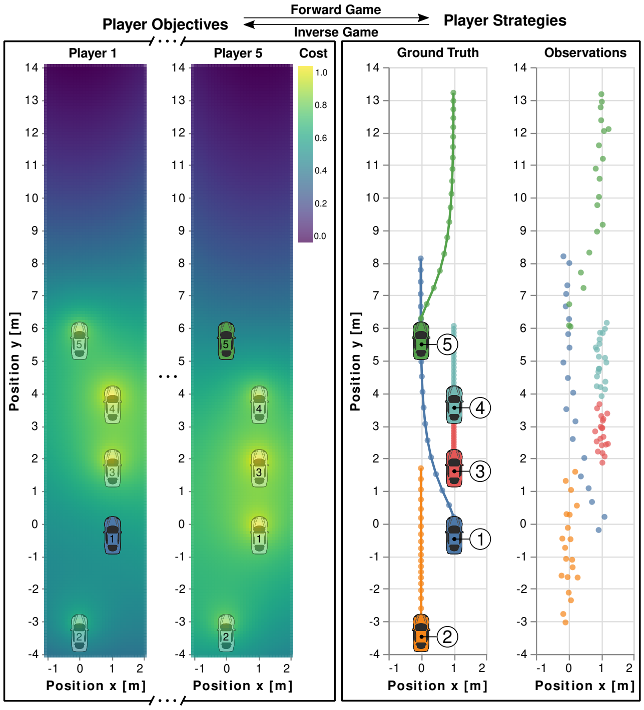


Figure 1. 5-player highway driving scenario, modeled as a dynamic game. Solving the “forward” problem amounts to finding optimal trajectories (right) for all cars, given their objectives (left). In contrast, this paper addresses the “inverse,” i.e., it estimates the objectives of each player given noise-corrupted observations of each agent’s trajectories. For example, our method can infer properties such as the degree to which each player wishes to keep a safe distance from others (heatmap, left). These learned objectives constitute an abstract model which can be used to predict players’ actions in the future.

learning (IRL) (Ng and Russell 2000; Ziebart et al. 2008). Unfortunately, however, these methods are fundamentally limited to the single-player setting. While recent efforts extend these ideas to multi-agent IRL (Šošić et al. 2016; Natarajan et al. 2010), those approaches are limited to games with *potential* cost structures (Monderer and Shapley 1996) and do not directly apply in general noncooperative settings. While initial work extends IOC methods to address this limitation (Rothfuß et al. 2017; Inga et al. 2019; Awasthi and Lamperski 2020), these inverse dynamic game solvers rely upon full observation of states and inputs of all players.

The main contribution of this work is a novel method for learning player’s objectives in noncooperative dynamic games from only noise-corrupted, partial state observations. In addition to learning a cost model for all players, our method also recovers a forward game solution consistent with the learned objectives by enforcing equilibrium constraints on latent trajectory estimates. This bilevel formulation further allows to couple observed and predicted behavior to recover player’s objectives even from temporally-incomplete interactions. As a result, our approach is amenable for online learning and prediction in receding horizon fashion.

This paper builds upon and extends our earlier work (Peters et al. 2021). In this work, we provide a more in-depth analysis of that approach. Additionally, while our original work was limited to offline operation and could therefore only recover players’ objectives for interactions which had already occurred, in this work we remove this requirement.

We evaluate our method in extensive Monte Carlo simulations in several traffic scenarios with varying numbers of players and interaction geometries. Empirical results show that our approach is more robust to partial state observations, measurement noise, and unobserved time-steps than existing methods, and consequently is more suitable for predicting agents’ actions in the future.

2 Prior Work

We begin by discussing recent advances in the well-studied area of IOC. While methods from that field address only single-player, cooperative settings, this body of work exposes many of the important mathematical and algorithmic concepts that appear in games. We discuss how some of these approaches have been applied in the noncooperative multi-player setting and emphasize the connections between existing approaches and our contributions.

2.1 Single-Player Inverse Optimal Control

The IOC problem has been extensively studied since the well-known work of Kalman (1964). In the context of IRL, early formulations such as that of Ng and Russell (2000) and maximum-entropy variants (Ziebart et al. 2008; Kretzschmar et al. 2016) have proven successful in treating problems with discrete state and control sets. In robotic applications, optimal control problems typically involve decision variables in a continuous domain. Hence, recent work in IOC differs from the IRL literature mentioned above as it is explicitly designed for smooth problems.

One common framework for addressing IOC problems with nonlinear dynamics and nonquadratic cost structures is bilevel optimization (Mombaur et al. 2010; Albrecht et al. 2011). Here, the *outer* problem is a least squares or maximum likelihood estimation (MLE) problem in which demonstrations are matched with a nominal trajectory estimate and decision variables parameterize the objective of the underlying optimal control problem. The *inner* problem determines the nominal trajectory estimate as the optimizer of the “forward” (i.e., standard) optimal control problem for the outer problem’s decision variables. A key benefit of bilevel IOC formulations is that they naturally adapt to settings with noise-corrupted partial state observations (Albrecht et al. 2011).

Early bilevel formulations for IOC utilize derivative-free optimization schemes to estimate the unknown objective parameters in order to avoid explicit differentiation of the solution to the inner optimal control problem (Mombaur et al. 2010). That is, the inner solver is treated as a black-box mapping from cost parameters to optimal trajectories which is utilized by the outer solver to identify the unknown parameters using a suitable derivative-free method. While black-box approaches can be simple to implement due to their modularity and lack of reliance on derivative information, they often suffer from a high sampling complexity (Nocedal and Wright 2006). Since each sample in the context of black-box IOC methods amounts to solving a full optimal control problem, such approaches remain intractable for scenarios with large state spaces or additional unknown parameters, such as unknown initial conditions.

Other works instead embed the Karush–Kuhn–Tucker (KKT) conditions of the inner problem as constraints on the outer problem. Since these techniques enforce only first-order necessary conditions of optimality, globally optimal observations are unnecessary and locally optimal demonstrations suffice. Yet, a key computational difficulty of KKT-constrained IOC formulations is that they yield a nonconvex optimization problem due to decision variables in the outer problem appearing nonlinearly with inner problem variables in KKT constraints. This occurs even in the relatively benign case of linear-quadratic IOC.

In contrast to bilevel optimization formulations where necessary conditions of optimality are embedded as constraints, recent methods (Levine and Koltun 2012; Englert and Toussaint 2018; Awasthi 2019; Menner and Zeilinger 2020; Jin et al. 2021) minimize the residual of these conditions directly at the demonstrations. Since the observed demonstration is assumed to satisfy any constraints of the underlying forward optimal control problem, this method can be formulated as fully unconstrained optimization. Additionally, these residual formulations yield a *convex* optimization problem if the class of objective functions is convex in the unknown parameters at the demonstration (Keshavarz et al. 2011; Englert and Toussaint 2018). This condition holds in the common setting of linearly-parameterized objective functions. Levine and Koltun (2012) propose a variant of this approach that utilizes quadratic approximations of the reward model around demonstrations to derive optimality residuals in a maximum entropy framework. Englert and Toussaint (2018) present an extensions of this method do accommodate inequality constraints on states and inputs. Much like KKT-constrained formulations, these residual methods operate on locally optimal demonstrations. However, an important limitation of residual methods is that they require observations of full state and input sequences. More recently, Menner and Zeilinger (2020) compared IOC techniques based on KKT constraints and residuals and demonstrated inferior performance of the latter even in problems with linear dynamics and quadratic target objectives.

Our work takes inspiration from the KKT-constraint formulation for single-player IOC as discussed by Albrecht et al. (2011) and Menner and Zeilinger (2020). While these works apply only to single-player settings, we utilize the necessary conditions for open-loop Nash equilibria (OLNEs) (Başar and Olsder 1999) to generalize this approach to noncooperative multi-player scenarios.

2.2 Multi-Player Inverse Dynamic Games

Many of the IOC techniques discussed above have close analogues in the context of multi-player inverse dynamic games.

As in single-player IOC, methods akin to black-box bilinear optimization have also been studied in the context of inverse games (Peters 2020; Le Cleac'h et al. 2021). Peters (2020) uses a particle-filtering technique for online estimation of human behavior parameters. This work demonstrates the importance of inferring human behavior parameters for accurate prediction in interactive scenarios. However, there, inference is limited to a single parameter and the work highlight the challenges associated with scaling this sampling based approach to high-dimensional latent parameter spaces. Le Cleac'h et al. (2021) employ a similar derivative-free filtering technique based on an unscented Kalman filter. While this approach drastically reduces the overall sample complexity, it still relies on exact observations of the state to reduce the required number of solutions to full dynamic games at the inner level.

Another line of research has put forth solution techniques for inverse games that follow from the residual methods outlined in Section 2.1 (Köpf et al. 2017; Rothfuß et al. 2017; Awasthi and Lamperski 2020; Inga et al. 2019). Köpf et al. (2017) study a special case of an inverse linear-quadratic game in which the equilibrium feedback strategies of all but one player are known. This assumption reduces the estimation problem to single-player IOC to which the residual methods discussed above can be applied directly. Rothfuß et al. (2017) present a more general approach that does not exploit such special structure but instead minimizes the residual of the first-order necessary conditions for a local OLNE. Inga et al. (2019) present a variant of this OLNE residual method in a maximum entropy framework, generalizing the single-player IOC algorithm proposed by Levine and Koltun (2012). Recently, Awasthi and Lamperski (2020) also extended the OLNE residual method of Rothfuß et al. (2017) to inverse games with state and input constraints. This approach extends that of Englert and Toussaint (2018) to noncooperative multi-player scenarios.

All of these inverse game KKT residual methods share many properties with their single-player counterparts. In particular, since they rely upon only local equilibrium criteria, they are able to recover player objectives even from local—rather than only global—equilibrium demonstrations. However, as in the single-player case, they rely upon observation of both state and input to evaluate the residuals.

In contrast to KKT residual methods (Rothfuß et al. 2017; Awasthi and Lamperski 2020; Inga et al. 2019), we enforce these conditions as constraints on a jointly estimated trajectory, rather than minimizing the residual of these conditions directly at the observation. By maintaining a trajectory estimate in this manner, our method explicitly accounts for observation noise, partial state observability, and unobserved control inputs. Furthermore, in contrast to black-box approaches to the inverse dynamic game problem (Peters 2020; Le Cleac'h et al. 2021), our method does not require repeated solutions of the underlying forward game. Moreover, our method returns a full forward game solution in addition to the estimated objective parameters for all players.

3 Background: Open-Loop Nash Games

While this work is concerned with the *inverse* game problem of learning objectives from observed behavior, we first provide a technical introduction to the theory of *forward* open-loop dynamic Nash games. These forward games correspond to the model that we seek to recover in this work. Furthermore, as we shall discuss in Section 4, they may be used at the inner level of a bilevel optimization problem to formulate the inverse game problem.

