IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 October 2023, accepted 22 October 2023, date of publication 25 October 2023, date of current version 1 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3327426

METHODS

Triangle Inequality for Inverse Optimal Control

SHO MITSUHASHI[1] **AND SHIN ISHII**[1,2]
[1]Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
[2]Advanced Telecommunications Research Institute International (ATR), Seika 619-0288, Japan

Corresponding author: Sho Mitsuhashi (mitsuhashi.sho.75p@st.kyoto-u.ac.jp)

This work was supported in part by the Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science (JSPS) KAKENHI, Japan, under Grant (JP22H04998, 23H04676); in part by the New Energy and Industrial Technology Development Organization (NEDO), Japan; and in part by the Establishment of University Fellowships Toward the Creation of Science Technology Innovation, Japan Science and Technology Agency (JST), Japan, under Grant JPMJFS2123.

ABSTRACT Inverse optimal control (IOC) is a problem of estimating a cost function based on the behaviors of an expert that behaves optimally with respect to the cost function. Although the Hamilton-Jacobi-Bellman (HJB) equation for the value function that evaluates the temporal integral of the cost function provides a necessary condition for the optimality of expert behaviors, the use of the HJB equation alone is insufficient for solving the IOC problem. In this study, we propose a triangle inequality which is useful for estimating the better representation of the value function, along with a new IOC method incorporating the triangle inequality. Through several IOC problems and imitation learning problems of time-dependent control behaviors, we show that our IOC method performs substantially better than an existing IOC method. Showing our IOC method is also applicable to an imitation of expert control of a 2-link manipulator, we demonstrate applicability of our method to real-world problems.

INDEX TERMS Cost estimation, imitation learning, inverse optimal control, inverse reinforcement learning.

I. INTRODUCTION

The optimal control problem (OCP) is the problem of finding optimal controls which minimize a specified objective function within a dynamical system, typically in continuous state, control signal (action), and time spaces [1]. This is advantageous over reinforcement learning (RL) because RL primarily focuses on Markov decision processes (MDPs), which often require certain restrictions, such as discrete state and action spaces [2]. Inverse optimal control (IOC), an inverse problem of OCP [3], inherits these characteristics when compared to its RL counterpart, inverse reinforcement learning (IRL) [4]. IOC methods estimate a cost function based on the observation of expert behaviors, assuming that the expert has performed optimally according to the cost function. Consequently, IOC presents several limitations when compared to IRL, particularly in its inability to accommodate stochastic controllers and environments. Nonetheless, IOC also offers numerous advantages, such as its capacity to handle a time-dependent cost function, a feature that is usually unavailable in IRL methods rooted in MDPs. This capability arises because the expert's cost function can be time-dependent in OCP [1], [5]. Therefore, IOC offers distinct advantages over IRL, as it possesses the potential to naturally address a time-dependent cost function as well as continuous variables in state and action spaces.

Interesting applications of inverse approaches such as IOC and IRL lie in imitation learning [6]. By solving an OCP with the cost function estimated by IOC, the optimal controlling behaviors demonstrated by the expert can be imitated. The setting in which an imitator solves an RL problem with the reward function estimated by the IRL is called apprenticeship learning [7]. In these imitation methods, designing complex reward/cost functions can be avoided by performing optimal behaviors based on the estimated reward/cost function [8]. Imitation learning that incorporates the inverse approach can be further advantageous over other imitation learning methods such as behavior cloning [9], [10]. Since the estimated cost function generalizes the expert objective, imitation learning with the inverse approach is effective even when the expert and imitator are in different environments (i.e., different system dynamics) [8], [11].

However, solving an IOC problem is difficult because of the ill-posedness of the cost function (i.e., expert

The associate editor coordinating the review of this manuscript and approving it for publication was Pinjia Zhang.

demonstrations are consistent with multiple cost functions). The value function, which evaluates the temporal integral of the cost function, is helpful in estimating the cost function. The Hamilton-Jacobi-Bellman (HJB) equation for the value function provides the necessary condition for the optimality of expert behaviors. However, the use of the HJB equation alone is insufficient for solving IOC well, because the HJB equation presents the local optimality just around the expert behaviors. In this study, we propose the use of a triangle inequality that presents the non-optimality of any bypath that goes through a via-point on a non-optimal trajectory. Because this inequality provides additional information about the value function, its use in IOC can improve the IOC solution by mitigating the ill-posedness possessed by the inverse problem.

Although the idea of triangle inequality can be applied to general IOC problems, we show several applications to time-dependent IOC problems; that is, the underlying cost function that the expert has used is dependent on time. Examples of such tasks include path tracking control of a moving target [12] and autonomous driving [13]. These time-dependent tasks can easily be found in the real world; therefore, well-established modern control methods, such as model predictive control, have been applied to time-dependent tasks [1], [14]. Considering this demand, we demonstrate the application of our new IOC method to the imitation learning of time-dependent tasks. To our knowledge, there are no prior studies that explicitly showed the imitation learning based on the time-dependent cost function estimated by IOC. The IOC methods for time-dependent tasks have not been studied much.

In existing studies on IOC and IRL, the value and cost functions have been approximated in various forms, such as neural networks [15], linear combinations of features such as Gaussian RBFs [16], [17], [18], and polynomials [3], [19]. In this study, we present a constrained linear-programming-based algorithm with polynomial approximation assumptions for the value and cost functions. Because this algorithm does not rely on a stochastic approximation, the implementation of the triangle inequality is straightforward.

II. BACKGROUND

A. OPTIMAL CONTROL THEORY

In optimal control theory, the value function $v(\mathbf{x})$, which represents the minimum total cost when moving from an arbitrary state $\mathbf{x}$ to a state in the terminal state set X_T, plays a central role. An OCP is a problem for obtaining optimal control sequence $\mathbf{u}(\cdot)$ that achieves the value function $v(\mathbf{x}_0)$ from a given initial state $\mathbf{x}_0$.

$$v(\mathbf{x}) = \min_{\mathbf{u}(\cdot), T(\geq t_0)} \int_{t_0}^{T} l(\mathbf{x}(\tau), \mathbf{u}(\tau)) d\tau \\ s.t.\ \ \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \mathbf{x}(t_0) = \mathbf{x},\ \mathbf{x}(T) \in X_T \quad (1)$$

Integrand l is the cost function that represents a scalar cost for a pair of state $\mathbf{x}$ and control $\mathbf{u}$. The value function $v(\mathbf{x})$ denotes the minimum cost integrated from the initial time t_0 to the terminal time T under the following three constraints: $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ is the system dynamics, which is assumed to be known throughout this study, $\mathbf{x}(t_0) = \mathbf{x}$ is the initial condition, and $\mathbf{x}(T) \in X_T$ is the terminal condition.

Using the value function, we can obtain the HJB equation, which is a necessary condition for optimal control $\mathbf{u}$ at any state $\mathbf{x}$.

$$0 = \min_{\mathbf{u}} \left\{ l(\mathbf{x}, \mathbf{u}) + \frac{\partial v}{\partial \mathbf{x}}^T (\mathbf{x}) \mathbf{f}(\mathbf{x}, \mathbf{u}) \right\} \quad (2)$$

Relaxing the HJB equation yields the following inequality:

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}) := l(\mathbf{x}, \mathbf{u}) + \frac{\partial v}{\partial \mathbf{x}}^T (\mathbf{x}) \mathbf{f}(\mathbf{x}, \mathbf{u}) \geq 0. \quad (3)$$

This implies that given a cost function $l(\mathbf{x}, \mathbf{u})$ and a value function $v(\mathbf{x})$, $\mathcal{L}(l, v)$ should not be negative for any pair of state $\mathbf{x}$ and control $\mathbf{u}$. The equality in (3) holds only when the control $\mathbf{u}$ is optimal at the state $\mathbf{x}$.

B. INVERSE OPTIMAL CONTROL WITH POLYNOMIAL OPTIMIZATION

IOC is an inverse problem of OCP, which estimates the cost function given a trajectory of the optimal control $(\mathbf{x}_0, \mathbf{u}_0, t_0), \ldots, (\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1})$. Here, we explain the linear-programming-based IOC method presented by Pauwels and colleagues [3], which was used as a baseline method in this study. Although the authors did not show applications to time-dependent IOC problems, their method could estimate a time-dependent cost function. For convenience, we describe a simplified version of the Pauwels' method in which the cost and value functions are assumed to be independent of time. This baseline IOC method estimates both the cost and value functions by optimizing the coefficients of the polynomial function approximators, which are designed to have all monomial bases up to the degree given as a hyperparameter. This method assumes that the dynamics of a system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ is given by a polynomial vector, and the domains of the state and control space X and U, are compact basic semi-algebraic sets of the form $X = \{\mathbf{x} | g_i(\mathbf{x}) \geq 0, i = 1, \ldots, m\}$, $U = \{\mathbf{u} | k_j(\mathbf{u}) \geq 0, j = 1, \ldots, h\}$ with $g_i (i = 1, \ldots, m)$ and $k_j (j = 1, \ldots, h)$ being polynomials of $\mathbf{x}$ and $\mathbf{u}$, respectively.

