Inverse Optimal Control from Demonstration Segments

Wanxin Jin*, Zihao Liang*, Shaoshuai Mou*

Abstract—This paper develops an inverse optimal control method to learn an objective function from segments of demonstrations. Here, each segment is part of an optimal trajectory within any time interval of the horizon. The unknown objective function is parameterized as a weighted sum of given features with unknown weights. The proposed method shows that each trajectory segment can be transformed into a linear constraint to the unknown weights, and then all available segments are incrementally incorporated to solve for the unknown weights. Effectiveness of the proposed method is shown on a simulated 2-link robot arm and a 6-DoF maneuvering quadrotor system, in each of which only segment data of the systems’ trajectories are available.

I. INTRODUCTION

With the capability of recovering an objective function of an optimal control system from observations of the system’s trajectories, inverse optimal control (IOC) has been widely applied in imitation learning [1], where a learner mimics an expert by learning the expert’s underlying objective function, autonomous vehicles [2], where human driver’s driving preference is learned and transferred to vehicle controllers, and human-robot interactions [3], where an objective function of human motor control is inferred to enable efficient prediction and coordination.

Existing IOC methods usually assume the unknown objective function could be parameterized as a linear combination of selected features (or basis functions) [4], [5]. Here, each feature characterizes one aspect of the performance of the system operation, such as energy cost, time consumption, risk levels, etc. Then, the goal of IOC becomes estimating the unknown weights for those features [6]. The authors of [7]–[11] have adopted a double-layer architecture, where the estimate of the weights is updated in an outer layer while the corresponding optimal trajectory is generated by solving the optimal control problem in an inner layer. Techniques based on the double-layer framework usually suffer high computational cost since optimal control problems need to be solved repeatedly [12]. Recent IOC techniques have been developed by leveraging optimality conditions, which the observed optimal trajectory must satisfy, and thus the unknown weights can be directly obtained by solving the established optimality equations. Related work along this direction includes [13]–[15], where Karush-Kuhn-Tucker conditions are used, [16], where Pontryagin’s maximum principle [17] are used.

Despite significant progress achieved as described above, most existing IOC methods cannot learn the objective function unless a complete system trajectory within an entire time horizon is observed. Such requirement of observations has limited their capabilities in the cases where only incomplete trajectory data, or even sparse data, is available, for example, due to limited sensing capability, sensor failures, or occlusion [18], [19]. In [19], given sparse corrections (demonstrations), the authors create an intended trajectory of full horizon based on the sparse data using trajectory shaping/interpolation [20], in order to utilize the maximum margin IOC approach [7]. Although successful in learning from human corrections, it is likely that the artificially-created trajectory might not exactly reflect the actual trajectory of a human expert. In [21], the authors model the missing data using a probability distribution, then both the objective function and the missing part are learned under the maximization-expectation framework. Besides huge computational cost, this work, however, has not provided how percentage of missing information affects learning performance. In the recent work [6], a notion of the recovery matrix has been introduced to solve IOC using incomplete observations, but it still requires the observation data to be consecutive and long enough to satisfy the recovery condition.

In recognition of the above limitations, this paper aims to develop an approach to learn the objective function directly from available demonstration segments, without the attempt to characterize missing information. By saying demonstration segments, we refer to a collection of segments of the system’s trajectory of states and inputs in any time intervals of the horizon; we allow a segment to be a single data point, i.e., a state/input at a single time instant. Each segment may be not sufficient to determine the objective function by itself, an incremental approach will be developed to incorporate all available segments to achieve an estimate of the unknown weights of the objective function.