As discussed in Section 1, dynamic games provide an expressive mathematical formalism for modeling the strategic interactions of multiple agents with differing objectives. Interested readers are directed to (Başar and Olsder 1999) for a more complete discussion. We note that dynamic games afford a wide variety of equilibrium concepts; our choice of open-loop Nash Equilibria in this work captures scenarios in which players do not account for future information gains and instead commit to a sequence of control decisions *a priori*. These conditions may occur when occlusions *prevent* future information gains or when bounded rationality causes players to *ignore* them. OLNE have been demonstrated to capture dynamic interaction when embedded in receding-horizon re-planning schemes (Wang et al. 2019; Le Cleac'h et al. 2020). Beyond that, restricting our attention to OLNE engenders computational advantages which are discussed below. Other choices of solution concept are possible and should be explored in future work. Recent methods such as those of Di and Lamperski (2019) and Le Cleac'h et al. (2020) facilitate efficient solutions to the "forward" open-loop games *given players' objectives a priori*.

3.1 Preliminaries

Consider a game played between N players over discrete time-steps $t \in [T] := \{1, \dots, T\}$. The game is comprised of three key components: dynamics, objectives (which are later presumed to be unknown in this work), and information structure.

We presume that the game is Markov with respect to state $x \in \mathbb{R}^n$. That is, given each player's control input $u^i \in \mathbb{R}^{m^i}$, $i \in [N]$, the state evolves according to the difference equation

$$x_{t+1} = f_t(x_t, u_t^1, \dots, u_t^N). \tag{1}$$

For clarity, we shall introduce the following shorthand notation:

$$\begin{aligned}\mathbf{x} &= (x_1, \dots, x_T),\\ \mathbf{u}^i &= (u_1^i, \dots, u_T^i),\\ \mathbf{u}_t &= (u_t^1, \dots, u_t^N),\\ \mathbf{u} &= (\mathbf{u}^1, \dots, \mathbf{u}^N).\end{aligned}$$

Observe that the state x pertains to the entire game, not only to a single player. In the examples presented in this paper, x is simply the concatenation of individual players' states, and correspondingly the dynamics are independent for all players. However, this is not always the case and the methods developed here apply in the more general settings as well.

The objective of player i is encoded by their distinct cost function J^i, which they seek to minimize. This cost can in general depend upon the sequence of states and inputs for all players.[*] In this paper, we presume that objectives are expressed in time-additive form, as is common across the optimal control and reinforcement learning literature:

$$J^i(\mathbf{x}, \mathbf{u}) := \sum_{t=1}^{T} g_t^i(x_t, u_t^1, \dots, u_t^N). \tag{2}$$

Since the state trajectory $\mathbf{x}$ follows Equation (1), these cost functions can also be written in terms of the inital condition x_1 and the sequence of control inputs for all players $\mathbf{u}$. For this reason, we shall also use the notation $J^i(\mathbf{u}; x_1)$, and refer to the tuple of initial state, dynamics, and objectives as

$$\Gamma := \left(x_1, \{f_t\}_{t \in [T]}, \{J^i\}_{i \in [N]}\right). \tag{3}$$

Finally, the information structure of a dynamic game refers to the information available to each player when they are required to make a decision at each time. At time t, then, Player-i's input is a function $\gamma_t^i : \mathcal{I}_t^i \to \mathbb{R}^{m^i}$, where $\mathcal{I}_t^i$ is the set of information available to Player-i at time t. In this paper, we consider *open-loop* information structures, i.e., where $\mathcal{I}_t^i = \{x_1\}$.[†] In open-loop information, then, it suffices for Player-i to specify their input sequence $\mathbf{u}^i$ given a fixed initial condition x_1. For this reason, we neglect a more detailed treatment of strategy spaces and information structure, and simply refer to the finite-dimensional sequence of control inputs for each player.

This characterization of a dynamic game is intentionally general. Our solution methods will rely upon established numerical methods for smooth optimization, however, and as such we require the following assumption.

Assumption 1. *(Smoothness) Dynamics f and objectives J^i have well-defined second derivatives in all state and control variables, at all times and for all players.*

Most physical systems of interest and interactions thereof are naturally modeled in this way. However, we note that, for example, hybrid dynamics such as those induced by contact do not satisfy this assumption.

We shall illustrate key concepts using a consistent "running example" throughout the paper.

Running example: *Consider an $N = 2$-player linear-quadratic (LQ) game—i.e., one in which dynamics f_t are linear in state x_t and control inputs $\mathbf{u}_t$, and costs J^i are quadratic in states and controls. Let each player independently follow the dynamics of a double integrator in the Cartesian plane. State $x = (p_x^1, p_y^1, \dot{p}_x^1, \dot{p}_y^1, p_x^2, p_y^2, \dot{p}_x^2, \dot{p}_y^2)$ then evolves with inputs $u^i = (\ddot{p}_x^i, \ddot{p}_y^i)$ according to*

$$x_{t+1} = \overbrace{\begin{bmatrix}\tilde{A} & 0\\ 0 & \tilde{A}\end{bmatrix}}^{A} x_t + \overbrace{\begin{bmatrix}\tilde{B}\\ 0\end{bmatrix}}^{B^1} u_t^1 + \overbrace{\begin{bmatrix}0\\ \tilde{B}\end{bmatrix}}^{B^2} u_t^2, \tag{4}$$

$$\textit{where } \tilde{A} = \begin{bmatrix}1 & 0 & \Delta t & 0\\ 0 & 1 & 0 & \Delta t\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}, \tilde{B} = \begin{bmatrix}0 & 0\\ 0 & 0\\ \Delta t & 0\\ 0 & \Delta t\end{bmatrix},$$

[*] State and input constraints are also possible, although they complicate the notion of equilibrium solution. Solution methods such as those of Dirkse and Ferris (1995) and Laine et al. (2021) address constrained forward games. The present paper readily extends to the constrained case; however, we neglect them for clarity of presentation.

[†] Recent work in solving forward games also considers *feedback* information in which $\mathcal{I}_t^i = \{x_t\}$; see Fridovich-Keil et al. (2020) and Laine et al. (2021).

and Δt is a uniform time discretization, e.g., 0.1s. Each player has a quadratic objective of the form

$$J^i = \frac{1}{2}\sum_{t=1}^{T}\left(\theta_Q^i x_t Q_t^i x_t + \sum_{j=1}^{N}\theta_R^{ij} u_t^{j\top} R_t^{ij} u_t^j\right). \tag{5}$$

In this simple example, Q_t^i and R_t^{ij} are known, positive definite matrices encoding the preferences of each player. The scalars $\theta_Q^i \in \mathbb{R}$ and $\theta_R^{ij} \in \mathbb{R}$ weight these known matrices. In this paper, we develop a technique to learn a priori *unknown parameters such as the costs weights above from both offline and online data. Note that this simple LQ game shall only serve to explain the general concepts of our method. For our experiments presented in Section 7, we consider more complex problems with nonlinear dynamics and nonquadratic costs, such as the 5-player highway navigation problem shown in Figure 1.*

3.2 The Nash Solution Concept

Combining these components, each player i in an open-loop dynamic game seeks to solve the following optimization problem

$$\forall i \in [N] \begin{cases} \min\limits_{\mathbf{x},\mathbf{u}^i} \; J^i(\mathbf{u}; x_1) & \text{(6a)} \\ \text{s.t. } x_{t+1} = f_t(x_t, \mathbf{u}_t), \forall t \in [T-1]. & \text{(6b)} \end{cases}$$

There exist a variety of distinct solution concepts for such smooth open-loop dynamic games. In this paper, we consider the well-known Nash equilibrium concept, wherein no player has a unilateral incentive to change its strategy. Mathematically, the Nash concept is defined as follows.

Definition 1. *(Open-loop Nash equilibrium) The strategies $\mathbf{u}^* := (\mathbf{u}^{1*}, \dots, \mathbf{u}^{N*})$ constitute an open-loop Nash equilibrium (OLNE) in the game $\Gamma = \left(x_1, \{f_t\}_{t\in[T]}, \{J^i\}_{i\in[N]}\right)$ if the following inequalities hold:*

$$J^{i*} = J^i(\mathbf{u}^*; x_1) \le J^i\left((\mathbf{u}^i, \mathbf{u}^{-i*}); x_1\right), \forall i \in [N]. \tag{7}$$

Here, we use the shorthand $(\mathbf{u}^i, \mathbf{u}^{-i*})$ to indicate the collection of strategies in which only *Player-i deviates from the Nash profile, i.e., $\mathbf{u}^i \neq \mathbf{u}^{i*}$.*

Note that, at a Nash equilibrium, each player must *independently* have no incentive to deviate from its strategy. Since players' objectives may generally conflict, the Nash concept encodes noncooperative, rational, and potentially selfish behavior.

Unfortunately, Nash equilibria are known to be very difficult to find in general (Daskalakis et al. 2009). In this work, we seek only *local* equilibria which satisfy the Nash conditions Equation (7) to first order. That is, following similar approaches in both single-player IOC (Albrecht et al. 2011; Englert and Toussaint 2018) and forward/inverse open-loop games (Le Cleac'h et al. 2020; Awasthi 2019), we encode forward optimality via the corresponding first-order necessary conditions. These first-order necessary conditions are given by the union of the individual players' KKT conditions, i.e.,

$$\mathbf{0} = \mathbf{G}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}) := \begin{bmatrix} \left.\begin{matrix} \nabla_{\mathbf{x}} J^i + \nabla_{\mathbf{x}} \mathbf{F}(\mathbf{x}, \mathbf{u})^\top \boldsymbol{\lambda}^i \\ \nabla_{\mathbf{u}^i} J^i + \nabla_{\mathbf{u}^i} \mathbf{F}(\mathbf{x}, \mathbf{u})^\top \boldsymbol{\lambda}^i \end{matrix}\right\} \forall i \in [N] \\ \mathbf{F}(\mathbf{x}, \mathbf{u}) \end{bmatrix}. \tag{8}$$

Here, the first two block-rows are repeated for all players, and the function $\mathbf{F}(\mathbf{x}, \mathbf{u})$ accumulates the dynamic constraints of Equation (6a) at all time steps, with the t^{th} row given by $x_{t+1} - f_t(x_t, u_t^1, \dots, u_t^N)$. Note that we have also introduced costate variables $\boldsymbol{\lambda}^i := (\lambda_1^i, \dots, \lambda_{T-1}^i)$ for each player, with $\lambda_t^i \in \mathbb{R}^n$ the Lagrange multiplier corresponding to Player-i's dynamics constraint in Equation (6a) at time step t. Note that, as with control inputs, we use the notation $\boldsymbol{\lambda} := (\boldsymbol{\lambda}^1, \dots, \boldsymbol{\lambda}^N)$.

Running example: *Consider the two-player LQ example above with double integrator dynamics given by Equation (4) and quadratic objectives given by Equation (5). The t^{th} block of the first row of Equation (8) is given by*

$$\mathbf{0} = \theta_Q^i Q_t^i x_t + \lambda_{t-1}^i - A^\top \lambda_t^i \tag{9}$$

for Player-i. Likewise, the t^{th} block of the second row of Equation (8) for Player-i is given by

$$\mathbf{0} = \theta_R^{ii} R_t^{ii} u_t^i - B^{i\top} \lambda_t^i. \tag{10}$$

Finally, the t^{th} block of the final row of Equation (8) is given by

$$\mathbf{0} = x_{t+1} - A x_t - B^1 u_t^1 - B^2 u_t^2. \tag{11}$$

Computationally, the KKT conditions of the forward game, given in Equation (8), are a set of, generally nonlinear, equality constraints in the variables $\mathbf{x}, \mathbf{u}$, and $\boldsymbol{\lambda}$. To find a solution—that is, a root of $\mathbf{G}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda})$—we may employ a root-finding algorithm such as a variant of Newton's method (Nocedal and Wright 2006, Chapter 11). This is the approach taken by, e.g., Le Cleac'h et al. (2020).