IOC is a problem of estimating the hidden cost function of an optimally behaving expert given the trajectory of the expert, $(\mathbf{x}_0, \mathbf{u}_0, t_0), \ldots, (\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1})$. In the baseline method, the cost function is recovered by solving the following constrained optimization problem:

$$\inf_{l, v, \epsilon} \epsilon + \lambda \|l\|_1 \quad (4a)$$

$$\text{s.t.} \quad \mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}) \geq 0, \forall (\mathbf{x}, \mathbf{u}) \in X \times U \quad (4b)$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \mathcal{L}(l, v)(\mathbf{x}_i, \mathbf{u}_i) \leq \epsilon \quad (4c)$$

$$v(\mathbf{x}) = 0, \forall \mathbf{x} \in X_T \quad (4d)$$

$$\mathcal{A}(\mathcal{L}(l, v)) = 1 \quad (4e)$$

Equation (3) leads to (4b) and (4c), and (4c) is an epsilon relaxation of the equality condition. Equation (4d) requires the value function in the terminal state to be zero. In (4e), $\mathcal{A}$ is a linear functional constraint for preventing the HJB function $\mathcal{L}$ from becoming a trivial function, such as the zero function; in our implementation the coefficient summation of the polynomial $\mathcal{L}(l, v)$ is restricted to unity. Importantly, there exists flexibility in designing the linear functional $\mathcal{A}$. This flexibility enables the incorporation of domain-specific prior knowledge into the estimation process. Equation (4a) attempts to minimize the slack variable ϵ plus L1-based regularizer of the coefficients of the cost function l; $\lambda > 0$ is a hyperparameter that controls the strength of the regularizer. $\inf_{l,v,\epsilon}$ indicates that this optimization problem is minimized by optimizing the coefficients of the polynomial function approximators for the cost l and value v functions, and the slack variable ϵ.

III. INVERSE OPTIMAL CONTROL WITH TRIANGLE INEQUALITY

Although the Pauwels' method is simple and widely applicable, its solution would not necessarily be good because of the shortage of constraints; the HJB equation only imposes constraints on the *derivative* of the value function, and represents the optimality condition of the expert behaviors only around the expert trajectory.

To address this constraint shortage problem, we present a new IOC method based on the triangle inequality, which is an inequality for the *value* itself of the value function. Section III-A introduces the triangle inequality that should exist behind the expert optimal trajectory. Section III-B describes the new IOC method which incorporates the triangle inequality.

A. TRIANGLE INEQUALITY

Here, we derive the triangle inequality in a simple time-independent setting, in which the cost and value functions taken by the expert are independent of time; however, its extension to address time-dependent settings is straightforward. The triangle inequality and IOC method for time-dependent settings are described in Appendix A.

Fig. 1 depicts the concept of triangle inequality in an OCP in two-dimensional state space. We assumed that route A is the optimal route with the minimal total cost from the initial state $\mathbf{x}_0$ to any terminal state in set X_T. Route B $\rightarrow$ C is the optimal route when it is constrained to pass through a via-point $\mathbf{x}$ that is not on the optimal route. The triangle inequality states that the total cost of B $\rightarrow$ C should be larger than that of A for any via-point $\mathbf{x}$.

Because the value function is defined as the minimum total cost to reach any terminal state, the total costs of A and C are given by $v(\mathbf{x}_0)$ and $v(\mathbf{x})$, respectively. However, the minimal total cost of Route B cannot be represented by the value function. To this end, we introduce an alternative value

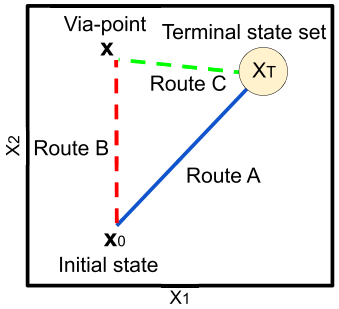


FIGURE 1. Conceptual diagram of triangle inequality.

function in the "time-reversed" OCP, which is given by

$$rv(\mathbf{x}) := \min_{\mathbf{u}(\cdot), t(\geq t_0)} \int_t^{t_0} -l(\mathbf{x}(\tau), \mathbf{u}(\tau))d\tau \quad s.t.\ \ \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \mathbf{x}(t) = \mathbf{x}, \mathbf{x}(t_0) = \mathbf{x}_0. \tag{5}$$

The left-hand side of (5), called the **reverse value function** in this study, denotes the minimal total *negative* cost from the via-point $\mathbf{x}$ to the initial state $\mathbf{x}_0$ in a backward manner. By reversing the integral interval, the reverse value function $rv(\mathbf{x})$ is shown to be equivalent to the minimum total cost from the initial state $\mathbf{x}_0$ to the arbitrary via-point $\mathbf{x}$. Therefore, the minimal total cost of Route B is given by $rv(\mathbf{x})$.

Accordingly, the triangle inequality is expressed as:

$$\begin{aligned} v(\mathbf{x}_0) &= \min_{\mathbf{u}(\cdot), T(\geq t_0)} \int_{t_0}^{T} l(\mathbf{x}(\tau), \mathbf{u}(\tau))d\tau \ (\mathbf{x}(T) \in X_T) \\ &\leq \min_{\mathbf{u}(\cdot), T'(\geq t_0)} \int_{t_0}^{T'} l(\mathbf{x}(\tau), \mathbf{u}(\tau))d\tau \\ &\quad (\mathbf{x}(T') \in X_T, \exists t \in [t_0, T']\ \mathbf{x}(t) = \mathbf{x}) \\ &= rv(\mathbf{x}) + v(\mathbf{x}), \end{aligned} \tag{6}$$

where the constraints $\mathbf{x}(t_0) = \mathbf{x}_0$ and $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ are omitted from parentheses for visibility. When the via-point $\mathbf{x}$ is on optimal Route A, the equality in (6) holds. If Route A is the sole optimal route in the state space, the inequality in (6) should hold strictly for any via-point that is not on Route A.

HJB Equation for the Reverse Value Function: Similar to the value function on the "time-forward" OCP, another HJB equation in (7) also holds for the reverse value function. The equality of this HJB equation is satisfied by the expert trajectory $(\mathbf{x}_0, \mathbf{u}_0, t_0), \ldots, (\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1})$ because the expert trajectory is optimal in both "time-forward" and "time-reversed" OCPs. The derivation is described in Appendix B.

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}) := l(\mathbf{x}, \mathbf{u}) - \frac{\partial rv}{\partial \mathbf{x}}^T(\mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}) \geq 0 \tag{7}$$

B. IMPLEMENTATION OF THE TRIANGLE INEQUALITY

In this subsection, we develop our new IOC method with the triangle inequality in time-independent settings. Our IOC method (8) is an extension of the baseline IOC method (4), with the additional constraints from (8b) to (8l), with the following assumptions: 1) the system dynamics is given

by a polynomial vector, 2) $\mathbf{x}_{n-1} \in X_T$ is satisfied by the expert trajectory $(\mathbf{x}_0, \mathbf{u}_0, t_0), \ldots, (\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1})$, and 3) the domains of state and control space are given as compact basic semi-algebraic sets of the form $X = \{\mathbf{x} | g_i(\mathbf{x}) \geq 0, i = 1, \ldots, m\}$, $U = \{\mathbf{u} | k_j(\mathbf{u}) \geq 0, j = 1, \ldots, h\}$ with $g_i (i = 1, \ldots, m)$ and $k_j (j = 1, \ldots, h)$ being polynomials of $\mathbf{x}$ and $\mathbf{u}$, respectively.

$$\inf_{l,v,rv,\epsilon_a,\epsilon_b,\epsilon_c} \epsilon_a + \epsilon_b + \epsilon_c + \lambda \|l\|_1 \tag{8a}$$

s.t. -Constraints from the baseline method-

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}) \geq 0, \forall (\mathbf{x}, \mathbf{u}) \in X \times U \tag{8b}$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \mathcal{L}(l, v)(\mathbf{x}_i, \mathbf{u}_i) \leq \epsilon_a \tag{8c}$$

$$v(\mathbf{x}) = 0, \forall \mathbf{x} \in X_T \tag{8d}$$

$$\mathcal{A}(\mathcal{L}(l, v)) = 1 \tag{8e}$$

-Constraints from the introduction of rv-

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}) \geq 0, \forall (\mathbf{x}, \mathbf{u}) \in X \times U \tag{8f}$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \mathcal{RL}(l, rv)(\mathbf{x}_i, \mathbf{u}_i) \leq \epsilon_b \tag{8g}$$

$$rv(\mathbf{x}_0) = 0 \tag{8h}$$

$$\mathcal{A}(\mathcal{RL}(l, rv)) = 1 \tag{8i}$$

$$rv(\mathbf{x}_{n-1}) = v(\mathbf{x}_0) \tag{8j}$$

-Constraints from the triangle inequality-

$$v(\mathbf{x}) + rv(\mathbf{x}) \geq v(\mathbf{x}_0), \forall \mathbf{x} \in X \tag{8k}$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \{v(\mathbf{x}_i) + rv(\mathbf{x}_i) - v(\mathbf{x}_0)\} \leq \epsilon_c \tag{8l}$$

Here, we describe how the additional constraints from (8b) to (8l) have been derived. Constraints (8k) and (8l) originate from the triangle inequality (6). Constraint (8k) requires the triangle inequality to hold for any state $\mathbf{x}$ in domain X. Constraint (8l) is the epsilon relaxation of the equality constraint, which should hold for any state $\mathbf{x}$ on the expert trajectory. The constraints on the reverse value function from (8f) to (8i) correspond to the constraints on the value function from (8b) to (8e), and have been introduced to identify the reverse value function using constrained linear programming. Furthermore, (8j) should be satisfied because $rv(\mathbf{x}_{n-1})$ and $v(\mathbf{x}_0)$ share the same optimal trajectory.