Notations

The column operator \( \text{col} \{ x_1, x_2, ..., x_k \} \) stacks its (vector) arguments into a column. \( x_{k_1:k_2} \) denotes a stack of multiple \( x \) from \( k_1 \) to \( k_2 \) (\( k_1 \leq k_2 \)), that is, \( x_{k_1:k_2} = \text{col} \{ x_{k_1}, ..., x_{k_2} \} \). \( A \) (bold-type) denotes a block matrix. Given a vector function \( f(x) \) and a constant \( x^* \), \( \frac{df}{dx} \) denotes the Jacobian matrix with respect to \( x \) evaluated at \( x^* \). Zero matrix/vector is denoted as \( 0 \), and identity matrix as \( I \), both with appropriate dimensions. \( A' \) denotes the transpose of matrix \( A \).
II. PROBLEM STATEMENT

Consider an optimal control system with discrete-time dynamics and initial condition as follows:

\[ x_{t+1} = f(x_t, u_{t+1}), \quad x_0 \in \mathbb{R}^n, \]

where vector function \( f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \) is differentiable; \( x_t \in \mathbb{R}^n \) denotes the system state; \( u_t \in \mathbb{R}^m \) is the control input; and \( t = 0, 1, \ldots \) is the time step. Let

\[ \xi = \{ \xi_t : t = 1, 2, \ldots, T \} \quad \text{with} \quad \xi_t = \{ x_t, u_t^* \} \]

denote a trajectory of system states and inputs in a time horizon \( T \). Consider that the system trajectory \( \xi \) is a result of optimizing the following objective function:

\[ J(x_{1:T}, u_{1:T}) = \sum_{t=1}^{T} \omega f(x_t, u_t). \]

Here, \( \omega : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \) is a vector of specified features (or basis functions), with each feature \( \phi_i \) differentiable; \( \omega \in \mathbb{R}^n \) is the unknown weight vector with the \( i \)th element \( \omega_i \) being the weight for feature \( \phi_i \), \( 1 \leq i \leq r \).

In inverse optimal control, the goal is to learn the unknown weights \( \omega \) for the given features \( \phi \) from the full trajectory \( \xi \). Note that scaling \( \omega \) by an non-zero constant does not affect the IOC problem because a scaled \( \omega \) will result in the same trajectory \( \xi \). Without losing any generality, one can always scale \( \omega \) such that its first entry is equal to 1, as adopted in [13], namely,

\[ \epsilon_0' \omega = 1 \quad \text{with} \quad \epsilon_0' = [1, 0, \ldots, 0]' \in \mathbb{R}^r. \]

Suppose that one is accessible to a collection of trajectory segments, denoted by \( S \), which is a set of data segments of \( \xi \), and \( S \subseteq \xi \). A segment in \( S \) is defined as a sequence of system states and inputs \( \xi_{t:t} \subseteq \xi \), where \( t \) and \( t' \) denote the starting and ending time of each segment, respectively, and \( 1 \leq t \leq t' \leq T \). Thus,

\[ S \triangleq \{ \xi_{t:t} : t = 1, 2, \ldots, N, \ldots \}. \]

where \( t \) and \( t' \) are the starting and end time of the \( i \)th available segment. It is worth mentioning that we do not put any restrictions on \( S \), which means that any segment in it can be the full trajectory \( \xi \) or even a single input-state point at a time instance in terms of \( t = t' \). Different segments are also allowed to have overlaps. \( N \) here is used to denote \( N \) number of the segments are currently available, and the total number of segments can be very large. Also, in the method developed below, we do not require the knowledge \( t \), i.e., the starting time of each segment relative to the starting time of the whole system trajectory.

Since each segment in \( S \) may not be sufficient to determine \( \omega \) by itself, the problem of interest is to develop an IOC algorithm to achieve the estimate of \( \omega \) by incrementally incorporating all segments in \( S \).

In [14], at time \( k \), we denote the control input as \( u_{k+1} \) instead of \( u_k \) due to notation simplicity of following expositions, as adopted in [22].

III. THE PROPOSED APPROACH

In this section, we first present the idea of how to establish a constraint on the feature weights from any available segment data, then develop the IOC approach.