Running example: *For our LQ example, it can be seen that a single step of Newton's method on $\mathbf{G}(\cdot)$ amounts to the well-known Riccati solution to an* open-loop *LQ game (Başar and Olsder 1999, Chapter 6).*[‡]

4 Problem Setup

Solving a *forward* Nash game amounts to identifying optimal strategies for all players, provided a priori knowledge of their objectives J^i. By contrast, in this work we are concerned with the *inverse* Nash problem, i.e., that of identifying players' objectives which explain their observed behavior. To develop the inverse Nash problem, here we shall presume that learning occurs offline, given a sequence of noisy, partial observations of all players' state. The method we develop for this setting, however, is amenable to trajectory prediction and online, receding horizon operation as discussed in Section 5.2.

[‡] Note that this Newton step *differs* from that given by the Riccati solution to a *feedback* LQ game.

We formulate the inverse Nash problem as one of offline learning, in which players' objectives belong to a known parametric function class. To that end, we make the following assumption.

Assumption 2. *(Parametric objectives) Player-i's cost function is fully described by a vector of parameters $\theta^i \in \mathbb{R}^{k^i}$. That is, $J^i(\cdot;\theta^i) \equiv \sum_{t=1}^{T} g_t^i(x_t, u_t^1, \dots, u_t^N; \theta^i)$.*

Recalling Assumption 1, the functions $g_t^i(\cdot;\theta^i)$ have well-defined derivatives in states x_t and controls u^i. We shall also extend this smoothness assumption to include the parameters themselves.

Assumption 3. *(Smoothness in parameter space) Extending Assumption 1, we require that stage cost functions $g_t^i(\cdot;\theta^i)$ have well-defined first- and second-derivatives with respect to the parameter vector θ^i.*

This smoothness assumption is quite general. For example, players' stage costs $g_t^i(\cdot;\theta^i)$ may be encoded as arbitrary function approximators such as artificial neural networks. In this work, we choose a more interpretable (though less flexible) parametric structure; we defer an investigation of more general cost structures for future work. In particular, the examples considered here use a *linearly-parameterized* structure in which $g_t^i(\cdot;\theta^i)$ is a linear function of θ^i, i.e., $g_t^i(\cdot;\theta^i) \equiv \theta^{i\top}\tilde{g}_t^i(\cdot)$ for some set of potentially nonlinear basis functions $\tilde{g}_t^i(\cdot)$. By incorporating appropriate domain-specific knowledge, however, these relatively simple cost structures are able to encapsulate complex, strategic interactions such as the highway lane changes of Figure 1.

Running example: *Recall the quadratic objectives of Equation (5), and take cost parameters $\theta^i = (\theta_Q^i, \theta_R^{ij})_{j\in[N]}$. Observe, therefore, that Player-i's objective depends linearly upon its cost parameters θ^i.*

Thus equipped, the objective learning problem reduces to maximizing the likelihood of a sequence of partial state observations $\mathbf{y} := (y_1, \dots, y_T)$ for the parametric class of games $\Gamma(\theta) = \big(x_1, f, \{J^{(i)}(\,\cdot\,;\theta^{(i)})\}_{i\in[N]}\big)$. Formally, we seek to solve a problem of the form

$$\max_{\theta,\mathbf{x},\mathbf{u}} \quad p(\mathbf{y} \mid \mathbf{x}, \mathbf{u}) \tag{12a}$$

$$\text{s.t.} \quad (\mathbf{x},\mathbf{u}) \text{ is an OLNE of } \Gamma(\theta) \tag{12b}$$

$$(\mathbf{x},\mathbf{u}) \text{ is dynamically feasible under } f, \tag{12c}$$

where θ aggregates all players' cost parameters, i.e., $\theta := (\theta^1, \dots, \theta^N)$, and $p(\mathbf{y} \mid \mathbf{x}, \mathbf{u})$ constitutes a known observation likelihood, or measurement, model.

Remark 1. *(Initial state) Observe that x_1 is an explicit decision variable in Equation (12), whereas it represents a constant (known) initial condition in the forward game problem discussed in Section 3. This reflects the fact that the state trajectory, including initial conditions, must be estimated as part of the inverse problem. As we shall see, estimating the state trajectory jointly with the cost parameters allows our method to be less sensitive to observation noise.*

This measurement model is arbitrary, though, following Assumption 1 and Assumption 3, it must be smooth. In the simplest instance, we may receive an exact measurement of the sequence of states and inputs for all players. In that case, the measurement model $p(\mathbf{y} \mid \mathbf{x}, \mathbf{u})$ reduces to a Dirac delta function. More generally, $p(\mathbf{y} \mid \mathbf{x}, \mathbf{u})$ may be an arbitrary smooth probability density function, making our formulation amenable to realistic sensors such as cameras or LiDARs.

Prior work in both single-player IOC, such as that of Englert and Toussaint (2018), and inverse games, such as those of Awasthi and Lamperski (2020) and Rothfuß et al. (2017), presumes a degenerate measurement model in which states and controls are observed directly without any noise. When perfect observations are unavailable, these methods naturally extend by first estimating a sequence of likely states and controls (a standard nonlinear filtering problem). In Section 6, we describe these sequential estimation methods in greater detail. In contrast, our formulation given in Equation (12) encodes a coupled estimation problem in which states, control inputs, and cost parameters must all be estimated simultaneously. Thus, our method exploits the additional coupling imposed by the Nash equilibrium constraints onto the unknowns. In Section 7, we conduct a series of Monte Carlo experiments to quantify the advantages afforded by simultaneous learning over sequential estimation.

5 Equilibrium-Constrained Cost Learning

Here we present our core contribution, a mathematical formulation of objective inference in multi-agent, noncooperative games. We express this problem as a nonconvex optimization problem with equilibrium constraints, which we relax into a standard-format equality-constrained nonlinear program.

5.1 Offline Learning

We first consider the problem of learning each player's objective from previously recorded data of prior interactions, *offline*.

Equation (12) is a mathematical program with equilibrium constraints (Luo et al. 1996; Ferris et al. 2005), with the nested equilibrium conditions of Equation (12b) encoding the Nash inequalities of Definition 1. Equilibrium constraints generalize bilevel programming, and computational approaches tend to be less mature than those for standard-form (in)equality-constrained programming.

We relax the equilibrium constraint of Equation (12b) by replacing it with its KKT conditions, i.e., by substituting Equation (8). This yields:

$$\max_{\theta,\mathbf{x},\mathbf{u},\boldsymbol{\lambda}} \quad p(\mathbf{y} \mid \mathbf{x}, \mathbf{u}) \tag{13a}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{x},\mathbf{u},\boldsymbol{\lambda};\theta) = \mathbf{0}. \tag{13b}$$

Here, we have explicitly written the KKT conditions from Equation (8) in terms of the cost parameters θ. Additionally, observe that in Equation (13), the costates $\boldsymbol{\lambda}$ required to evaluate the KKT conditions $\mathbf{G}(\cdot;\theta)$ appear as *additional primal variables*. The constraints of Equation (13b) will be assigned their own Lagrange multipliers, which are distinct from the original costates. By letting states, control inputs, and costates be primal variables, the KKT conditions $\mathbf{G}(\cdot)$ do not depend explicitly upon the observations $\mathbf{y}$. Thus, solving Equation (13) does not require complete state or input observations; rather, the equilibrium constraints of Equation (13b) allow us to reconstruct this missing information while we estimate cost parameters θ, simultaneously. Several remarks are in order.

Remark 2. *(Multiple observed trajectories) We have developed Equation (13) for the setting in which a single trajectory $(\mathbf{x}, \mathbf{u})$ has been observed, yielding a measurement sequence $\mathbf{y}$. However, our approach affords straightforward extension to settings in which player's objectives are learned from multiple demonstrations. In this instance, the primal variables $(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda})$ would be replicated for all trajectories, although the cost parameters θ would be shared. The objective given by Equation (13a) would be replaced by the joint probability of all measurements conditioned on all underlying trajectories, and the equilibrium constraints in Equation (13b) would be concatenated for all trajectories.*

Remark 3. *(Regularizing parameters) Depending upon the parametric structure of players' objectives $J^i(\cdot; \theta^i)$, and hence the structure of KKT residual $\mathbf{G}(\cdot; \theta)$, it can be critical to regularize and/or constrain cost parameters. For example, if there exists a choice of θ^i for Player-i such that $J^i(\mathbf{x}, \mathbf{u}; \theta^i)$ is constant for all dynamically-feasible trajectories $(\mathbf{x}, \mathbf{u})$, then every such trajectory would satisfy the equilibrium constraint of Equation (12b). Such choices of θ must be avoided, e.g., by regularizing or otherwise constraining parameters.*

Running example: *Following Remark 3, we constrain the parameters $\theta^i \geq c > 0$. Moreover, to account for scale invariance, we constrain their sum to unity, i.e., $\sum_{i\in[N]} \left(\theta^i_Q + \sum_{j\in[N]} \theta^{ij}_R\right) = 1$.*

5.1.1 Least Squares

A common observation model $p(\mathbf{y} \mid \mathbf{x}, \mathbf{u})$ is the additive white Gaussian noise (AWGN) model. Here, each observation y_t depends only upon the current state x_t and control inputs $\mathbf{u}_t$, i.e.,

$$y_t = h_t(x_t, \mathbf{u}_t) + n_t, \tag{14}$$

where the (potentially nonlinear) function h_t computes the expected measurement, and n_t is a zero-mean Gaussian white noise process with known covariance, i.e., $n_t \sim \mathcal{N}(0, \Sigma_t)$. In this case, following standard methods in maximum likelihood estimation (Gallager 2013), it is straightforward to express the maximization in Equation (13a) as nonlinear least squares by taking the negative logarithm of $p(\mathbf{y} \mid \mathbf{x}, \mathbf{u})$:

$$\min_{\theta,\mathbf{x},\mathbf{u},\boldsymbol{\lambda}} \quad \sum_{t=1}^{T} \big(y_t - h_t(x_t)\big)^\top \Sigma_t^{-1} \big(y_t - h_t(x_t)\big) \tag{15a}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}; \theta) = \mathbf{0}. \tag{15b}$$

In summary, this inverse problem entails the following task: Find those parameters θ for which the corresponding game solution generates expected observations near the observed data. This formulation of the inverse game problem can be encoded using well-established numerical modeling languages such as CasADi (Andersson et al. 2019) or JuMP (Dunning et al. 2017), and solved using off-the-shelf optimization routines such as IPOPT (Wächter and Biegler 2006) or SNOPT (Gill et al. 2005).

5.1.2 Problem Complexity

Let us examine the structure of the least squares problem in Equation (15) more carefully. In general, the observation map $h_t(\cdot)$ and KKT conditions $\mathbf{G}(\cdot; \cdot)$ may be arbitrarily nonlinear. Therefore, without further structural assumptions, our formulation is an equality-constrained nonlinear least squares problem. Due primarily to the nonlinearities in $\mathbf{G}$, Equation (15) is generally nonconvex. Solution methods, therefore, may be sensitive to initial values of primal variables; we discuss a straightforward initialization scheme in Section 6.1.