1) APPLICATION TO MULTIPLE TRAJECTORIES SETTINGS

Depending on the situation, we can observe multiple expert trajectories each from a different initial state to a state in the shared terminal state set X_T. Our linear-programming-based method with triangle inequality can handle these situations by defining a reverse value function for each expert trajectory. For k expert trajectories, k reverse value functions $\{rv_j\}_{j=1,..,k}$ can be defined, each with a different initial state as a terminal condition. Although the constraints from (8b) to (8e) are applied to the single value function, we should duplicate the constraints from (8f) to (8l) to cover multiple reverse value functions. We present the algorithm and some further details in Appendix C.

2) IMPLEMENTATION DETAILS

The optimization problem (8) is solved in the same manner as solving the baseline problem (4); the latter was solved based on polynomial optimization and linear matrix inequalities [3]. The inequalities (8f) and (8k) were reduced to linear matrix inequalities because each of them is an inequality over a compact basic semi-algebraic set, as is inequality (4b) in the baseline method. For the optimization, we used the YALMIP toolbox in MATLAB [20], which is suitable for handling the polynomial constraints included in (8). We present further details in Appendix D.

IV. EXPERIMENTAL RESULTS

A. EXPERIMENTAL SETTINGS

We compared our IOC method with the baseline IOC method by using three types of simple OCP tasks. In particular, the third is in an imitation learning setting when the expert is taking time-dependent behaviors.

1) HYPERPARAMETERS

The polynomials to approximate l, v, and rv were set to the same degree, assuming that they were of comparable complexities. Parameter tuning was performed for the regularization parameter λ for each IOC method before each experiment was performed. More details are described in Appendix E.

2) TASKS

The following three OCPs were solved by the steepest descent method [1]. Although the domains of state and control space in the three OCPs were normalized as $[-1, 1]$ on each coordinate to stabilize the numerical calculation, its enlargement is straightforward.

3) OCP 1

Time-independent control to the origin of the two-dimensional state space with the terminal time T being free, starting from a randomly (uniformly) sampled initial state.

$$l(\mathbf{x}, \mathbf{u}) = x_1^2 + x_2^2 + u_1^2 + u_2^2, \ \dot{\mathbf{x}} = \mathbf{u}, \ X_T = \{(0, 0)\}$$
$$X = \{\mathbf{x} | \ |x_i| \leq 1, i \in \{1, 2\}\},$$
$$U = \{\mathbf{u} | \ |u_i| \leq 1, i \in \{1, 2\}\}$$

4) OCP 2

Time-dependent control to chase a target that moves from $(-1, -1)$ to $(1, 1)$ along time $[0, 1]$, starting from a randomly (uniformly) sampled initial state.

$$l(\mathbf{x}, \mathbf{u}, t) = (x_1 - 2t + 1)^2 + (x_2 - 2t + 1)^2 + 2(u_1^2 + u_2^2),$$
$$\dot{\mathbf{x}} = 5\mathbf{u}, X_T = \{(1, 1)\}, X = \{\mathbf{x} | \ |x_i| \leq 1, i \in \{1, 2\}\},$$
$$U = \{\mathbf{u} | \ -1 \leq u_i \leq 1/2, i \in \{1, 2\}\}, \ 0 \leq t \leq 1$$

5) OCP 3

Time-dependent non-polynomial control with an intersecting trajectory, which requires different controls at the same state visited at different times. The cost function consists of the non-polynomial time-dependent term expressing 2 loops, $l'(\mathbf{x}, t)$, and the input penalty term, $u_1^2 + u_2^2$.

$$
\begin{aligned}
&l(\mathbf{x}, \mathbf{u}, t) := l'(\mathbf{x}, t) + u_1^2 + u_2^2 \\
&l'(\mathbf{x}, t) = \begin{cases}
(x_1 - 9t + 0.5)^2 + x_2^2 & (0 \le t < 1/9) \\
(x_1 - 0.5)^2 + (x_2 - 9t + 1)^2 & (1/9 \le t < 2/9) \\
(x_1 + 9t - 2.5)^2 + (x_2 - 1)^2 & (2/9 \le t < 3/9) \\
(x_1 + 0.5)^2 + (x_2 + 9t - 4)^2 & (3/9 \le t < 4/9) \\
(x_1 - 9t + 4.5)^2 + x_2^2 & (4/9 \le t < 5/9) \\
(x_1 - 0.5)^2 + (x_2 + 9t - 5)^2 & (5/9 \le t < 6/9) \\
(x_1 + 9t - 6.5)^2 + (x_2 + 1)^2 & (6/9 \le t < 7/9) \\
(x_1 + 0.5)^2 + (x_2 - 9t + 8)^2 & (7/9 \le t < 8/9) \\
(x_1 - 9t + 8.5)^2 + x_2^2 & (8/9 \le t \le 1)
\end{cases} \\
&\dot{\mathbf{x}} = 20\mathbf{u}, \ \mathbf{x}(t_0) = (-0.5, 0), \ X_T = \{(0.5, 0)\}, \\
&X = \{\mathbf{x} | \ |x_i| \le 1, i \in \{1, 2\}\}, \\
&U = \{\mathbf{u} | \ |u_i| \le 1, i \in \{1, 2\}\}, \ 0 \le t \le 1
\end{aligned}
$$

B. ACCURACY OF THE ESTIMATED COST FUNCTIONS

First, we examined the accuracy of the cost functions estimated by our IOC and the baseline IOC methods in OCP 1 and OCP 2. The experiments were performed in two different situations: one in which a single expert trajectory was given, and the other in which multiple expert trajectories were given. Each experiment was evaluated using various degrees of polynomials for approximating the cost, value and reverse value functions. When applying IOC methods to the optimal (expert) trajectories in OCP 1 and OCP 2, we used time-independent and time-dependent polynomials, respectively. The following error function (9) was used to evaluate the estimated cost functions:

$$
\min_{\alpha} \sqrt{\frac{\int_{t_0}^{T} \int_U \int_X (l(\mathbf{x}, \mathbf{u}, t) - \alpha \hat{l}(\mathbf{x}, \mathbf{u}, t))^2 d\mathbf{x} d\mathbf{u} dt}{\int_{t_0}^{T} \int_U \int_X l(\mathbf{x}, \mathbf{u}, t)^2 d\mathbf{x} d\mathbf{u} dt}}. \tag{9}
$$

This function represents the difference between the correct cost function l and the estimated cost function $\hat{l}$ over the entire domain and was normalized between 0 and 1. Because there was an indeterminacy in the global magnitude of the cost function, we prepared a scalar α to compensate for this. Equation (9) was used to evaluate a time-dependent cost function, whereas the integral over time was eliminated when evaluating a time-independent cost function.

Results: Fig. 2 shows the accuracies of the cost function in terms of the normalized error (9) averaged over ten trials. In each trial, the expert trajectories were started from different initial states that were randomly sampled from the state space. Fig. 2a and Fig. 2b show the accuracies with a single expert trajectory setting when our IOC and baseline IOC methods were applied to a single trajectory generated by OCP 1 and OCP 2 experts, respectively. Our IOC method exhibited smaller errors than the baseline IOC method in both time-independent and time-dependent settings, particularly when the degree of the approximation polynomials was large. The computation time of our IOC method was on average 1.88 times longer than that of the baseline method (Appendix D). Fig. 2c and Fig. 2d show the accuracies with multiple trajectory settings, each applied to three expert trajectories generated by OCP 1 and OCP 2 experts. Our IOC method performed better than the baseline method, even when multiple expert trajectories were given.

C. VALUE FUNCTION VISUALIZATION

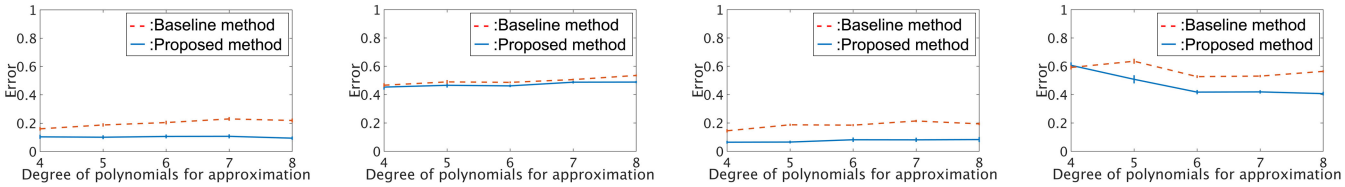
To understand the benefits of the triangle inequality, we here visualize the value functions estimated by the baseline and our IOC methods. Fig. 3 shows the value functions of OCP 1, estimated by the baseline IOC method (Fig. 3a and Fig. 3c) and our IOC method (Fig. 3b and Fig. 3d). We also examined two settings of the degree of approximation polynomials: degree of 4 (Fig. 3a and Fig. 3b) and 8 (Fig. 3c and Fig. 3d). The correct value function of OCP 1 is $v(\mathbf{x}) = x_1^2 + x_2^2$.