A. Key Idea to Utilize Any Trajectory Segment in IOC

Let \( \xi_{t:t} \) be any segment of the full trajectory \( \xi \) with \( 1 \leq t \leq t' \leq T \). Since the full trajectory \( \xi \) is generated by the system \( (1) \) minimizing \( (3) \), there exist a sequence of Lagrange multipliers (or costates) \( \lambda_{t:t} \triangleq \text{col} \{ \lambda_{t:1}^*, \ldots, \lambda_{t:t}^* \} \) such that following KKT optimality conditions [23] hold for \( \xi \), that is,

\[ \frac{\partial L}{\partial x_{t:t}} = 0 \quad \text{and} \quad \frac{\partial L}{\partial u_{t:t}} = 0, \]

where

\[ L = J(x_{1:T}, u_{1:T}) + \sum_{t=1}^{T} \lambda_t^* (f(x_{t-1}, u_t) - x_t) \]

is Lagrangian of the optimization (optimal control) problem. From (3), one has the following equations for any \( 0 \leq t \leq T \):

\[ \lambda_{t:1}^* - \frac{\partial f}{\partial x_t} \lambda_{t+1:1}^* - \frac{\partial f}{\partial u_t} \omega = 0, \]

\[ \frac{\partial f}{\partial u_t} \lambda_{t:1}^* + \frac{\partial f}{\partial \phi_t} \omega = 0, \]

which can also be achieved based on Pontryagin’s maximum principle [17]. It follows that for any trajectory segment \( \xi_{t:t} \), by stacking (8)-(9) for \( t \leq t' \) one has

\[ A(\xi_{t:t}) \lambda_{t:t}^* - M(\xi_{t:t}) \lambda_{t+1:t}^* = V(\xi_{t:t}) \lambda_{t+1:t}^*, \]

\[ B(\xi_{t:t}) \lambda_{t+1:t}^* + N(\xi_{t:t}) \omega = 0, \]

with

\[ A \triangleq \begin{bmatrix} I & \frac{\partial f}{\partial x_t} \\ 0 & I \end{bmatrix}, \quad B \triangleq \begin{bmatrix} \frac{\partial f}{\partial u_t} \\ \frac{\partial f}{\partial \phi_t} \end{bmatrix}, \quad M \triangleq \begin{bmatrix} \frac{\partial f}{\partial x_{t+1}} \\ \frac{\partial f}{\partial u_{t+1}} \\ \frac{\partial f}{\partial \phi_{t+1}} \end{bmatrix}, \quad N \triangleq \begin{bmatrix} \frac{\partial f}{\partial u_{t+1}} \\ \frac{\partial f}{\partial \phi_{t+1}} \end{bmatrix}, \quad V \triangleq \text{col} \{ 0, 0 \} \]

and

\[ \lambda_{t+1:t}^* = 0. \]

Dimensions of the above matrices are \( A \in \mathbb{R}^{n(\xi_{t:t}) \times n(\xi_{t:t})}, \quad B \in \mathbb{R}^{m(\xi_{t:t}) \times n(\xi_{t:t})}, \quad M \in \mathbb{R}^{m(\xi_{t:t}) \times r}, \quad N \in \mathbb{R}^{m(\xi_{t:t}) \times r}, \quad \) and \( V \in \mathbb{R}^{n(\xi_{t:t}) \times n} \), respectively. In above (10), since \( \lambda_{T+1:t}^* \) is undefined when \( t = T \), we define \( \lambda_{T+1:t}^* \equiv 0 \). Since the matrix \( A(\xi_{t:t}) \) is non-singular, one can eliminate \( \lambda_{t+1:t}^* \) by combining (10) and (11) and obtain

\[ F(\xi_{t:t}) \omega + E(\xi_{t:t}) \lambda_{t+1:t}^* = 0. \]
Here
\[
F(\xi_{t:t}) = BA^{-1}M + N \in \mathbb{R}^{m(t+1) \times r},
\]
\[
E(\xi_{t:t}) = BA^{-1}V \in \mathbb{R}^{m(t+1) \times n}.
\]

Note that (15) establishes a relation between any data segment \( \xi_{t:t} \), the unknown weights \( \omega \), and the costate \( \lambda_{t+1}^{*} \). Note that \( \lambda_{t+1}^{*} \) is unknown and actually related to the value function of future information [12]. In order to further eliminate \( \lambda_{t+1}^{*} \) and measure the contribution of each data segment \( \xi_{t:t} \) to solving \( \omega \), we introduce the following concept of data effectiveness for IOC problems.