Perhaps surprisingly, this nonconvexity persists in the LQ setting of our running example, even when $h_t(\cdot)$ is the identity.

Running example: *Consider the LQ setting, with $\theta^i = (\theta^i_Q, \theta^{ij}_R)_{j\in[N]}$ as before. Let the observation map be the identity, i.e., $h_t(x_t) = x_t$ and presume AWGN. The resulting nonlinear least squares problem in Equation (15) has constraints of the form given in Equations (9) to (11). Let us consider the first of these constraints for a single time step t and Player-i:*

$$\mathbf{0} = \theta^i_Q Q^i_t x_t + \lambda^i_{t-1} - A^\top \lambda^i_t.$$

Recall that the decision variables in our formulation are $(\theta, \mathbf{x}, \mathbf{u}, \boldsymbol{\lambda})$. Here, we see that θ^i multiplies x_t. At best, therefore, this constraint is a bilinear *equality, making the overall problem in Equation (15) nonconvex even for this minimal inverse LQ game.*

When we directly observe both state and control inputs without noise, i.e., $y_t \equiv (x_t, \mathbf{u}_t)$, these constraints become *linear* even in the general non-LQ setting, so long as players' objectives are linearly parameterized. In this setting, we may rewrite Equation (9) as

$$\mathbf{0} = \nabla_{x_t} \overbrace{g^i_t(x_t, u_t; \theta^i)}^{\theta^{i\top}\tilde{g}^i_t(\cdot)} + \lambda^i_{t-1} - \nabla_{x_t} f_t(x_t, u_t)^\top \lambda^i_t \tag{16a}$$

$$= \theta^{i\top} \nabla_{x_t} \tilde{g}^i_t(x_t, u_t) + \lambda^i_{t-1} - \nabla_{x_t} f_t(x_t, u_t)^\top \lambda^i_t. \tag{16b}$$

With this observation model, then, the only decision variables are $(\theta^i, \lambda^i_t, \lambda^i_{t-1})$, which all appear linearly. Furthermore, the least squares objective in Equation (15a) becomes unnecessary, since, by assumption, the measurements $\mathbf{y}$ already include the states $\mathbf{x}$ exactly. Incorporating these simplifications, the entire constrained least squares problem of Equation (15) reduces to the problem

$$\text{find} \quad \theta, \boldsymbol{\lambda} \tag{17a}$$

$$\text{s.t.} \quad \mathbf{0} = \theta^{i\top} \nabla_{x_t} \tilde{g}^i_t(x_t, u_t) + \lambda^i_{t-1} - \nabla_{x_t} f_t(x_t, u_t)^\top \lambda^i_t, \forall i, t \tag{17b}$$

$$\mathbf{0} = \theta^{i\top} \nabla_{u^i_t} \tilde{g}^i_t(x_t, u_t) - \nabla_{u^i_t} f_t(x_t, u_t)^\top \lambda^i_t, \forall i, t. \tag{17c}$$

Because the constraints in Equation (17) are linear, the problem is equivalent to a linear system of equations. Moreover, since the constraints are completely decoupled for each player, they may be solved separately and in parallel for all players to obtain cost parameters θ^i and costates $\boldsymbol{\lambda}^i$. This reduction forms the basis for the state-of-the-art in solving inverse dynamic games (Rothfuß et al.

2017; Awasthi and Lamperski 2020), which only apply in settings with perfect state and input observations. To compare against these methods in more general settings that feature noise, unobserved inputs, and partial state measurements, we augment these methods with a sequential optimization procedure in Section 6. Comparative Monte Carlo studies of all approaches are presented in Section 7.

5.2 Online Learning

While Section 5.1 estimates the objectives of interacting agents from recorded data *offline*, our formulation for inverse Nash problems extends naturally to an *online* learning setting; i.e., cost learning from observations of ongoing interactions. As we shall discuss below, our method can perform online cost learning and trajectory prediction simultaneously, making it suitable for receding horizon applications.

5.2.1 Learning with Prediction Equipped with a tractable solution strategy for the setting of offline learning, we now consider a coupled *prediction* and learning problem. Similar problems have been considered in the single-agent setting by, e.g., (Jin et al. 2021; Mukadam et al. 2019). Here, we aim to learn the cost parameters θ from only a *subset* of the game horizon; i.e., we presume that observations $\mathbf{y} = (y_1, \dots, y_{\tilde{T}})$ where the observation horizon $\tilde{T} \leq T$. Despite this change, the original problem of Equation (12) remains effectively unchanged; only the objective has changed. In particular, by substituting the KKT conditions for an OLNE in place of the original equilibrium constraint as in Equation (13), and making AWGN assumptions, we recover a variant of the constrained least squares formulation of Equation (15):

$$\min_{\theta,\mathbf{x},\mathbf{u},\boldsymbol{\lambda}} \quad \sum_{t=1}^{\tilde{T}} \big(y_t - h_t(x_t)\big)^\top \Sigma_t^{-1} \big(y_t - h_t(x_t)\big) \tag{18a}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{x},\mathbf{u},\boldsymbol{\lambda};\theta) = \mathbf{0}. \tag{18b}$$

Note that the upper limit of addition is $\tilde{T}$, rather than T as in Equation (15a), while the OLNE KKT conditions in Equation (18b) depend upon states, inputs, and costates for all times $t \in \{1, \dots, \tilde{T}, \dots, T\}$.

Despite the similarities between this problem and Equation (15), the Nash trajectory $(\mathbf{x}^*, \mathbf{u}^*)$, which emerges as a solution affords a new interpretation. In particular, for times $t \leq \tilde{T}$ these equilibrium states and controls constitute filtered estimates of the observed quantities $\mathbf{y}$, while for times $t > \tilde{T}$ they represent *predictions* of the future. Importantly, however, extending trajectories beyond the observation horizon $\tilde{T}$ adds additional constraints to Equation (15). This ability to incorporate future, unobserved states makes the method more robust and data efficient when only a fraction of the game horizon is observed. Consequently, this formulation can be employed for online learning in scenarios of ongoing interactions. We provide a detailed empirical analysis of this setting in Section 7.2.2. A summary of this variant of our inverse game solver is provided in Figure 2(a).

5.2.2 Receding Horizon Learning Our method is directly amenable to receding horizon, online operation. Here, we suppose that the agents interact over the half-open time-interval $t \in \{1, \dots, \tilde{T}, \dots, \infty\}$, and that observations exist for $t \leq \tilde{T}$. Here, $\tilde{T}$ may be interpreted as the current time and, as time elapses, both $\tilde{T}$ and the overall prediction horizon T increase accordingly. Unfortunately, however, increasing the overall problem horizon increases the number of variables in Equation (12), eventually making the problem intractable.

To simplify matters, we approximate the learning problem at each instant by neglecting all times outside the interval $\{\tilde{T} - s_\mathrm{o}, \dots, \tilde{T}, \dots, \tilde{T} + s_\mathrm{p}\}$, where s_o is the length of a fixed-lag buffer of past observations, and s_p is the horizon of future state predictions. In this setting, the total number of variables remains constant (since the length of this interval is constant), rendering Equation (12) tractable to solve online. More precisely, at time $\tilde{T}$ (and under AWGN assumptions), we solve a modified version of Equation (18)

$$\min_{\theta,\mathbf{x},\mathbf{u},\boldsymbol{\lambda}} \quad \sum_{t=\tilde{T}-s_\mathrm{o}}^{\tilde{T}} \big(y_t - h_t(x_t)\big)^\top \Sigma_t^{-1} \big(y_t - h_t(x_t)\big) \tag{19a}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{x},\mathbf{u},\boldsymbol{\lambda};\theta) = \mathbf{0}, \tag{19b}$$

where the KKT constraint $\mathbf{G}(\cdot)$ is understood to depend upon times $t \in \{\tilde{T} - s_\mathrm{o}, \dots, \tilde{T}, \dots, \tilde{T} + s_\mathrm{p}\}$ and states, control inputs, and costates are also limited to that interval. At each later time, we solve a problem with identical structure, with the understanding that $\tilde{T}$ will have changed to reflect the elapsed time. In effect, this procedure amounts to simultaneous fixed-lag smoothing and receding-horizon prediction. We simulate this online learning procedure in Section 7.3.2.

6 Baseline

Recall the discussion of Section 5.1.2, in which we show that—with noiseless observations of states $\mathbf{x}$ and controls $\mathbf{u}$, and linear cost parameterization $g_t^i(\cdot;\theta^i) \equiv \theta^{i\top} \tilde{g}_t^i(\cdot)$—our formulation reduces to the linear system of equations of Equation (17). This reduction underlies state of the art methods for learning the objectives of players in games (Rothfuß et al. 2017; Awasthi and Lamperski 2020). Therefore, such methods unfortunately require noiseless observations of the full state and input sequences for all players. In contrast, our approach in Equation (13) is amenable to noisy, partial observations.

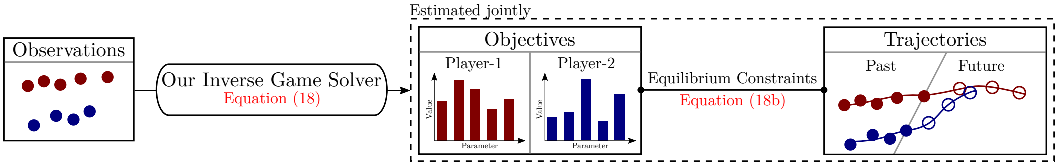
6.1 Recovering Unobserved Variables

To provide a meaningful comparison between our proposed technique and the state-of-the-art in settings with imperfect observations, we augment (Rothfuß et al. 2017; Awasthi and Lamperski 2020) with a pre-processing to estimate unobserved states and inputs. To that end, we solve the following relaxed version of Equation (13):

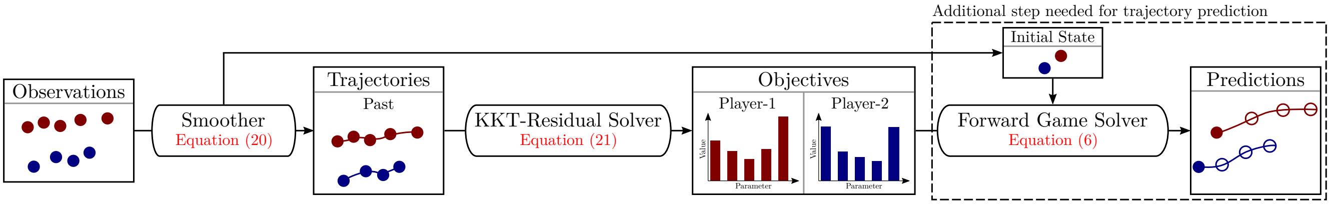
$$\tilde{\mathbf{x}}, \tilde{\mathbf{u}} := \arg\max_{\mathbf{x},\mathbf{u}} \quad p(\mathbf{y} \mid \mathbf{x}, \mathbf{u}) \tag{20a}$$

$$\text{s.t.} \quad \mathbf{F}(\mathbf{x},\mathbf{u}) = \mathbf{0}. \tag{20b}$$

As in Section 5.1.1, under a AWGN assumption Equation (20) becomes equality-constrained nonlinear least squares. However, unlike Equation (15), we have neglected



(a) Ours



(b) Baseline

Figure 2. Schematic overview of inverse game solvers set up for online operation. (a) Our method computes player's objectives, state estimates, and trajectory predictions jointly. (b) The baseline requires full knowledge of states and inputs and therefore must preprocess raw observations before it can estimate players' objectives. In order to generate trajectory predictions, the baseline must solve an additional forward game formulated over the estimated initial states and objectives.

the first two rows of the equilibrium constraint given in Equation (8). That is, Equation (20) computes a maximum likelihood estimate of states and inputs irrespective of the underlying game structure.