Overall, our method exhibited better approximations for the value functions than the baseline method. The correct value function was mirror symmetric along the diagonal lines in the two-dimensional state space and took its minimum at the origin. Although the baseline method approximated it as asymmetric for both degrees ((3a) and (3c)), our method estimated it more symmetrically ((3b) and (3d)). This can be understood that the region apart from the expert trajectory was poorly estimated by the baseline method because of the shortage of constraints, whereas our method estimated it well by utilizing the additional information from the triangle inequality.

Also, the value function estimated by the baseline method with degree 8 was worse than that with the degree 3. The estimated value function was more asymmetric so that the corner near the expert trajectory took higher value than the other corners. This can be seen as an over-adaptation; because the number of base monomials was relatively large with the polynomial degree being 8, the shortage of constraints became more severe for the distant region from the observed expert trajectory. This result suggests that our IOC method could prevent over-adaptation particularly when the number of monomial bases for the approximation is large.

D. IMITATION LEARNING OF TIME-DEPENDENT CONTROLS

Here, we applied the baseline and our IOC methods to imitate the expert trajectory for a time-dependent and non-polynomial control problem in OCP 3. In both mimickers, that is, the baseline and our IOC mimicker, we first estimated the cost function based on a single expert trajectory for OCP 3 (Fig. 4), then we solved the OCP for the estimated cost functions. Note that both IOC mimickers utilized an extended time-dependent version of the IOC methods, which are described in Appendix A. Because this is a time-dependent



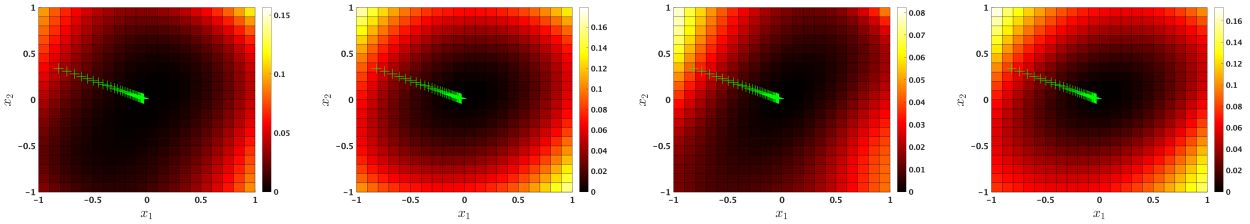
(a) Time-independent IOC on OCP 1 with a single expert trajectory

(b) Time-dependent IOC on OCP 2 with a single expert trajectory

(c) Time-independent IOC on OCP 1 with three expert trajectories

(d) Time-dependent IOC on OCP 2 with three expert trajectories

FIGURE 2. Normalized errors by the two IOC methods for various degrees of approximation polynomials. The lines and error bars denote the mean and standard error over ten runs. In Fig. 2a and Fig. 2b, IOC performed with a single expert trajectory. In Fig. 2c and Fig. 2d, IOC performed with 3 expert trajectories. Each expert trajectory started from an initial state randomly sampled from the state space (hence different from the other initial states).



(a) Baseline method degree 4

(b) Proposed method degree 4

(c) Baseline method degree 8

(d) Proposed method degree 8

FIGURE 3. Visualization of the estimated value functions. The green + marks show the expert trajectories used by the IOC methods. The correct value function is $v(\mathbf{x}) = x_1^2 + x_2^2$, which has circular contours.

control problem, the expert trajectory was intersected twice in the state space near the origin; these intersections cannot be imitated by most other traditional IOC methods which estimate a time-independent cost function. In this experiment, the degree of the approximation polynomials and regularization parameter λ were chosen to best mimic the expert data (with respect to the squared error when reproducing the expert trajectory) among the following ranges: *polynomial degree* $\in \{4, 6, 8\}$ and $\lambda \in \{10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$.

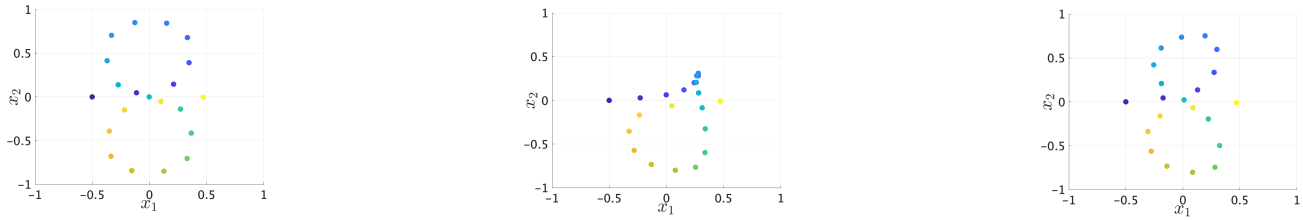
Fig. 4b and Fig. 4c show the trajectories generated by the baseline and our IOC mimickers, respectively. Although our IOC mimickers could reproduce 2 circular motions in the state space, the baseline mimicker could only imitate 1 circular motion, demonstrating the usefulness of the proposed method even in the scenario of imitation learning.

As an additional experiment, we further examined the control generalization of imitation learning by the two IOC mimickers. As in the previous experiment, the OCP behaviors with the estimated cost functions were compared with the OCP behaviors with the correct cost function, but the initial states of these OCPs were different from that of the expert trajectory in Fig. 4. This setup is to examine the generalization capability of the IOC mimickers. The experiment was performed with the hyperparameters that had been shown to be optimal in the previous experiment. Fig. 5 shows the imitation behaviors starting from a different initial state, where our IOC mimicker successfully produced a similar trajectory to the optimal trajectory whereas the baseline mimicker failed. For statistical evaluation, we repeated this imitation ten times starting from ten different initial states taken uniformly from the state space. Table 1 lists the mean squared error and its standard error over these ten trials, for the baseline and our IOC mimickers. From this table, we can see that our IOC-based imitation learning could reproduce the expert OCP's behaviors much better than the baseline method, even when initial states were different from the initial state with which the cost function had been estimated.

E. IMITATION LEARNING IN A COMPLEX DYNAMICS

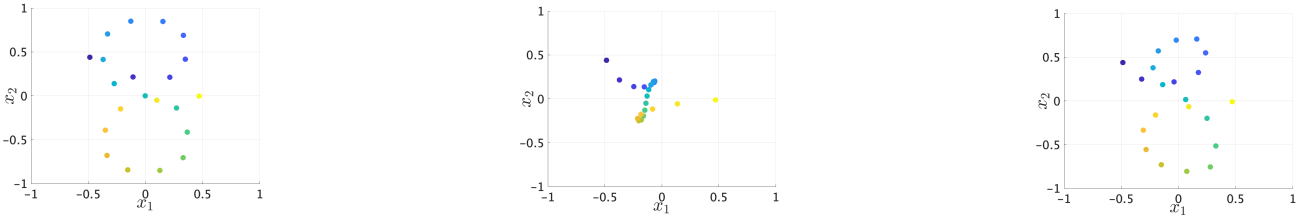
We have so far assumed linear (or polynomial) dynamics, which is a requirement for utilizing our IOC algorithm based on polynomial optimization. To show further applicability to non-polynomial dynamics, which often arises in real-world applications, here we examined an imitation learning task of 2-link manipulator control. Since the dynamics is represented as a complicated differential equation including trigonometric functions, we applied our IOC method to the approximated dynamics with the Taylor expansion up to the second order. Unlike previous experiments that used optimal control, the expert trajectory was manually crafted to perform back-and-forth control.

In this task, the state space was four-dimensional as $\mathbf{x} = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]$ and the control space was two-dimensional as $\mathbf{u} = (\tau_1, \tau_2)$. Here, θ_i and τ_i are the angle and torque of the ith joint, respectively. Each coordinate of the state and control space was rescaled to be within the interval $[-1, 1]$ to avoid possible numerical instability. The upper and lower rows in Fig. 6 visualize a series of expert and imitation behaviors, respectively. Our IOC method could well imitate back-and-forth control demonstrated by the expert, even when the



(a) Expert data (b) Baseline method (c) Proposed method

FIGURE 4. Imitation control by the baseline IOC method (Fig. 4b) and our IOC method (Fig. 4c), based on a single expert trajectory (Fig. 4) that has solved OCP 3. The squared error of the baseline IOC method was 2.72×10^{0} with the degree of approximation polynomials being 6 and $\lambda = 10^{-8}$. The squared error of our IOC method was 1.74×10^{-1} with the degree of the approximation polynomials being 6 and $\lambda = 10^{-8}$. The gradual change in color from blue (dark) to yellow (light) indicates the progress of time.



(a) Optimal trajectory (b) Baseline method (c) Proposed method

FIGURE 5. Imitation control starting at a different initial state $(-0.49, 0.44)$ from that $(-0.50, 0.00)$ of the original expert trajectory. Fig. 5a, Fig. 5b, and Fig. 5c show the OCP trajectories with the correct cost function, with the cost function estimated by the baseline IOC method (i.e., baseline IOC mimicker), and with the cost function estimated by our IOC method (i.e., our IOC mimicker), respectively.