**Definition 1** (Effective Data for IOC). Given system (7) and an arbitrary segment \( \xi_{t:t} = \{ x_{t:t}^{*}, u_{t:t}^{*} \} \subset \xi, 1 \leq t \leq T \), we say the segment \( \xi_{t:t} \) is data effective if

\[
\text{rank } E(\xi_{t:t}) = n,
\]

where \( E(\xi_{t:t}) \in \mathbb{R}^{m(t+1) \times n} \) is as defined in (17).

It follows from Definition 1 that for any effective segment \( \xi_{t:t} \), the corresponding quantity \( E' \) is non-singular. Thus by multiplying \( E' \) to both sides of (15), one can solve

\[
\lambda_{t+1}^{*} = -(E')^{-1}E'F\omega,
\]

which together with (15) lead to

\[
R(\xi_{t:t})\omega = 0
\]

with

\[
R(\xi_{t:t}) = F - E'(E'E)^{-1}E'F.
\]

In summary, we have the following lemma.

**Lemma 1.** For any segment \( \xi_{t:t} \subseteq \xi \) that is data effective in the sense of Definition 1, \( \omega \) must satisfy (20).

Lemma 1 bridges between any data-effective segment and the unknown objective function weights; that is, any effective segment enforces a set of linear constraints to weights \( \omega \). Thus, more data-effective segments result in more constraints for recovering \( \omega \).

Note that \( E(\xi_{t:t}) \), defined in (17), is uniquely determined by \( A(\xi_{t:t}), B(\xi_{t:t}) \) and \( V(\xi_{t:t}) \), which only rely on the data in \( \xi_{t:t} \) and system dynamics \( f \). Thus, whether a data segment is effective or not is independent of choices of features \( \phi \), and only determined by the data segment itself and the system model. Furthermore, the effective data condition (15) can be fulfilled efficiently by including additional state-input points into the current data segment, as suggested in following analysis.

**Lemma 2.** For any \( 1 \leq t \leq \bar{t} < T \), one has

\[
E(\xi_{t:t}) = \begin{bmatrix}
E(\xi_{t:t}) \\
\frac{\partial f^{t}}{\partial x_{t+1}^{t}} \\
\frac{\partial f^{t}}{\partial u_{t+1}^{t}}
\end{bmatrix}.
\]

The proof of Lemma 2 will be given in Appendix. Lemma 2 implies when \( \frac{\partial f^{t}}{\partial x_{t+1}^{t}} \) is non-singular,

\[
\text{rank } E(\xi_{t:t+1}) \geq \text{rank } E(\xi_{t:t}).
\]

The rank non-decreasing property of \( E(\xi_{t:t}) \) suggests that the more data points a segment contains, the more likely it will be effective. Indeed, as we will show in later simulations, a segment \( \xi_{t:t} \) is usually data-effective when

\[
\bar{t} - t + 1 \geq \left\lceil \frac{T}{m} \right\rceil
\]

with \( \lceil \cdot \rceil \) being ceiling operator, which is a necessary condition directly suggested by the size of \( E(\xi_{t:t}) \). Specifically, if \( m = n \), even a single state/input point can be effective. Interestingly, from (22) we find that the definition of data-effectiveness is equivalent to the definition of controllability for linear systems. This means that for any controllable linear system, as long as the length of a segment satisfies (24), the segment is also data-effective.

**B. Incremental IOC from Demonstration Segments**

Based on Lemma 1 given a collection of \( N \) data segments

\[
S = \{ \xi_{t_{i}:\bar{t}_{i}} : i = 1, 2, ..., N \} \text{ in } \mathbb{R},
\]

one has

\[
R(\xi_{t_{i}:\bar{t}_{i}})\omega = 0
\]

for each segment \( \xi_{t_{i}:\bar{t}_{i}} \), if it is effective, where \( R(\xi_{t_{i}:\bar{t}_{i}}) \) is defined in (21). Then for all data-effective segments in \( S \), one has the linear equation of the weights:

\[
R(S)\omega = 0,
\]

with

\[
R(S) = \left\{ \text{col } \{ R(\xi_{t_{i}:\bar{t}_{i}}) \} : \text{rank } E(\xi_{t_{i}:\bar{t}_{i}}) = n, 1 \leq i \leq N \right\}
\]

Here \( R(S) \) is a stack of \( R(\xi_{t_{i}:\bar{t}_{i}}) \) for which the corresponding segment \( \xi_{t_{i}:\bar{t}_{i}} \) is effective. With all data-effective segments stacking into matrix \( R(S) \), the following Lemma provides a sufficient condition for successfully estimating the unknown weight vector \( \omega \).