The solution of this smoothing problem is used as an estimate of states an inputs when the baseline is employed in partially observed settings. Beyond that, the same procedure serves as simple, yet effective initialization scheme for our method to tackle issues of non-convexity discussed in Section 5.1.2.

6.2 Minimizing KKT Residuals

Like our proposed method, the state-of-the-art methods developed by Rothfuß et al. (2017) and Awasthi and Lamperski (2020) use the forward game's KKT conditions to measure the quality of a set of cost parameters θ. While we compare to this derivative-based, KKT condition approach, we note that other approaches outlined in Section 2.2 such as (Le Cleac'h et al. 2021) utilize black-box optimization methods and do not require or exploit derivative information. These significant algorithmic differences—and the resulting differences in sample complexity, locality of solutions, etc.—make a direct comparison difficult to interpret.

Specifically, the KKT residual method of (Awasthi and Lamperski 2020; Rothfuß et al. 2017) fixes the state and input sequences to their observed—or in our case, estimated via Equation (20)—values. Fixing these variables, however, the resulting linearly-constrained satisfiability problem of Equation (17) may be infeasible, depending upon the parametric structure of costs $g_t^i(\cdot;\theta^i)$. In lieu, state-of-the-art approaches minimize the KKT residual itself, i.e.,

$$\min_{\theta,\boldsymbol{\lambda}} \|\mathbf{G}(\tilde{\mathbf{x}},\tilde{\mathbf{u}},\boldsymbol{\lambda};\theta)\|_2^2 . \tag{21}$$

In prior work (Awasthi and Lamperski 2020; Rothfuß et al. 2017), $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{u}}$ are assumed to be directly observed. As discussed in Section 6.1, here we presume they are the results of the pre-processing step given in Equation (20). Additionally, like the linear system of equations in Equation (17), the only decision variables here are the objective parameters θ and the costates $\boldsymbol{\lambda}$. In effect, the baseline does *not* refine the state and input estimates given by the pre-processing step of Equation (20). Furthermore, as in Equation (17), the problem may be decomposed into separate problems for each player and solved in parallel. In essence, then, this KKT residual formulation neglects the coupling between players' actions which is encoded in the equilibrium conditions; computationally, it reduces to solving separate IOC problems for each player neglecting game-theoretic interactions with others.

A schematic overview of this baseline approach is depicted in Figure 2. By first estimating the states $\mathbf{x}$ and inputs $\mathbf{u}$ from measurements $\mathbf{y}$, and only afterward learning the cost parameters θ and associated costates $\boldsymbol{\lambda}$, the KKT residual method can be thought of as a sequential decomposition of our approach. By contrast, our formulation maintains $(\mathbf{x},\mathbf{u})$ as decision variables and refines the initial guess of $(\tilde{\mathbf{x}},\tilde{\mathbf{u}})$ by *identifying all variables simultaneously*.

7 Experiments[§]

In this work, we develop a technique for learning players' objectives in continuous dynamic games from noise-corrupted, partial state observations. We conduct a series of Monte Carlo studies to examine the relative performance of our proposed methods and the KKT residual baseline in both offline and online learning settings.

7.1 Experimental Setup

We implement our proposed approach as well as the KKT residual baseline of Rothfuß et al. (2017) in the Julia programming language (Bezanson et al. 2017), using the mathematical modeling framework JuMP (Dunning et al. 2017). As a consequence, our implementation encodes an abstract description of Equation (13), making it straightforward to use in concert with a variety of optimization routines. In this work, we use the open

[§] Some result figures and descriptions are drawn from the earlier conference version of this work (Peters et al. 2021).

source COIN-OR IPOPT algorithm (Wächter and Biegler 2006). The source code for our implementation is publicly available.¶

To evaluate the relative performance of our proposed approach with the KKT residual baseline, we perform several Monte Carlo studies. The details of these studies are described below. However, all of these studies share the following overall setup: we fix a cost parameterization for each player, find corresponding OLNE trajectories as roots of Equation (8) using the well-known iterated best response (IBR) algorithm Wang et al. (2019), and simulate noisy observations thereof with additive white Gaussian noise (AWGN) as in Equation (14). Each study then presents samples across a different problem parameter to test the sensitivity of both approaches to observation noise (Sections 7.2.1 and 7.3.1) and unobserved time-steps (Section 7.2.2) in two different problem settings.

In each of the studies below, we consider N vehicles navigating traffic, and instantiate game dynamics and player objectives as follows. Each vehicle has its own state x^i such that the global game state is concatenated as $x = (x^1, \dots, x^N)$. Further, each vehicle follows unicycle dynamics at time discretization Δt:

$$x^i_{t+1} = \begin{cases} \textit{(x-position)}\ p^i_{x,t+1} & = p^i_{x,t} + \Delta t\, v^i_t \cos\psi^i_t \\ \textit{(y-position)}\ p^i_{y,t+1} & = p^i_{y,t} + \Delta t\, v^i_t \sin\psi^i_t \\ \textit{(heading)}\quad \psi^i_{t+1} & = \psi^i_t + \Delta t\, \omega^i_t \\ \textit{(speed)}\quad v^i_{t+1} & = v^i_t + \Delta t\, a^i_t, \end{cases} \tag{22}$$

where $u^i_t = (\omega_t, a_t)$ includes the yaw rate and longitudinal acceleration. Finally, each player's objective is characterized by a stage cost g^i_t which is a weighted sum of several basis functions, i.e.,

$$g^i_t = \sum_{\ell=1}^{5} w^i_\ell g^i_{\ell,t} \begin{cases} g^i_{1,t} = \mathbf{1}(t \geq T - t_{\text{goal}})\mathrm{d}(x^i_t, x^i_{\text{goal}}) & \text{(23a)} \\ g^i_{2,t} = -\sum_{j\neq i} \log(\|p_i - p_j\|^2_2) & \text{(23b)} \\ g^i_{3,t} = (v^i)^2 & \text{(23c)} \\ g^i_{4,t} = (\omega^i_t)^2 & \text{(23d)} \\ g^i_{5,t} = (a^i_t)^2. & \text{(23e)} \end{cases}$$

Here, the cost parameters $\theta^i = (w^i_\ell)_{\ell\in[5]}, w^i_\ell \in \mathbb{R}_+$ are positive weights for each cost component. Further, $p_i = (p^i_x, p^i_y)$ denotes the planar position of Player-i, and $\mathrm{d}(\cdot,\cdot)$ is an arbitrary distance mapping. For example, we may choose $\mathrm{d}(x^i_t, x^i_{\text{goal}}) = \|p^i_t - p^i_{\text{goal}}\|^2_2$ to compute squared distance from a fixed goal position. Note, however, that this map is generic and can also be used to encode more complex goal-reaching specifications as in the highway lane-changing example depicted in Figure 1. Taken together, the basis functions encode the following aspects of each player's preferences:

1. Be close to the goal state within the last t_{goal} time steps (23a)
2. Avoid close proximity to other vehicles (23b)
3. Avoid high speeds (23c)
4. Avoid large control efforts (23d, 23e)

Games of this form are inherently noncooperative since players must compete to reach their own goals efficiently while avoiding collision with one another. Hence, they must negotiate these conflicting objectives and thereby find an equilibrium of the underlying game.

In all of the Monte Carlo studies, we evaluate the approaches for two different noisy observation models h^{full}_t and h^{partial}_t. In $h^{\text{full}}_t(x_t) := x_t$, estimators observe the *full* state, and in $h^{\text{partial}}_t(x_t) := (p^1_t, \psi^1_t, \dots, p^N_t, \psi^N_t)$, estimators observe the position and heading but not the speed of each agent; i.e., they receive a *partial* state observation.

7.2 Detailed Analysis of a 2-Player Game

We first study the performance of our method in a simplified, $N = 2$-player game. This set of experiments demonstrates the performance gap of our approach and the KKT residual baseline in methods in a conceptually simple and easily interpretable scenario. Here, the game dynamics are given as in Equation (22), and player objectives are parameterized as in Equation (23). In particular, we let $\mathrm{d}(x^i_t, x^i_{\text{goal}}) = \|p^i_t - p^i_{\text{goal}}\|^2_2$. In summary, therefore, each vehicle wishes to reach a fixed, known goal position in the plane while avoiding collision with the other.

7.2.1 Offline Learning We begin by studying both our method's and the baseline's ability to infer the unknown objective parameters θ, as developed in Section 5.1. To do so, we conduct a Monte Carlo study for the aforementioned 2-player collision-avoidance application.

We generate 40 random observation sequences at each of 22 different levels of isotropic observation noise. For each of the resulting 880 observation sequences we run both our method and the baseline to recover estimates of weights $\theta^i = (w^i_\ell)_{\ell\in[5]}$ for each player. Note that in this offline setting both methods learn these objective parameters from noisy observations of a single, complete game trajectory. That is, each estimate relies upon 25 s of simulated interaction history from a single scenario.

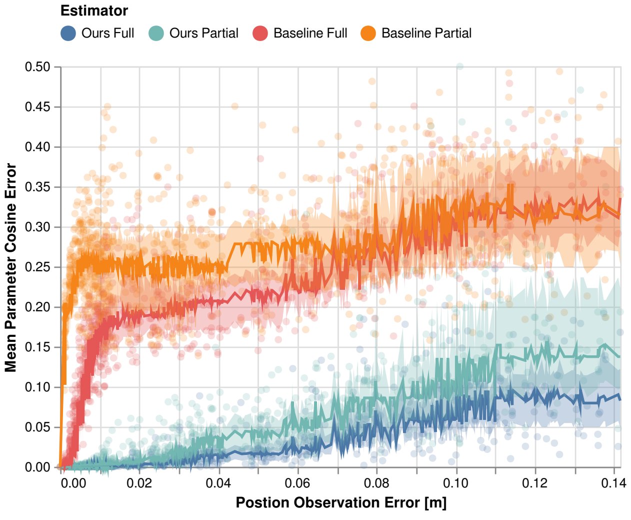
Figure 3 shows the estimator performance for varying levels of observation noise in two different metrics. Figure 3(a) reports the mean cosine error of the objective parameter estimates. That is, we measure cosine-dissimilarity between the unobserved true model parameters θ_{true} and the learned estimates θ_{est} according to

$$D_{\cos}(\theta_{\text{true}}, \theta_{\text{est}}) = 1 - \frac{1}{N} \sum_{i\in[N]} \frac{\theta^{i\top}_{\text{true}} \theta^i_{\text{est}}}{\left\|\theta^i_{\text{true}}\right\|_2 \left\|\theta^i_{\text{est}}\right\|_2}, \tag{24}$$

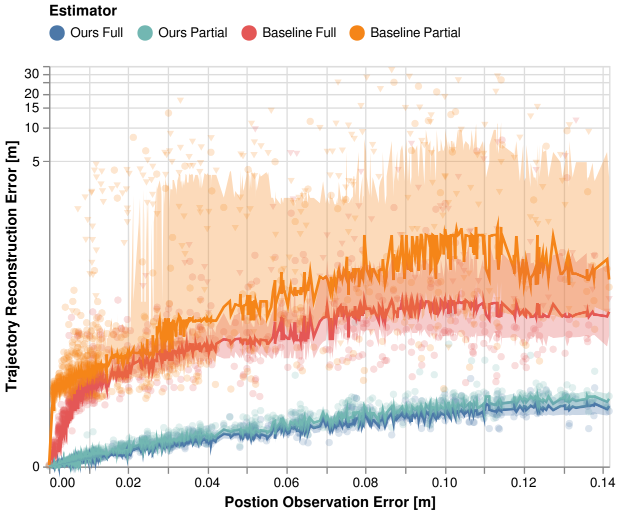
where the mean is taken over the N players. The normalization of the parameter vectors in Equation (24) reflects the fact that the absolute scaling of each player's objective parameters does not effect their optimal behavior, holding other players' parameters fixed. In sum, this metric measures the estimator performance in objective *parameter space*.