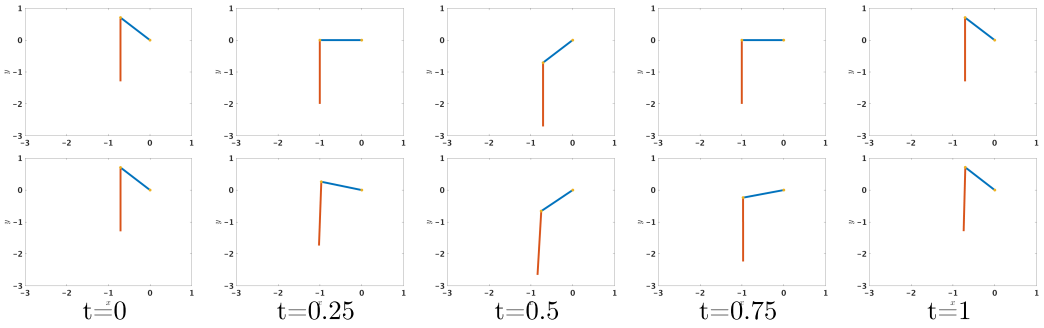


FIGURE 6. Imitation learning of control of a 2-link robot. The upper and lower figures display expert and imitation behaviors, respectively. Each column is a snapshot of the behaviors at a certain time.

TABLE 1. Mean squared errors of imitation trajectories starting from different initial states.

Baseline method	Proposed method
4.22±0.31	2.01±0.39

dynamics was non-polynomial and hence approximated by the Taylor expansion. Further discussion and details are provided in Appendix F.

V. CONCLUSION AND FUTURE WORK

In this study, we presented a new IOC method based on the newly introduced triangle inequality. Although the HJB equation is a well-established condition for optimality in control theory, it is insufficient for effectively solving the inverse problem in IOC. In our experiments, we found that the triangle inequality was effective in estimating the better representation of the value function, thus improving the accuracy of the cost function estimated by the value-function-based IOC method (Section IV-B). We also found that the improved IOC method was preferable in the imitation learning scenario (Section IV-D). Our IOC-based mimicker imitated expert control well, even starting from an initial state that was different from that of the expert trajectory. Moreover, our imitation learning method worked even in a time-dependent OCP setting. As far as we know, IOC's applicability to imitation learning of time-dependent tasks has not been studied by other research. We believe that this improved performance and enlarged applicability in the imitation of optimal controls would also enlarge the application domains of the OCP and IOC.

Because our formulation is based on the optimal control theory, which assumes that the system dynamics are deterministic and known, an extension to the situations where the

system is stochastic and/or unknown requires some additional devices. One possible direction in this regard would be to introduce spatial constraints similar to the triangle inequality to IRL. In many IRL formulations, we maximize the likelihood or posterior probability based on the gradient optimization method [21], [22]. Our IOC method with the triangle inequality could be extended in a similar way by seeking the zero point of the stochastically identified Bellman equation. However, the extension of our method to this IRL formulation has a disadvantage in that it has to dispose of the continuous action/state space and continuous time assumptions employed in optimal control theory. Moreover, in principle, conventional RL formulations based on MDPs cannot handle time-dependent cost functions. The path-integral-based RL formulation may become the foundation for extending our method to stochastic environments without losing continuous space and time-dependent cost function assumptions [23].

Although our study assumed that the expert trajectory was optimal, this may not be satisfied in many practical applications, suggesting a possible limitation of IOC when compared to IRL. However, as shown in Appendix G, our IOC method demonstrates remarkable resilience in the presence of observation noise within the expert trajectory. Therefore, our IOC method can be applied even in stochastic environments, at least to some extent, without losing the advantages of IOC.

To address the challenge of handling non-optimal trajectories in a principled manner, it is necessary to extend our triangle inequality to weakened constraints, as proposed by [24]. Additionally, the deterministic dynamics assumption may not always be available in practical applications. Nonetheless, in Section IV-E and Appendix G, we demonstrated that our method outperforms the baseline method even for complex dynamics with system-identification errors and noisy systems, respectively. Thus, we believe that our IOC method is useful not only for deterministic dynamics but also for non-deterministic ones.

In this study, we relied on the existing software for constrained linear programming [20], which restricted our tasks to being relatively low-dimensional. Applications to high-dimensional tasks can be realized using kernel methods [25] or nonlinear neural networks [26]. Because these function approximators are often optimized based on a (stochastic) gradient descent method to seek a zero-point of the Monte Carlo-based gradient function, we can utilize these function approximators by extending our constrained optimization problem to a gradient-based method.

APPENDIX A
PROPOSED METHOD FOR TIME-DEPENDENT SETTINGS

In this section, we present the triangle inequality and an associated Inverse Optimal Control (IOC) method in time-dependent settings. Here, we assume that the expert trajectory $(\mathbf{x}_0, \mathbf{u}_0, t_0)$,..,$(\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1}(= T))$ from the initial state $\mathbf{x}_0$ and time t_0 is observed, where T is the fixed terminal time and $\mathbf{x}_{n-1}$ is in the terminal state set, $\mathbf{x}_{n-1} \in X_T$. We also assume that the dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t)$ is given as a polynomial vector, and the domains of the state and control space, X and U, are given as compact basic semi-algebraic sets of the form $X = \{\mathbf{x} | g_i(\mathbf{x}) \geq 0, i = 1, \ldots, m\}$, $U = \{\mathbf{u} | k_j(\mathbf{u}) \geq 0, j = 1, \ldots, h\}$ with $g_i (i = 1, \ldots, m)$ and $k_j (j = 1, \ldots, h)$ being polynomials of $\mathbf{x}$ and $\mathbf{u}$, respectively.

The objective of a time-dependent optimal control problem is to find the control sequence $\mathbf{u}(\cdot)$ that achieves the value function $v(t_0, \mathbf{x}_0)$ from the initial state $\mathbf{x}_0$ at the initial time t_0. This time-dependent value function is defined using the time-dependent cost function $l(\mathbf{x}, \mathbf{u}, t)$ and the fixed terminal time T. Arguments $t, \mathbf{x}$ should satisfy $t_0 \leq t \leq T, \mathbf{x} \in X$. The dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t)$ is allowed to be time-dependent.

$$v(t, \mathbf{x}) = \min_{\mathbf{u}(\cdot)} \int_t^T l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad \text{s.t. } \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \mathbf{x}(t) = \mathbf{x}, \ \mathbf{x}(T) \in X_T \tag{10}$$

The following HJB equation is derived from the value function (10).

$$-\frac{\partial v}{\partial t}(\mathbf{x}, t) = \min_{\mathbf{u}} \{ l(\mathbf{x}, \mathbf{u}, t) + \frac{\partial v}{\partial \mathbf{x}}^T (\mathbf{x}, t) \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \} \tag{11}$$

This HJB equation (11) is converted into the following inequality:

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}, t) := l(\mathbf{x}, \mathbf{u}, t) + \frac{\partial v}{\partial \mathbf{x}}^T (\mathbf{x}, t) \mathbf{f}(\mathbf{x}, \mathbf{u}, t) + \frac{\partial v}{\partial t}(\mathbf{x}, t) \geq 0.$$

Given the expert trajectory $(\mathbf{x}_0, \mathbf{u}_0, t_0), \ldots, (\mathbf{x}_{n-1}, \mathbf{u}_{n-1}, t_{n-1})$, the baseline method [3], which can deal with time-dependent problems in principle, recovers the cost function using the following minimization problem:

$$\begin{aligned} &\inf_{l, v, \epsilon} \epsilon + \lambda \|l\|_1 \\ &\text{s.t. } \mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}, t) \geq 0, \forall (\mathbf{x}, \mathbf{u}, t) \in X \times U \times [t_0, T] \\ &\quad \frac{1}{n} \sum_{i=0}^{n-1} \mathcal{L}(l, v)(\mathbf{x}_i, \mathbf{u}_i, t_i) \leq \epsilon \\ &\quad v(T, \mathbf{x}) = 0, \forall \mathbf{x} \in X_T \\ &\quad \mathcal{A}(\mathcal{L}(l, v)) = 1. \end{aligned}$$

To derive the triangle inequality, the following time-dependent reverse value function rv is introduced. Arguments $t, \mathbf{x}$ must satisfy $t_0 \leq t \leq T, \mathbf{x} \in X$.

$$rv(t, \mathbf{x}) := \min_{\mathbf{u}(\cdot)} \int_t^{t_0} -l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad \text{s.t. } \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \mathbf{x}(t) = \mathbf{x}, \mathbf{x}(t_0) = \mathbf{x}_0 \tag{12}$$

Using this reverse value function, the triangle inequality can be derived as follows (constraints $\mathbf{x}(t_0) = \mathbf{x}_0, \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t)$ are omitted from parentheses for simplicity):

$$v(t_0, \mathbf{x}_0) = \min_{\mathbf{u}(\cdot)} \int_{t_0}^T l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad (\mathbf{x}(T) \in X_T) \tag{13a}$$

$$\leq \min_{\mathbf{u}(\cdot)} \int_{t_0}^{T} l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad (13b)$$
$$(\mathbf{x}(T) \in X_T, \mathbf{x}(t) = \mathbf{x})$$
$$= rv(t, \mathbf{x}) + v(t, \mathbf{x}). \quad (13c)$$

Equation (13a) represents the total cost of the optimal route from the initial state $\mathbf{x}_0$ and time t_0. Equation (13b) indicates that the total cost is greater than or equal to (13a) when there is an additional constraint to pass through a via-point $\mathbf{x}$ at time t. Dividing the integral into two terms separated at time t, i.e., $rv(t, \mathbf{x})$ and $v(t, \mathbf{x})$, we have the equation in (13c). The equality of this triangle inequality (13) is satisfied when pair $(t, \mathbf{x})$ is on the optimal trajectory.