**Lemma 3.** Given \( R(S) \) in (26b), if

\[
\text{rank } R(S) = r - 1,
\]

then any vector \( \bar{\omega} \) from the kernel of \( R(S) \) is the scaled version of weight \( \omega \), i.e., \( \bar{\omega} = c\omega \) where \( c \in \mathbb{R} \) is a scalar.

**Proof.** Condition (27) indicates that the kernel space of \( R(S) \) is one dimensional. Since (26a) is a necessary condition for the true weight vector \( \omega \), any vector \( \bar{\omega} \in \text{ker}(R(S)) \) must satisfy \( \bar{\omega} = c\omega \) where \( c \in \mathbb{R} \) is a scalar.

Lemma 3 is the sufficient condition to guarantee successfully estimation of the true weight \( \omega \). Failure to satisfy this condition indicates that \( \text{rank } R(S) < r - 1 \) and thus \( R(S) \) has a kernel with dimension larger than one, which means a vector in the kernel is not guaranteed to be a scaled version of the true weight. To cope with the rank deficiency, more data segments should be included in \( R(S) \) in order to fulfill the condition (27). It should be also noted that for a single segment, say \( \xi_{t:t} \), even though it is data effective, i.e., \( \text{rank } E(\xi_{t:t}) = n \), it does not mean that it is able to satisfy the sufficient condition in (27) and suffice for the successful
recovery, for example, it could be rank $\mathbf{R}(\xi_{t,i}) < r - 1$ even though rank $\mathbf{E}(\xi_{t,i}) = n$. Recall that $\mathbf{E}(\xi_{t,i})$ only relies on segment data and dynamics, while $\mathbf{R}(\xi_{t,i})$ additionally relies on features, thus is stricter. We will also illustrate this in the later experiment.

In implementation, since the observation noise and/or sub-optimality exist, directly computing the weights $\omega$ from (26a) thus may only lead to trivial solutions. Thus, as adopted in previous IOC methods [13]–[16], one can choose to obtain a least square estimate for the weights by solving the following equivalent optimization,

$$\hat{\omega} = \arg \min_{\omega} \frac{1}{2} \| \mathbf{R}(\mathbf{S}) \omega \|^2,$$

subject to

$$[1, 0, \cdots, 0] \omega = 1.$$  \hspace{1cm} (28a)

Here, $\| \cdot \|$ stands for the $l_2$ norm; and $\hat{\omega}$ is called a least-square estimate to the unknown weights $\omega$.

Based on the formulation in (28a)–(28b), if we consider the segments in $\mathbf{S}$ are given incrementally, i.e., one segment each time, the following lemma presents an incremental way to solve for the least square estimate $\hat{\omega}$.

**Lemma 4.** Given the $i$th segment $\xi_{t,i}$, $1 \leq i \leq N$, let

$$W_i = \begin{cases} W_{i-1} + \mathbf{R}(\xi_{t,i})' \mathbf{R}(\xi_{t,i}) & \text{if } \xi_{t,i} \text{ effective}, \\ W_{i-1} + 0 & \text{otherwise}, \end{cases}$$

with $W_0 = 0$ and $\mathbf{R}(\xi_{t,i})$ defined in (20). Then the least-square estimate $\hat{\omega}$ in (28) given previous $N$ segments is

$$\hat{\omega} = \frac{W_N^{-1} e_1}{e_1' W_N^{-1} e_1}.$$  \hspace{1cm} (29)

A proof of the above lemma will be given in Appendix. Lemma 4 shows that the least square estimate of the weights in (28) can be achieved incrementally by adding the new segment information to the matrix $W_i$. As $W_i \in \mathbb{R}^{r \times r}$ is of fixed dimension, there is no additional memory consumption as new available data is included. Given previous $N$ segments, the least square estimate of the unknown weights are solved by (30). Based on Lemma 4 we present the IOC algorithm using demonstration segments in Algorithm 1.