Figure 3(b) shows the mean absolute position error for trajectory *reconstructions* computed by finding a root of Equation (8) using the estimated objective parameters. Reconstruction error allows us to inspect the quality of learned cost parameters for explaining observed vehicle

¶https://github.com/PRBonn/PartiallyObservedInverseGames.jl



(a) Parameter estimation



(b) Trajectory reconstruction

Figure 3. Estimation performance of our method and the baseline for the 2-player collision-avoidance example, with noisy full and partial state observations. (a) Error measured directly in parameter space using Equation (24). (b) Error measured in position space using Equation (25). Triangular data markers in (b) highlight objective estimates which lead to ill-conditioned games. Solid lines and ribbons indicate the median and IQR of the error for each case.

motion, providing a more tangible metric of algorithmic quality. In addition to the raw data, we highlight the median as well as the interquartile range (IQR) of the estimation error over a rolling window of 60 data points.

Figure 3(a) shows that both our method and the baseline recover the true parameters θ reliably even for partial observations, if the observations are noiseless. However, the performance of the baseline degrades rapidly with increasing noise variance. This pattern is particularly pronounced in the setting of partial observations. On the other hand, our estimator recovers the unknown cost parameters more accurately in both settings, and with a smaller variance than the baseline. Thus, compared to the KKT residual baseline, the performance of our method degrades gracefully when both full and partial observations are corrupted by noise.

Next, we study these methods' relative performance as measured by reconstruction error, as shown in Figure 3(b). Here, reconstruction error is measured according to

$$D_{\text{rec}}(\theta_{\text{true}}, \theta_{\text{est}}) = \frac{1}{NT} \sum_{i \in [N]} \sum_{t \in [T]} \|p^i_{\text{rec},t} - p^i_{\text{true},t}\|_2, \quad (25)$$

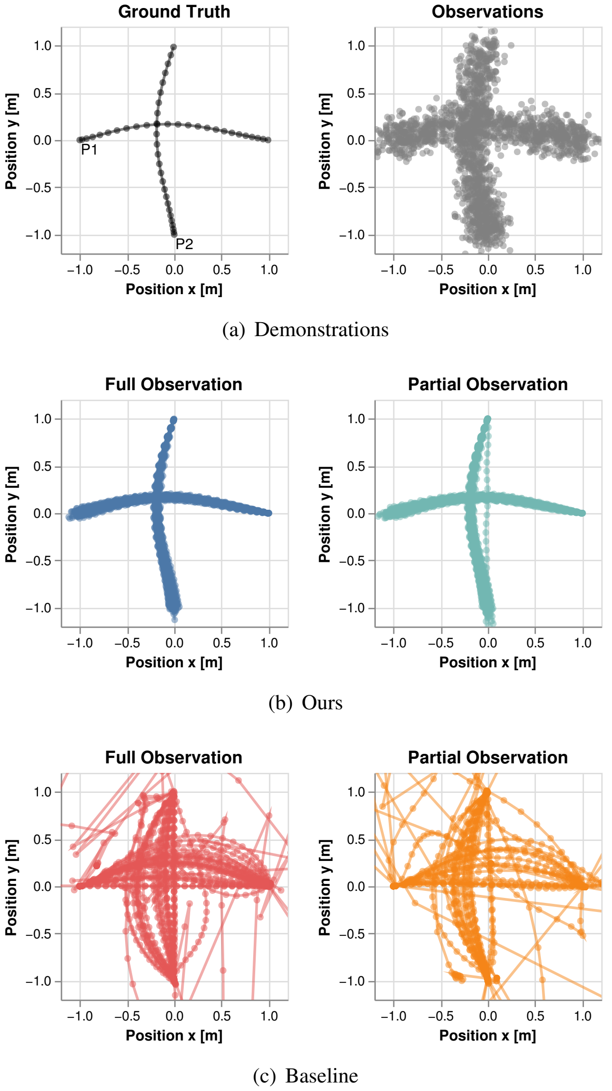
where $p^i_{\text{true},t}$ denotes the true position of Player-i at time step t, and $p^i_{\text{rec},t}$ denotes the position reconstructed from a Nash solution to the game with estimated cost parameters θ_{est}. We see similar patterns here as in the parameter error space, indicating the reliability of our method in both noisy full and partial observation settings.

Additionally, note that we have denoted some data points for the baseline method with triangular markers. For these Monte Carlo samples, the learned parameters θ_{est} specify ill-conditioned objectives that prevent us from recovering roots of Equation (8)—essentially rendering the parameter estimates useless for downstream applications. This can happen, for example, when proximity costs dominate control input costs. For the baseline, a total of 104 out of 880 estimates result in an ill-conditioned forward game when states are fully observed. In the case of partial observations, the number of learning failures increases to 218. In contrast, our method recovers well-conditioned player objectives for all demonstrations and allows for accurate reconstruction of the game trajectory.

For additional intuition of the performance gap, Figure 4 visualizes the reconstruction results in trajectory space for a fixed initial condition. Figure 4(a) shows the noise corrupted demonstrations generated for isotropic AWGN with standard deviation $\sigma = 0.1$. Figure 4(b) and Figure 4(c) show the corresponding trajectories reconstructed by solving the game using the objective parameters learned by our method and the baseline, respectively. Note that our method generates a far smaller fraction of outliers than the baseline. Furthermore, the performance of our method is only marginally effected by partial state observability, whereas baseline performance degrades substantially.

7.2.2 Online Learning with Prediction Next, we study the performance of both our proposed method and the KKT residual baseline in the setting of objective learning with prediction. Following the problem description of Section 5.2.1, here, only the beginning of an unfolding dynamic game is observed. This problem naturally describes a single time frame of online operations where observations accumulate as time evolves.

We conduct a Monte Carlo analysis of the two-player collision-avoidance game from Section 7.1 in which we vary the number of observed time steps of a fixed-length game. For this truncated observation sequence, each method is tasked to learn the players' underlying cost parameters θ^i *and* predict their motion for the next $s_{\text{p}} = 10$ time steps. Our method accomplishes these coupled tasks jointly by solving Equation (18). The KKT residual baseline, however, operates on the estimates provided the preceding smoothing step, therefore, cannot couple unobserved, future time steps with cost inference. Instead, it achieves this task in a two-stage procedure: First, parameter estimates are recovered from a truncated game over only the observed $\tilde{T}$ time steps. With these parameters in hand, the baseline then predicts

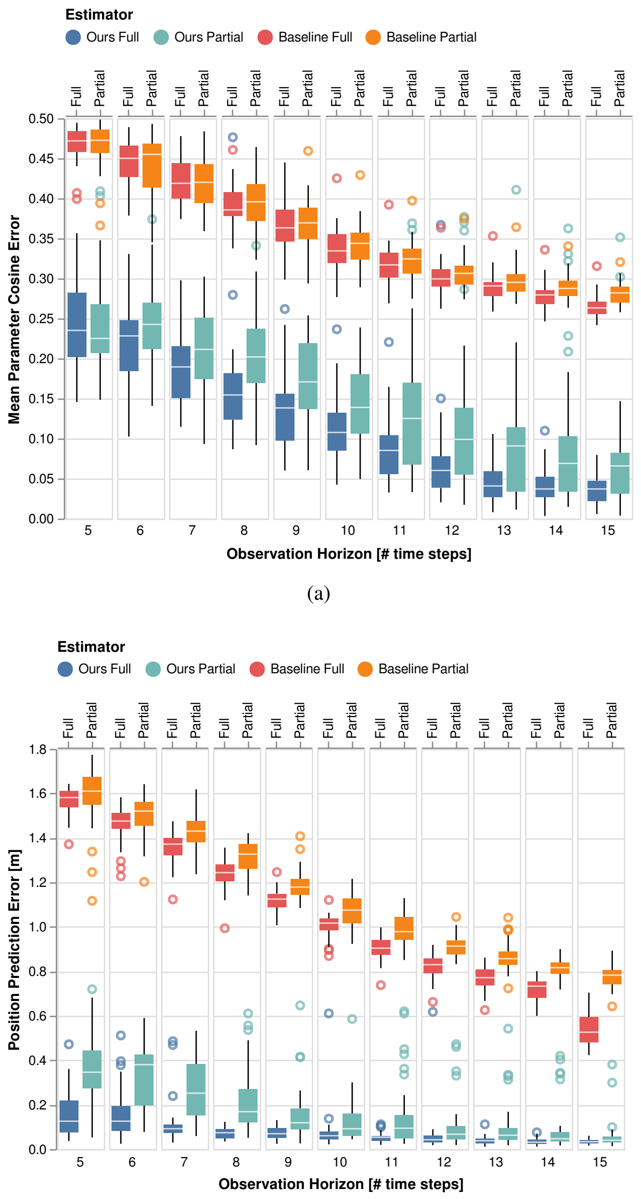


(a) Demonstrations

(b) Ours

(c) Baseline

Figure 4. Qualitative reconstruction performance for the 2-player collision avoidance example at noise level $\sigma = 0.1$ for 40 different observation sequences. (a) Ground truth trajectory and observations, where each player wishes to reach a goal location opposite their initial position. (b, c) Trajectories recovered by solving the game at the estimated parameters for our method and the baseline using noisy full and partial state observations. h



(a)

(b)

Figure 5. Estimation performance for our method and the baseline for varying numbers of observations of the 2-player collision-avoidance example at a fixed noise level of $\sigma = 0.05$. (a) Estimation performance measured directly in parameter space using Equation (24). (b) Prediction error over the next $10\,\mathrm{s}$ beyond the observation horizon using Equation (25).

future game states by re-solving a forward game starting from the final state estimate $\tilde{x}_{\tilde{T}}$ with time steps simulated from $t \in \{\tilde{T}, \dots, \tilde{T} + 10\}$.