Moreover, as derived in Appendix B, the HJB equation for the reverse value function is expressed as an inequality:

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}, t)$$
$$:= l(\mathbf{x}, \mathbf{u}, t) - \frac{\partial rv}{\partial \mathbf{x}}^T (\mathbf{x}, t)\mathbf{f}(\mathbf{x}, \mathbf{u}, t) - \frac{\partial rv}{\partial t}(\mathbf{x}, t) \geq 0. \quad (14)$$

Using the derived inequalities in (13) and (14), the constraints from (15f) to (15l) can be derived in the same manner as for the time-independent case. All these derived constraints yield the following constrained optimization problem:

$$\inf_{l, v, rv, \epsilon_a, \epsilon_b, \epsilon_c} \epsilon_a + \epsilon_b + \epsilon_c + \lambda \|l\|_1 \quad (15a)$$

s.t. -Constraints from the baseline method-

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}, t) \geq 0, \forall (\mathbf{x}, \mathbf{u}, t) \in X \times U \times [t_0, T] \quad (15b)$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \mathcal{L}(l, v)(\mathbf{x}_i, \mathbf{u}_i, t_i) \leq \epsilon_a \quad (15c)$$

$$v(T, \mathbf{x}) = 0, \forall \mathbf{x} \in X_T \quad (15d)$$

$$\mathcal{A}(\mathcal{L}(l, v)) = 1 \quad (15e)$$

-Constraints from the introduction of rv-

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}, t) \geq 0, \forall (\mathbf{x}, \mathbf{u}, t) \in X \times U \times [t_0, T] \quad (15f)$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \mathcal{RL}(l, rv)(\mathbf{x}_i, \mathbf{u}_i, t_i) \leq \epsilon_b \quad (15g)$$

$$rv(t_0, \mathbf{x}_0) = 0 \quad (15h)$$

$$\mathcal{A}(\mathcal{RL}(l, rv)) = 1 \quad (15i)$$

$$rv(T, \mathbf{x}_{n-1}) = v(t_0, \mathbf{x}_0) \quad (15j)$$

-Constraints from the triangle inequality-

$$v(t, \mathbf{x}) + rv(t, \mathbf{x}) \geq v(t_0, \mathbf{x}_0), \forall (t, \mathbf{x}) \in [t_0, T] \times X \quad (15k)$$

$$\frac{1}{n} \sum_{i=0}^{n-1} \{v(t_i, \mathbf{x}_i) + rv(t_i, \mathbf{x}_i) - v(t_0, \mathbf{x}_0)\} \leq \epsilon_c. \quad (15l)$$

APPENDIX B
HAMILTON-JACOBI-BELLMAN (HJB) EQUATION FOR THE REVERSE VALUE FUNCTION

In this section, we derive the HJB equation for the reverse value function. Considering a small change in time, $\Delta t (> 0)$, the reverse value function (12) can be transformed as follows:

$$rv(t, \mathbf{x}) := \min_{\mathbf{u}(\cdot)} \int_{t}^{t_0} -l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau$$
$$= \min_{\mathbf{u}(\cdot)} \{ \int_{t}^{t-\Delta t} -l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau$$
$$+ \int_{t-\Delta t}^{t_0} -l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau \}$$
$$= \min_{\mathbf{u}(\cdot)} \{ \int_{t}^{t-\Delta t} -l(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) d\tau$$
$$+ rv(t - \Delta t, \mathbf{x}(t - \Delta t)) \}.$$

Application of the Taylor expansion yields the following equation:

$$= \min_{\mathbf{u}} \{ l(\mathbf{x}, \mathbf{u}, t)\Delta t + rv(t, \mathbf{x}) - \frac{\partial rv}{\partial t}(t, \mathbf{x})\Delta t$$
$$- \frac{\partial rv}{\partial \mathbf{x}}^T (t, \mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}, t)\Delta t + o(\Delta t) \}.$$

Dividing by Δt and taking the limit $\Delta t \to 0$ yields the following HJB equation:

$$0 = \min_{\mathbf{u}} \{ l(\mathbf{x}, \mathbf{u}, t) - \frac{\partial rv}{\partial t}(t, \mathbf{x}) - \frac{\partial rv}{\partial \mathbf{x}}^T (t, \mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}, t) \}.$$

The corresponding inequality to this HJB equation is obtained as follows:

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}, t)$$
$$= l(\mathbf{x}, \mathbf{u}, t) - \frac{\partial rv}{\partial t}(t, \mathbf{x}) - \frac{\partial rv}{\partial \mathbf{x}}^T (t, \mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}, t) \geq 0.$$

In the time-independent cases, this inequality becomes as simple as:

$$\mathcal{RL}(l, rv)(\mathbf{x}, \mathbf{u}) = l(\mathbf{x}, \mathbf{u}) - \frac{\partial rv}{\partial \mathbf{x}}^T (\mathbf{x})\mathbf{f}(\mathbf{x}, \mathbf{u}) \geq 0.$$

APPENDIX C
PROPOSED METHOD FOR MULTIPLE TRAJECTORIES SETTINGS

In this section, we describe our IOC method in multiple expert trajectories settings. There are k expert trajectories each indexed by j: $\{(\mathbf{x}_0^j, \mathbf{u}_0^j, t_0^j), \ldots, (\mathbf{x}_{n_j-1}^j, \mathbf{u}_{n_j-1}^j, t_{n_j-1}^j)\}_{j=1,\ldots,k}$, where each trajectory has n_j tuples of state, control and time stamp. The last states $\{\mathbf{x}_{n_j-1}^j\}_{j=1,\ldots,k}$ in the trajectories are assumed to be in the terminal state set X_T.

The triangle inequality is derived for each trajectory. So, we define k reverse value functions $\{rv_j\}_{j=1,..,k}$, each having a different initial state $\mathbf{x}_0^j$ in the expert trajectories as a terminal condition. We introduced a modified epsilon relaxation in (16c) to deal with the k trajectories. Constraints (16f) to (16l) are defined for each trajectory because each trajectory utilizes its specific reverse value function.

$$\inf_{l, v, \epsilon_a, \{rv_j, \epsilon_b^j, \epsilon_c^j\}_{j=1,\ldots,k}} \epsilon_a + \sum_{j=1}^{k} \{\epsilon_b^j + \epsilon_c^j\} + \lambda \|l\|_1 \quad (16a)$$

s.t. -Constraints from the baseline method-

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}) \geq 0, \forall(\mathbf{x}, \mathbf{u}) \in X \times U \quad (16b)$$

$$\frac{1}{\sum_{j=1}^{k} n_j} \sum_{j=1}^{k} \sum_{i=0}^{n_j-1} \mathcal{L}(l, v)\left(\mathbf{x}_i^j, \mathbf{u}_i^j\right) \leq \epsilon_a \quad (16c)$$

$$v(\mathbf{x}) = 0, \forall \mathbf{x} \in X_T \quad (16d)$$

$$\mathcal{A}(\mathcal{L}(l, v)) = 1 \quad (16e)$$

-Constraints from the introduction of $\{rv_j\}_{j=1,\ldots,k}$-

$$\mathcal{RL}(l, \mathrm{rv}_j)(\mathbf{x}, \mathbf{u}) \geq 0, \quad \forall(\mathbf{x}, \mathbf{u}) \in \mathrm{X} \times \mathrm{U}\ (j = 1, \ldots, k) \quad (16f)$$

$$\frac{1}{n_j} \sum_{i=0}^{n_j-1} \mathcal{RL}(l, rv_j)\left(\mathbf{x}_i^j, \mathbf{u}_i^j\right) \leq \epsilon_b^j\ (j = 1, \ldots, k) \quad (16g)$$

$$rv_j(\mathbf{x}_0^j) = 0\ (j = 1, \ldots, k) \quad (16h)$$

$$\mathcal{A}(\mathcal{RL}(l, rv_j)) = 1\ (j = 1, \ldots, k) \quad (16i)$$

$$rv_j(\mathbf{x}_{n_j-1}^j) = v(\mathbf{x}_0^j)\ (j = 1, \ldots, k) \quad (16j)$$

-Constraints from the triangle inequality-

$$v(\mathbf{x}) + rv_j(\mathbf{x}) \geq v(\mathbf{x}_0^j), \forall \mathbf{x} \in X\ (j = 1, \ldots, k) \quad (16k)$$

$$\frac{1}{n_j} \sum_{i=0}^{n_j-1} \{v(\mathbf{x}_i^j) + \mathrm{rv}_j(\mathbf{x}_i^j) - v(\mathbf{x}_0^j)\} \leq \epsilon_c^j \quad (j = 1, \ldots, k) \quad (16l)$$