**Algorithm 1:** IOC from demonstration segments

**Input:** demonstration segments $\{\xi_{t,i} : i = 1, 2, \cdots, N\}$; a feature vector $\phi$.

**Initialize:** $W_0 = 0$

for $t = 1 : N$

| Obtain the $i$th segment $\xi_{t,i}$;  
| if $\xi_{t,i}$ is effective in (18) then 
| $W_i \leftarrow W_{i-1} + \mathbf{R}(\xi_{t,i})' \mathbf{R}(\xi_{t,i})$ 
| else  
| $W_i \leftarrow W_{i-1} + 0$ 
| end

Compute the least-square estimate $\hat{\omega}$ via (30).

end

IV. Numerical Experiments

In this section, we evaluate the proposed method on a simulated robot arm and a 6-DoF quadrotor UAV system.

A. Two-link robot arm

As shown in Fig. 1 we consider that a two-link robot arm moves in vertical plane with continuous dynamics given by [24, p. 209]

$$M(\theta) \ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} + g(\theta) = \tau,$$  \hspace{1cm} (31)

where $\theta = [\theta_1, \theta_2]' \in \mathbb{R}^2$ is the joint angle vector; $M(\theta) \in \mathbb{R}^{2 \times 2}$ is the inertia matrix; $C(\theta, \dot{\theta}) \in \mathbb{R}^{2 \times 2}$ is the Coriolis matrix; $g(\theta) \in \mathbb{R}^2$ is the gravity vector; and $\tau = [\tau_1, \tau_2]' \in \mathbb{R}^2$ are the torques applied to each joint. The parameters used here follows [24, p. 209]: the link mass $m_1 = m_2 = 1$kg, the link length $l_1 = l_2 = 1$m; the distance from joint to center of mass (COM) $r_1 = r_2 = 0.5$m, and the moment of inertia with respect to COM $I_1 = I_2 = 1/12$kgm$^2$. By defining the states and control inputs of the robot arm system

$$x \triangleq [\theta_1 \ \dot{\theta}_1 \ \theta_2 \ \dot{\theta}_2]' \quad \text{and} \quad u \triangleq [\tau_1 \ \tau_2]'$$  \hspace{1cm} (32)

respectively, one could write (31) in state-space representation $\dot{x} = g(x, u)$ and further approximate it by the following discrete-time form

$$x_{k+1} \approx x_k + \Delta \cdot g(x_k, u_{k+1}) \triangleq f(x_k, u_{k+1}),$$  \hspace{1cm} (33)

where $\Delta = 0.001$s is the discretization interval. The motion of the robot arm is controlled to minimize the objective function (3), which here is set as a weighted distance to the goal state $x^g = [\theta_1^g, \dot{\theta}_1^g, \theta_2^g, \dot{\theta}_2^g]' = [0, 0, 0, 0]'$ plus the control effort $\|u\|^2$. Here, the corresponding features and weights defined are as follows.

$$\phi = \begin{bmatrix} (\theta_1 - \theta_1^g)^2 \\ (\dot{\theta}_1 - \dot{\theta}_1^g)^2 \\ (\theta_2 - \theta_2^g)^2 \\ (\dot{\theta}_2 - \dot{\theta}_2^g)^2 \\ ||u||^2 \end{bmatrix}, \quad \omega = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$  \hspace{1cm} (34)

The initial condition of the robot arm is set as $x_0 = [\frac{\pi}{3}, 0, \frac{\pi}{2}, 0]'$, and time horizon is set as $T = 100$. We set the ground-truth weights as in (34), and the resulting optimal trajectory of states and inputs is plotted in Fig. 2.
recover the true weight vector. This is because, as the system trajectory in Fig. 2, the first segment in Trial 1, the second segment in Trial 2, and the second segment in Trial 3 just reach the lower bound length of segment reach the lower bound. Data-effectiveness is a precondition for a segment to be used for solving IOC problems. Although a single segment is data-effective, it may not necessarily suffice for recovering the weight. The IOC sufficient condition (27) is stricter than the data-effectiveness condition (18), because \( E(\xi_{t_i}) \) only relies on segment data and dynamics, while \( R(\xi_{t_i}) \) additionally relies on features.