In Figure 5, we vary the observation horizon $\tilde{T} \in \{5, \dots, 15\}$ for a ground-truth game played over 25 time steps. For each value of $\tilde{T}$, we sample 40 sequences of observations $\{y_t\}_{t=1}^{\tilde{T}}$. Here, we fix an isotropic Gaussian noise level of $\sigma = 0.05$, and measure the performance of both our method and the baseline using two distinct metrics. In Figure 5(a), we measure learning performance in parameter space using the metric given in Equation (24). As shown, our approach consistently estimates the cost parameters more accurately than the baseline. Furthermore, as the observation horizon $\tilde{T}$ increases, both methods improve. In Figure 5(b), we see that these patterns persist when we measure performance in trajectory space, applying the metric of Equation (25) to the predicted states $x_t, t \in \{\tilde{T}, \dots, \tilde{T} + 10\}$. Indeed, in this case, the performance gap is even more pronounced. By observing only $\tilde{T} = 5$ steps, our method reliably outperforms the baseline even when the baseline is given triple the number of observations.

To inspect these results more closely, in Figure 6 we show the output of both methods for a single observation sequence of length $\tilde{T} = 10$. This visualization highlights a key advantage of our approach compared with the baseline. In this scenario, Player-2 (bottom) turns left early on in order to avoid Player-1 (left) later along the path to its goal. Their ground truth trajectories are shown in black. However, the methods only receive noise-corrupted partial state observations of the first $\tilde{T} = 10$ time steps shown

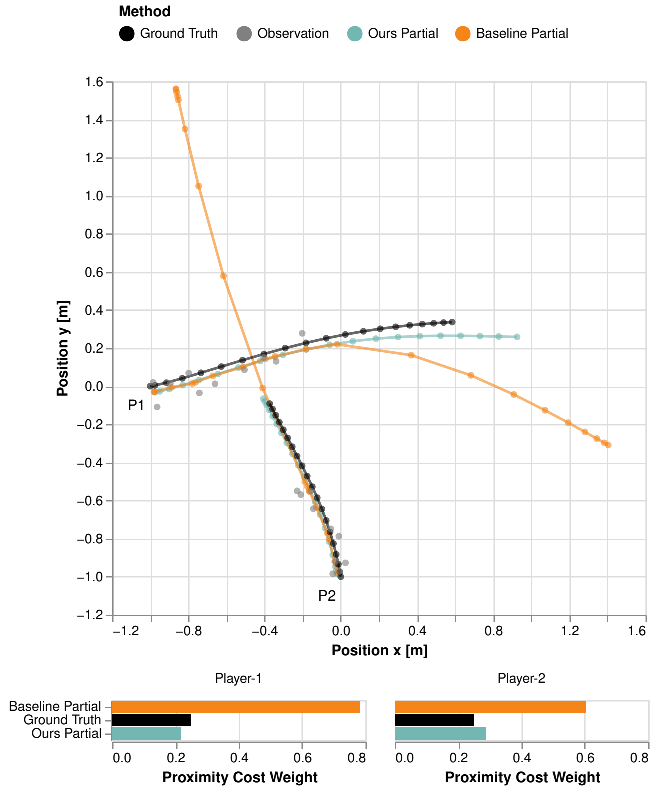


Figure 6. Qualitative prediction performance of our method and the baseline for the 2-player collision avoidance example when only the first 10 out of 25 time steps are observed.

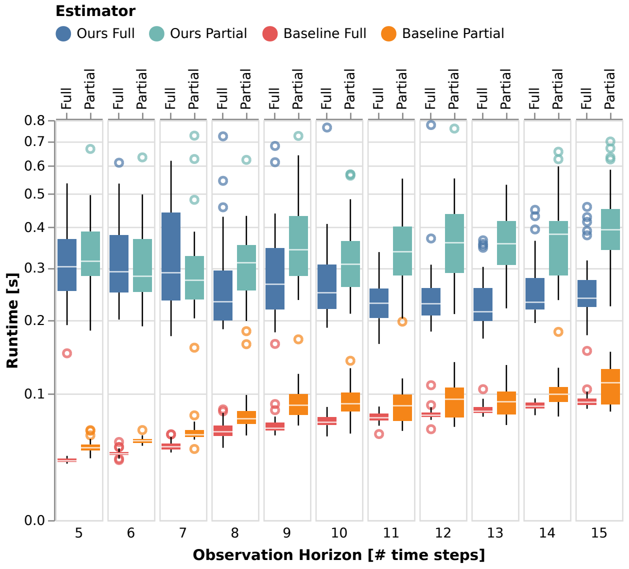


Figure 7. Runtime of our method and the baseline for varying numbers of observations of the 2-player collision-avoidance example at a fixed noise level of $\sigma = 0.05$.

in gray. Our method models the players' interactions as continuing into the future, allowing it to attribute observed behavior to future costs. In this instance, our method correctly explains Player-2's observed left turn as the result of a modest penalty on proximity, which becomes important only later in the trajectory when the players are close to one another. Cost estimation is shown at the bottom of Figure 6. The KKT residual baseline is incapable of such attributions. More precisely, it can only consider the KKT residuals $\mathbf{G}(\cdot;\theta)$ of Equation (21) for time steps $t \in [\tilde{T}]$. Hence, the baseline must presume that the game terminates at $\tilde{T}$ rather than at some time in the future. Thus, it cannot anticipate the immediate future consequences of particular cost models. In Figure 6, the baseline can only explain the players' early observed collision avoidance maneuver with an extremely large penalty on proximity to their opponents. As a result, it predicts that the players will quickly drive away from one another. Unlike our method, the baseline's prediction rapidly diverges from the ground truth.

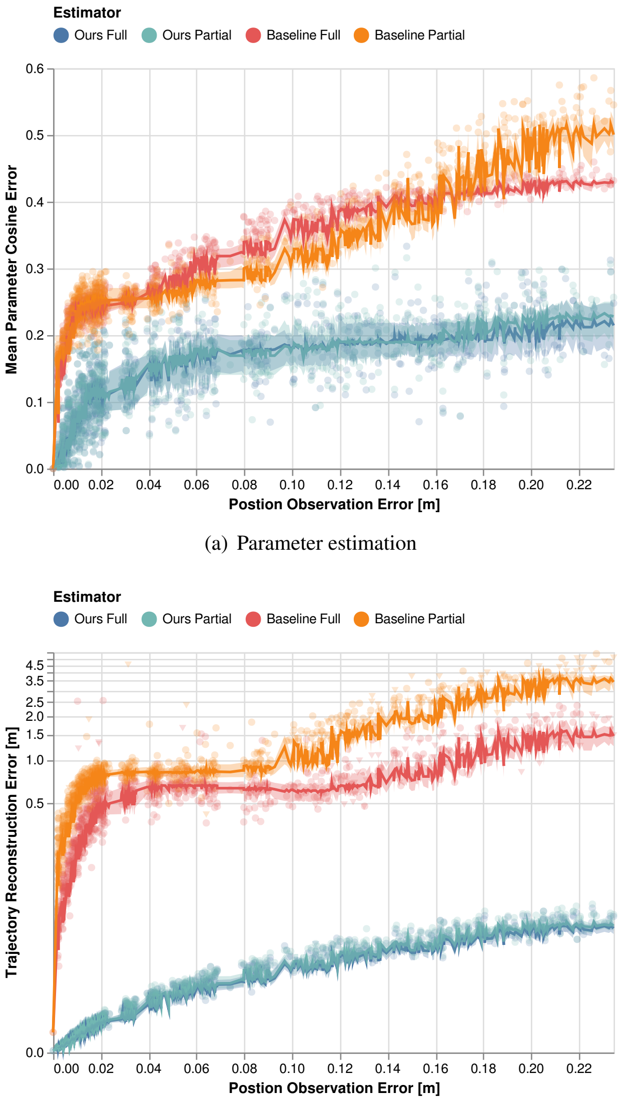
Beyond inference and prediction accuracy, a key factor for online operation is the computational complexity. To investigate this point, Figure 7 shows the computation time of both methods for the same dataset underpinning Figure 5. These timing results were obtained on a AMD Ryzen 9 5900HX laptop CPU. Overall, we observe that the KKT residual baseline has a lower runtime than our approach. The reduced runtime can be attributed to the fact that, by fixing the states and inputs a priori, the KKT residual formulation yields a simpler *convex* optimization problem in Equation (21). Nonetheless, our method's runtime still remains moderate and scales gracefully with the observation horizon. We note that our current implementation is not optimized for speed. In practical applications in the context of receding-horizon applications—a topic that we shall discuss in Section 7.3.2—the runtime may be further reduced via improved warm-starting and memory sharing across planner invocations.

7.3 Scaling to Larger Games

While our approach is more easily analyzed in the small, two-player collision-avoidance game of Section 7.2, it readily extends to larger multi-agent interactions. In order to demonstrate scalability of the approach, we therefore replicate the offline learning analysis of Section 7.2.1 in a larger 5-player highway driving scenario depicted in Figure 1. Finally, we demonstrate a proof of concept for online, receding horizon learning in this scaled setting following the setup of Section 5.2.

In the highway scenario discussed through the remainder of this section, each player wishes to make forward progress in a particular lane at an unknown nominal speed, rather than reach a desired position as above. Therefore, ground-truth objectives use a quadratic penalty on deviation from a desired state that encodes each player's target lane and preferred travel speed rather than a specific goal location. Despite these differences, this class of objectives is still captured by the cost structure introduced in Equation (23).

7.3.1 Offline Learning First, we study the performance of our method and the KKT residual baseline in the setting of offline learning without trajectory prediction. Figure 8 displays these results, using the same metrics as in Section 7.2.1 to measure performance in parameter space—Figure 8(a)—and position space—Figure 8(b). As before, our method demonstrably outperforms the baseline in both fully and partially observed settings. Furthermore, whereas our method performs comparably according to both metrics in the full and partial observation settings, the baseline performance differs between the two metrics. That is, while



(a) Parameter estimation

(b) Trajectory Reconstruction

Figure 8. Estimation performance of our method and the baseline for the 5-player highway overtaking example, with noisy full and partial state observations. (a) Error measured directly in parameter space using Equation (24). (b) Error measured in position space using Equation (25). Triangular data markers in (b) highlight objective estimates which lead to ill-conditioned games. Solid lines and ribbons indicate the median and IQR of the error for each case.

the performance of the baseline measured in parameter space is not significantly effected by less informative observations, the effect is significant in trajectory space. This inconsistency can be attributed the fact that certain objective parameters have stronger influence on the resulting game trajectory than others. Since our method's objective is observation fidelity, here measured by the measurement likelihood of Equation (13a), it directly accounts for these varying sensitivities. The baseline, however, greedily optimizes the KKT residual of Equation (21), irrespective of the resulting equilibrium trajectory.

7.3.2 *Online Learning and Receding Horizon Prediction*

Finally, we demonstrate the application of our method for simultaneous online learning and receding-horizon prediction in the 5-player highway navigation scenario depicted in Figure 1.