In time-dependent cases, k expert trajectories $\{(\mathbf{x}_0^j, \mathbf{u}_0^j, t_0^j), \ldots, (\mathbf{x}_{n_j-1}^j, \mathbf{u}_{n_j-1}^j, t_{n_j-1}^j (= T))\}_{j=1,\ldots,k}$ share the common terminal time T, and each last state $\{\mathbf{x}_{n_j-1}^j\}_{j=1,\ldots,k}$ is in the terminal state set X_T. Initial time t_0^j for each trajectory can vary with the assumption $t_0^j \in [t_0, T]$ for some scalar t_0. Using (12), the k time-dependent reverse value functions $\{rv_j\}_{j=1,\ldots,k}$ are defined as having initial time and state $t_0^j, \mathbf{x}_0^j$ as the terminal condition. Using the time-dependent constraints derived in Appendix A, the following program is formulated:

$$\inf_{l, v, \epsilon_a, \{rv_j, \epsilon_b^j, \epsilon_c^j\}_{j=1,\ldots,k}} \epsilon_a + \sum_{j=1}^{k} \{\epsilon_b^j + \epsilon_c^j\} + \lambda \|l\|_1$$

s.t.-Constraints from the baseline method-

$$\mathcal{L}(l, v)(\mathbf{x}, \mathbf{u}, t) \geq 0, \forall(\mathbf{x}, \mathbf{u}, t) \in X \times U \times [t_0, T]$$

$$\frac{1}{\sum_{j=1}^{k} n_j} \sum_{j=1}^{k} \sum_{i=0}^{n_j-1} \mathcal{L}(l, v)\left(\mathbf{x}_i^j, \mathbf{u}_i^j, t_i^j\right) \leq \epsilon_a$$

$$v(T, \mathbf{x}) = 0, \forall \mathbf{x} \in X_T$$

$$\mathcal{A}(\mathcal{L}(l, v)) = 1$$

-Constraints from the introduction of $\{rv_j\}_{j=1,\ldots,k}$-

$$\mathcal{RL}(l, \mathrm{rv}_j)(\mathbf{x}, \mathbf{u}, t) \geq 0, \quad \forall(\mathbf{x}, \mathbf{u}, t) \in \mathrm{X} \times \mathrm{U} \times [t_0^j, T](j = 1, .., k)$$

$$\frac{1}{n_j} \sum_{i=0}^{n_j-1} \mathcal{RL}(l, rv_j)\left(\mathbf{x}_i^j, \mathbf{u}_i^j, t_i^j\right) \leq \epsilon_b^j\ (j = 1, \ldots, k)$$

$$rv_j(t_0^j, \mathbf{x}_0^j) = 0 \quad (j = 1, \ldots, k)$$

$$\mathcal{A}(\mathcal{RL}(l, rv_j)) = 1 \quad (j = 1, \ldots, k)$$

$$rv_j(T, \mathbf{x}_{n_j-1}^j) = v(t_0^j, \mathbf{x}_0^j) \quad (j = 1, \ldots, k)$$

-Constraints from the triangle inequality-

$$v(t, \mathbf{x}) + \mathrm{rv}_j(t, \mathbf{x}) \geq v(t_0^j, \mathbf{x}_0^j), \quad \forall(t, \mathbf{x}) \in [t_0^j, T] \times \mathrm{X}(j = 1, .., k)$$

$$\frac{1}{n_j} \sum_{i=0}^{n_j-1} \{v(t_i^j, \mathbf{x}_i^j) + rv_j(t_i^j, \mathbf{x}_i^j) - v(t_0^j, \mathbf{x}_0^j)\} \leq \epsilon_c^j \quad (j = 1, .., k).$$

APPENDIX D
COMPUTATIONAL RESOURCES

All the experiments were performed using a MacBook Pro with 16 GB memory and an Apple M1 chip. The MATLAB version is *R2020b*. As we described in Section III of the main text, we followed the original method [3] to let the inequality hold over the whole domain. In this method, the inequality constraints are transformed into the class of SOS (sum of squares) polynomials, which are subsequently converted into linear matrix inequalities. This transformation allows us to efficiently solve the optimization problem using a semi-definite programming solver, enabling an efficient algorithm execution. Semi-definite programming, employed in both the baseline and our IOC methods, is one of the convex optimization problems. Due to the convexity of the IOC problems, the optimized parameters are guaranteed to be globally optimal, except for numerical inaccuracies. Because of this, the current comparison in terms of the optimized cost function directly reflects the best performance of each IOC method, indicating their respective effectiveness.

Since our new method employs a similar technique to implement the triangle inequality and the inequality from the HJB equation, the number of constraints has increased further. However, the computational time did not increase significantly, at most doubling in a single trajectory setting.

We experimentally evaluate the computational resources in this section. Table 2 lists the mean computational time for estimating the cost function in OCP 1 and its standard error over 10 trials. Each row represents the degree of approximation polynomials. As shown in the degree 8 row in Table 2, the computational time required for our IOC method was 1.88 times longer than that of the baseline method.

In the multiple expert trajectories setting, the computational time for the baseline and our IOC methods increases with the number of trajectories. The computational time for these IOC methods was compared using OCP 1 with the approximation polynomial degree being 4 throughout this experiment. Table 3 lists the mean computational time and its standard error over 10 trials. Each row represents the

TABLE 2. Computational time (seconds) in the single expert trajectory setting with various polynomial degrees.

Degree	Baseline method	Proposed method
4	1.141±0.105	1.821±0.131
5	1.301±0.029	2.152±0.034
6	1.946±0.029	3.302±0.027
7	2.609±0.018	4.603±0.039
8	5.294±0.034	9.935±0.115

TABLE 3. Computational time (seconds) in the single expert trajectory setting with various numbers of expert trajectories.

#trajectories	Baseline method	Proposed method
2	1.517±0.092	3.332±0.512
4	1.677±0.051	4.537±0.149
6	1.858±0.038	6.360±0.121
8	2.027±0.033	8.131±0.134
10	2.217±0.036	10.417±0.176

TABLE 4. List of parameter λ used in each setting.

OCP setting	(2a)	(2b)	(2c)	(2d)
Baseline method	10^{-2}	10^{-2}	10^{-10}	10^{-10}
Proposed method	10^{-6}	10^{-6}	10^{-8}	10^{-6}

number of expert trajectories. Although the computational time for both methods increased almost linearly with the number of expert trajectories, our method showed a larger rate of increase in the computational time.

We consider this larger rate of increase in our IOC method was caused by the additional constraints implemented for each trajectory. Since the number of constraints for HJB equations and triangle inequalities increased linearly with the number of expert trajectories, the computational time increased almost linearly as well.

APPENDIX E
HYPERPATAMETERS FOR REPRODUCIBILITY

This section describes the parameter-tuning process and regularization parameter λ used in OCP 1 and OCP 2. Before each experiment in the main text, both IOC methods were performed with the following range of $\lambda \in \{10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$. The polynomial degree 6, which is the center in the polynomial interval, was used in this process. Approximation errors calculated by the error function (9) were averaged over 5 trials, and the λ producing the least mean error was chosen. Table 4 lists the hyperparameters chosen for each setting. (2a)-(2d) in Table 4 represent the figures in the main text, which corresponds to the following settings: OCP 1 with a single expert trajectory, OCP 2 with a single expert trajectory, OCP 1 with 3 expert trajectories, OCP 2 with 3 expert trajectories, respectively. Since the hyperparameters of both IOC methods are meticulously tuned in every experimental settings, we can evaluate and compare the best performance of the IOC methods, ensuring a fair comparison of the problem formulations between the IOC methods.

TABLE 5. Parameters of a 2-link manipulator.

m_1(Kg)	m_2(Kg)	I_1(Kg/m^2)	I_2(Kg/m^2)
1	1	1/12	1/3

l_{a1}(m)	l_{a2}(m)	l_1(m)	l_2(m)	g(m/s^2)
1/2	1	1	2	0

APPENDIX F
EXPERIMENTAL DETAILS OF A 2-LINK MANIPULATOR IMITATION

A. EXPERIMENTAL DETAILS

This section describes the details of the experiment presented in Section IV-E. Equation (18) represents the dynamics of a 2-link manipulator [27] used in this study.

$$
\begin{aligned}
&\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta_1} \\ \ddot{\theta_2} \end{bmatrix} + \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \\
&M_{11} = m_1 l_{a1}^2 + m_2 \left(l_1^2 + l_{a2}^2 + 2 l_1 l_{a2} \cos\theta_2 \right) + I_1 + I_2 \\
&M_{12} = M_{21} = m_2 \left(l_{a2}^2 + l_1 l_{a2} \cos\theta_2 \right) + I_2 \\
&M_{22} = m_2 l_{a2}^2 + I_2 \\
&H_1 = -m_2 l_1 l_{a2} \sin\theta_2 \dot{\theta}_2^2 - 2 m_2 l_1 l_{a2} \sin\theta_2 \dot{\theta}_2 \dot{\theta}_1 \\
&H_2 = m_2 l_1 l_{a2} \sin\theta_2 \dot{\theta}_1^2 \\
&G_1 = (m_1 l_{a1} + m_2 l_1)\, g \cos\theta_1 + m_2 l_{a2} g \cos(\theta_1 + \theta_2) \\
&G_2 = m_2 l_{a2} g \cos(\theta_1 + \theta_2)
\end{aligned} \tag{18}
$$

The state space was four-dimensional as $\mathbf{x} = (\theta_1, \theta_2, \dot{\theta_1}, \dot{\theta_2})$, where θ_1 and θ_2 are angles of the two joints of the manipulator. The control space was two-dimensional as $\mathbf{u} = (\tau_1, \tau_2)$, i.e., the torques applied to the two joints. The parameters of the manipulator are shown in Table 5.