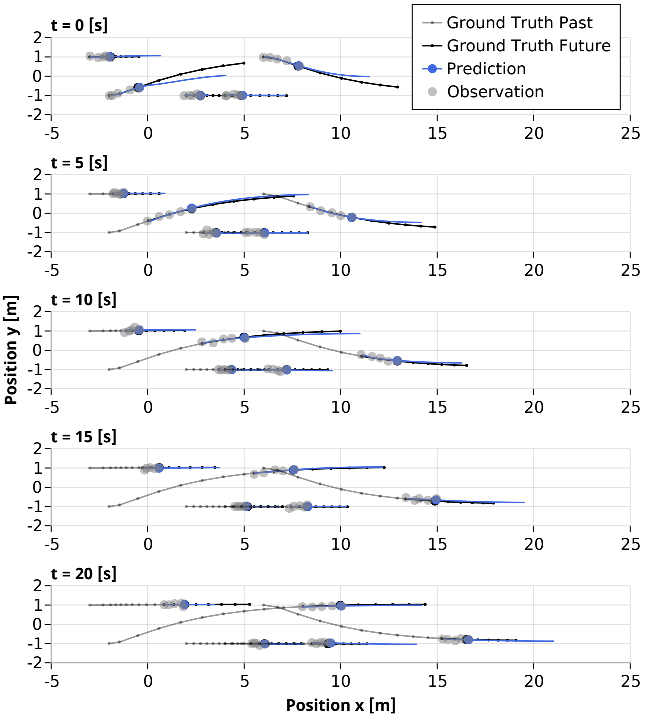


Figure 9. Demonstration of our method in an *online* application of simultaneous objective learning and trajectory prediction for the 5-player highway navigation scenario. At each time step, objective learning is performed on a fixed-lag buffer of $5\,\mathrm{s}$ of observation data which is coupled with trajectory prediction $10\,\mathrm{s}$ into the future.

Here, the information available to the estimator evolves over time and the problem only admits access to *past* observations of the game state for cost learning. Following the proposed procedure of Section 5.2, here, we limit the computational complexity of the estimation problem by considering only a fixed-lag buffer of observations over the last $5s$ and predict all player's behavior over the next $10s$. The qualitative performance of our method under noise-corrupted partial state observation is shown in Figure 9. As can be seen, from only a few seconds of data, our method learns player objectives that accurately predict the evolution of the game over a receding prediction horizon. Note that, by design, objective learning and behavior prediction is achieved *simultaneously* by solving a single joint optimization problem as in Equation (13). This ability to couple online learning and prediction makes it particularly suitable for online applications.

8 Conclusion

In this paper, we have introduced a novel approach to learn the parameters of players' objectives in dynamic, noncooperative interactions, given only noisy, partial observations. This *inverse* dynamic game arises in a wide variety of multi-robot and human-robot interactions and generalizes well-studied problems such as inverse optimal control, inverse reinforcement learning, and learning from demonstrations. Contrary to prior work, our method learns players' cost parameters while *simultaneously* recovering the forward game trajectory consistent with those parameters, with overall performance measured according to observation fidelity. We have shown how this formulation naturally extends to both *offline* learning and prediction problems, as well as *online*, receding horizon learning.

We have conducted extensive numerical simulations to characterize the performance of our method and compare it to a state-of-the-art baseline method (Rothfuß et al. 2017; Awasthi and Lamperski 2020). These simulations clearly demonstrate our method's improved robustness to both observation noise and partial observations. Indeed, existing methods presume noiseless, full state observations and thus require *a priori* estimation of states and inputs. Our method recovers objective parameters, reconstructs past game trajectories, and predicts future trajectories far more accurately than the baseline. Beyond that, our method's structure allows to perform all of these tasks jointly as the solution of a single optimization problem. This feature renders our method suitable for online learning and prediction in a receding horizon fashion.

In light of these encouraging results, there are several directions for future research. Most immediately, our method lends itself naturally to deployment onboard physical robotic systems such as the autonomous vehicles considered in the examples of Section 7. In particular, the online, receding horizon learning and prediction procedure of Section 5.2 may be run onboard an autonomous car. Here, the "ego" agent would seek to learn other vehicles' objective parameters while simultaneously using the receding horizon game solution to respond to predicted opponent strategies.

Another exciting, more theoretical direction consists of extending our formulation to more complex equilibrium concepts than OLNE. For example, recent solution methods for forward games in state feedback Nash equilibria (Fridovich-Keil et al. 2020; Laine et al. 2021; Di and Lamperski 2021) might be adapted to solve inverse games along the lines of Equation (12).

9 Declaration of Conflicts of Interest

The authors declare that there is no conflict of interest.

References

Albrecht S, Ramirez-Amaro K, Ruiz-Ugalde F, Weikersdorfer D, Leibold M, Ulbrich M and Beetz M (2011) Imitating human reaching motions using physically inspired optimization principles. In: *Proc. of the IEEE Intl. Conf. on Humanoid Robots*. IEEE.

Andersson JAE, Gillis J, Horn G, Rawlings JB and Diehl M (2019) CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation* 11(1): 1–36.

Awasthi C (2019) *Forward and Inverse Methods in Optimal Control and Dynamic Game Theory*. Master's Thesis, University of Minnesota.

Awasthi C and Lamperski A (2020) Inverse differential games with mixed inequality constraints. In: *Proc. of the IEEE American Control Conference (ACC)*. IEEE.

Başar T and Olsder GJ (1999) *Dynamic noncooperative game theory*, volume 23. Society for Industrial and Applied Mathematics (SIAM).

Bezanson J, Edelman A, Karpinski S and Shah VB (2017) Julia: A fresh approach to numerical computing. *SIAM Review (SIREV)* 59(1): 65–98.

Daskalakis C, Goldberg PW and Papadimitriou CH (2009) The complexity of computing a Nash equilibrium. *SIAM Journal on Computing* 39(1): 195–259.

Di B and Lamperski A (2019) Newton's method and differential dynamic programming for unconstrained nonlinear dynamic games. In: *Proceedings of the Conference on Decision Making and Control (CDC)*. IEEE.

Di B and Lamperski A (2021) Newton's method, Bellman recursion and differential dynamic programming for unconstrained nonlinear dynamic games. *Dynamic Games and Applications* : 1–49.

Dirkse SP and Ferris MC (1995) The path solver: a nommonotone stabilization scheme for mixed complementarity problems. *Optimization methods and software* 5(2): 123–156.

Dunning I, Huchette J and Lubin M (2017) JuMP: A modeling language for mathematical optimization. *SIAM Review (SIREV)* 59(2): 295–320.

Englert P and Toussaint M (2018) Inverse KKT: Learning cost functions of manipulation tasks from demonstrations. *Intl. Journal of Robotics Research (IJRR)* : 57–72.

Ferris MC, Dirkse SP and Meeraus A (2005) Mathematical programs with equilibrium constraints: Automatic reformulation and solution via constrained optimization. *Frontiers in applied general equilibrium modeling* : 67–93.

Fridovich-Keil D, Ratner E, Peters L, Dragan AD and Tomlin CJ (2020) Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games. In: *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*. IEEE.

Gallager RG (2013) *Stochastic processes: theory for applications*. Cambridge University Press.

Gill PE, Murray W and Saunders MA (2005) SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review (SIREV)* 47: 99–131.

Inga J, Bischoff E, Köpf F and Hohmann S (2019) Inverse dynamic games based on maximum entropy inverse reinforcement learning. *arXiv preprint arXiv:1911.07503* .

Isaacs R (1954-1955) Differential games i-iv. Technical report, RAND CORP SANTA MONICA CA SANTA MONICA.

Jin W, Kulić D, Mou S and Hirche S (2021) Inverse optimal control from incomplete trajectory observations. *Intl. Journal of Robotics Research (IJRR)* 40(6-7): 848–865.

Kalman RE (1964) When Is a Linear Control System Optimal? *ASME Journal of Basic Engineering* 86(1): 51–60. DOI: 10.1115/1.3653115.

Keshavarz A, Wang Y and Boyd S (2011) Imputing a convex objective function. In: *Proc. of the Intl. Symp. on Intelligent Control (ISIC)*. IEEE.

Köpf F, Inga J, Rothfuß S, Flad M and Hohmann S (2017) Inverse reinforcement learning for identification in linear-quadratic dynamic games. *IFAC-PapersOnLine* 50(1): 14902–14908.

Kretzschmar H, Spies M, Sprunk C and Burgard W (2016) Socially compliant mobile robot navigation via inverse reinforcement learning. *Intl. Journal of Robotics Research (IJRR)* 35(11): 1289–1307.

Laine F, Fridovich-Keil D, Chiu CY and Tomlin C (2021) The computation of approximate generalized feedback nash equilibria. *arXiv preprint arXiv:2101.02900* .

Le Cleac'h S, Schwager M and Manchester Z (2020) ALGAMES: A fast solver for constrained dynamic games. In: *Proc. of Robotics: Science and Systems (RSS)*.

Le Cleac'h S, Schwager M and Manchester Z (2021) LUCIDGames: Online unscented inverse dynamic games for adaptive trajectory prediction and planning. *IEEE Robotics and Automation Letters (RA-L)* 6(3): 5485–5492.

Levine S and Koltun V (2012) Continuous inverse optimal control with locally optimal examples. *Proc. of the Int. Conf. on Machine Learning (ICML)* .

Luo ZQ, Pang JS and Ralph D (1996) *Mathematical programs with equilibrium constraints*. Cambridge University Press.

Menner M and Zeilinger MN (2020) Maximum likelihood methods for inverse learning of optimal controllers. *arXiv preprint arXiv:2005.02767* .

Mombaur K, Truong A and Laumond JP (2010) From human to humanoid locomotion—an inverse optimal control approach. *Autonomous Robots* 28(3): 369–383.

Monderer D and Shapley LS (1996) Potential games. *Games and economic behavior* 14(1): 124–143.

Mukadam M, Dong J, Dellaert F and Boots B (2019) Steap: simultaneous trajectory estimation and planning. *Autonomous Robots* 43(2): 415–434.

Natarajan S, Kunapuli G, Judah K, Tadepalli P, Kersting K and Shavlik J (2010) Multi-agent inverse reinforcement learning. In: *2010 ninth international conference on machine learning and applications*. IEEE, pp. 395–400.

Ng AY and Russell SJ (2000) Algorithms for inverse reinforcement learning. In: *Proc. of the Int. Conf. on Machine Learning (ICML)*.

Nocedal J and Wright S (2006) *Numerical optimization*. Springer Verlag.

Peters L (2020) *Accommodating Intention Uncertainty in General-Sum Games for Human-Robot Interaction*. Master's Thesis, Hamburg University of Technology.

Peters L, Fridovich-Keil D, Rubies-Royo V, Tomlin CJ and Stachniss C (2021) Inferring objectives in continuous dynamic games from noise-corrupted partial state observations. In: *Proc. of Robotics: Science and Systems (RSS)*.

Rothfuß S, Inga J, Köpf F, Flad M and Hohmann S (2017) Inverse optimal control for identification in non-cooperative differential games. *IFAC-PapersOnLine* 50(1): 14909–14915.

Šošić A, KhudaBukhsh WR, Zoubir AM and Koeppl H (2016) Inverse reinforcement learning in swarm systems. *arXiv preprint arXiv:1602.05450* .

Wächter A and Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106(1): 25–57.

Wang Z, Spica R and Schwager M (2019) Game theoretic motion planning for multi-robot racing. *Distributed Autonomous Robotic Systems* : 225–238.

Ziebart BD, Maas AL, Bagnell JA and Dey AK (2008) Maximum entropy inverse reinforcement learning. In: *Proc. of the Conference on Advancements of Artificial Intelligence (AAAI)*.