The non-polynomial dynamics was approximated by applying the Taylor expansion around $[\theta_1, \theta_2, \dot{\theta_1}, \dot{\theta_2}, \tau_1, \tau_2] = [-\pi/2, \pi/2, 0, 0, 0, 0]$ up to the second order.

Unlike OCP 1 and OCP 2, which used quadratic cost functions, expert trajectory was given by the following equations:

$$
\theta_1(t) = \begin{cases} \text{if } 0 \le t < 1/2 \\ \quad 96\pi t^5 - 120\pi t^4 + 40\pi t^3 - 3\pi/4 \\ \text{if } 1/2 \le t \le 1 \\ \quad -96\pi (t-0.5)^5 + 120\pi (t-0.5)^4 \\ \qquad -40\pi (t-0.5)^3 - \pi/4 \end{cases}
$$

TABLE 6. Squared errors of imitation trajectories.

λ	10^{-8}	10^{-6}	10^{-4}	10^{-2}
Baseline method	409.7	455.5	507.9	431.6
Proposed method	377.1	33.3	32.3	471.2

$$\theta_2(t) = \begin{cases} \text{if } 0 \le t < 1/2 \\ \quad -96\pi t^5 + 120\pi t^4 - 40\pi t^3 + 3\pi/4 \\ \text{if } 1/2 \le t \le 1 \\ \quad 96\pi(t-0.5)^5 - 120\pi(t-0.5)^4 \\ \qquad +40\pi(t-0.5)^3 + \pi/4. \end{cases} \tag{19}$$

Equation (19) was designed to perform back-and-forth control between the two angles, $(\theta_1, \theta_2) = (-3\pi/4, 3\pi/4)$ and $(\theta_1, \theta_2) = (-\pi/4, \pi/4)$. The velocity and acceleration at each angle were 0. Obviously, the cost function associated with the expert trajectory given by the function (19) is time-dependent.

The imitation behaviors were produced by solving the OCP with the following settings: 1) the cost function estimated by IOC with the approximated dynamics, 2) the initial and terminal states set to $(\theta_1, \theta_2, \dot{\theta_1}, \dot{\theta_2}) = (-3\pi/4, 3\pi/4, 0, 0)$, and 3) the correct dynamics (18).

B. ADDITIONAL EXPERIMENT

As shown in Section IV-E, our method based on the triangle inequality successfully imitated the expert behaviors. As an additional experiment, we compared the baseline and our IOC methods in terms of imitation performance with various settings of the hyper-parameter λ. During this experiment, the degree of approximation polynomials was consistently set at 4 to keep the computation time feasible.

Table 6 presents the squared error between the expert's and mimicker's control trajectories. Our method could imitate the expert's behaviors most accurately with $\lambda = 10^{-4}$. Fig. 6 in the main text shows the result in this setting.

APPENDIX G
SENSITIVITY TO NOISES

Most IOC methods, including the baseline [3] and our method, assume that the demonstrated trajectories are optimal, i.e., the solutions of OCP. Here, we examined how well our IOC method works, when this optimality is not fully satisfied; in particular, we applied our IOC method to a situation in which there were expert trajectories that had been disturbed by observation noise. As an experimental setup, we re-used the setup for OCP1 with 3 expert trajectories setting (Fig. 2c in the main text), but we applied two modifications: 1) the random noise from the uniform distribution of $[-0.05.0.05]$ was added to each state and control signal along the expert trajectories, and 2) the initial states were sampled from the uniform distribution of $[-0.95, 0.95]$ for each coordinate instead of $[-1, 1]$ to ensure the noisy trajectories stayed in the domain X. Fig. 7 shows the accuracy of the estimated cost functions over ten trials with this setting. Our IOC method estimated the cost function more accurately than the baseline method even when the expert trajectories included observation noise (i.e. the observed dynamics is stochastic). This result suggested that the incorporation of our triangle inequality has effectively mitigated the inherent uncertainties introduced by observation noise.

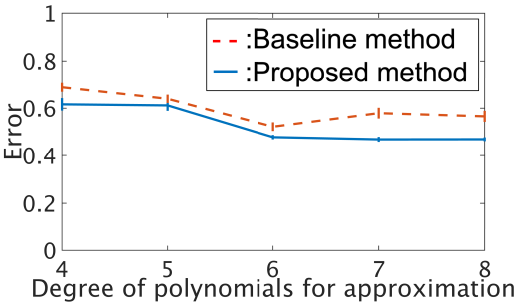
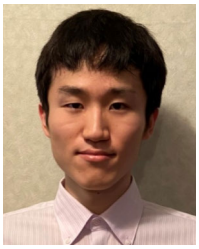


FIGURE 7. Normalized errors by the two IOC methods for various degrees of approximation polynomials. The lines and error bars denote the mean and standard error over ten trials. According to the procedure in Appendix E, λ was tuned to 10^{-8} and 10^{-6} in the baseline and our methods, respectively.

REFERENCES

[1] D. E. Kirk, *Optimal Control Theory: An Introduction*. Chelmsford, MA, USA: Courier Corporation, 2004.
[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018, pp. 69–94. [Online]. Available: http://incompleteideas.net/book/RLbook2020.pdf
[3] E. Pauwels, D. Henrion, and J.-B. Lasserre, "Inverse optimal control with polynomial optimization," in *Proc. 53rd IEEE Conf. Decis. Control*, Dec. 2014, pp. 5581–5586.
[4] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 1, 2000, p. 2.
[5] E. Adida and G. Perakis, "A nonlinear continuous time optimal control model of dynamic pricing and inventory control with no backorders," *Nav. Res. Logistics*, vol. 54, no. 7, pp. 767–795, Oct. 2007.
[6] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–35, Mar. 2018.
[7] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2004, p. 1.
[8] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2011, pp. 182–189.
[9] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Müller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*.
[10] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1, 1988, pp. 1–9.
[11] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–15.
[12] H. Wang, B. Liu, X. Ping, and Q. An, "Path tracking control for autonomous vehicles based on an improved MPC," *IEEE Access*, vol. 7, pp. 161064–161073, 2019.
[13] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007.
[14] F. Oldewurtel, A. Parisio, C. N. Jones, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and M. Morari, "Use of model predictive control and weather forecasts for energy efficient building climate control," *Energy Buildings*, vol. 45, pp. 15–27, Feb. 2012.
[15] Q. Zou, H. Li, and R. Zhang, "Inverse reinforcement learning via neural network in driver behavior modeling," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1245–1250.

[16] K. Dvijotham and E. Todorov, "Inverse optimal control with linearly-solvable MDPs," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 335–342.
[17] R. Kamalapurkar, "Linear inverse reinforcement learning in continuous time and space," in *Proc. Annu. Amer. Control Conf. (ACC)*, Jun. 2018, pp. 1683–1688.
[18] R. Self, M. Harlan, and R. Kamalapurkar, "Online inverse reinforcement learning for nonlinear systems," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Aug. 2019, pp. 296–301.
[19] E. Pauwels, D. Henrion, and J.-B. Lasserre, "Linear conic optimization for inverse optimal control," *SIAM J. Control Optim.*, vol. 54, no. 3, pp. 1798–1825, Jan. 2016.
[20] J. Lofberg, "Pre- and post-processing sum-of-squares programs in practice," *IEEE Trans. Autom. Control*, vol. 54, no. 5, pp. 1007–1011, May 2009.
[21] J. Choi and K.-E. Kim, "Map inference for Bayesian inverse reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 24, 2011, pp. 1–9.
[22] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 8, 2008, pp. 1433–1438.
[23] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *J. Mach. Learn. Res.*, vol. 11, pp. 3137–3181, Nov. 2010.
[24] S. Tschiatschek, A. Ghosh, L. Haug, R. Devidze, and A. Singla, "Learner-aware teaching: Inverse reinforcement learning with preferences and constraints," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.
[25] S. Levine, Z. Popovic, and V. Koltun, "Nonlinear inverse reinforcement learning with Gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 24, 2011, pp. 1–9.
[26] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," 2015, *arXiv:1507.04888*.
[27] A. Kumar, S. Kasera, and L. B. Prasad, "Optimal control of 2-link under-actuated robot manipulator," in *Proc. Int. Conf. Innov. Inf., Embedded Commun. Syst. (ICIIECS)*, Mar. 2017, pp. 1–6.



SHO MITSUHASHI received the B.E. degree in electrical engineering and bioscience from Waseda University, Tokyo, Japan, in 2021, and the master's degree in informatics from Kyoto University, Kyoto, Japan, in 2023, where he is currently pursuing the Ph.D. degree in informatics. His research interests include imitation learning, inverse reinforcement learning, and reinforcement learning.

SHIN ISHII received the B.E., M.E., and Ph.D. degrees from The University of Tokyo, in 1986, 1988, and 1997, respectively. He was an Associate Professor with the Nara Institute of Science and Technology, Nara, Japan, in 1997, and a Professor, in 2001. He has been a Full Professor with the Graduate School of Informatics, Kyoto University, Kyoto, Japan, since 2007. He has been the Director of ATR Neural Information Analysis Laboratories, Kyoto, since 2018. His research interests include machine learning, computational neuroscience, and intelligent robotics.