### B. Quadrotor UAV

Next, we apply the proposed method to learn the objective function for a 6-DoF quadrotor UAV maneuvering system. Consider a quadrotor UAV with the following dynamics

\[
\begin{align*}
\dot{p}_I &= v_I, \\
\dot{v}_I &= m g_I + F_I, \\
\dot{q}_{B/I} &= \frac{1}{2} \Omega(\omega_I) q_{B/I}, \\
J_B \dot{\omega}_B &= M_B - \omega \times J_B \omega_B.
\end{align*}
\]

Here, the subscription \( B \) and \( I \) denote a quantity is expressed in the body frame and inertial (world) frame, respectively; \( m \) and \( J_B \in \mathbb{R}^{3 \times 3} \) are the mass and moment of inertia with respect to body frame of the UAV, respectively; \( p \in \mathbb{R}^3 \) and \( v \in \mathbb{R}^3 \) are the position and velocity vector of the UAV; \( \omega_B \in \mathbb{R}^3 \) is the angular velocity vector of the UAV; \( q_{B/I} \in \mathbb{R}^4 \) is the unit quaternion \([25]\) that describes the attitude of UAV with respect to the inertial frame; \( \Omega(\omega_B) \) is defined as:

\[
\Omega(\omega_B) = \begin{bmatrix}
0 & -\omega_z & -\omega_y \\
\omega_z & 0 & -\omega_x \\
\omega_y & \omega_x & 0
\end{bmatrix},
\]

\( M_B \in \mathbb{R}^3 \) is the torque applied to the UAV; \( F_I \in \mathbb{R}^3 \) is the force vector applied to the UAV center of mass. The total force magnitude \( f = |F_I| \in \mathbb{R} \) (along z-axis of the body frame) and torque \( M_B = [M_x, M_y, M_z]^T \) are generated by thrust from four rotating propellers \([T_1, T_2, T_3, T_4]^T\), their relationship can be expressed as:

\[
\begin{bmatrix}
f \\
M_x \\
M_y \\
M_z
\end{bmatrix} = \begin{bmatrix}
1 & 1 & 1 & 1 \\
0 & -l_w/2 & 0 & l_w/2 \\
c & -c & c & -c
\end{bmatrix} \begin{bmatrix}
T_1 \\
T_2 \\
T_3 \\
T_4
\end{bmatrix},
\]

where \( l_w \) is the wing length of the UAV and \( c \) is a fixed constant. Similar to (33), we discretize the above dynamics with discretization interval of 0.1s. The parameters in dynamics are given in Table I.

As shown in Table I, in all trials the algorithm successfully calculates the \( \hat{\omega} \) to the feature weights in (34) except feature weights in (33). This is because although the segment \( \xi_{1:4} \) in Trial 4 is data-effective (i.e., rank \( E(\xi_{1:4}) = 4 \)), however, rank \( R(\xi_{1:4}) = 3 < 5 \) and thus it does not meet the sufficient condition stated in Lemma 3 for successful estimation. To address this, we add another segment \( \xi_{10:13} \) as shown in Trial 5 in order to fulfill the sufficient condition in (27), and now rank \( R(S) = 4 \). Therefore, Trial 5 successfully estimates the true weight vector \( \omega \).

The above results show that the data-effectiveness condition (18) is mild and easy to fulfill, for example, let the length of segment reach the lower bound. Data-effectiveness is a precondition for a segment to be used for solving IOC problems. Although a single segment is data-effective, it may not necessarily suffice for recovering the weight. The IOC sufficient condition (27) is stricter than the data-effectiveness condition (18), because \( E(\xi_{t_i}) \) only relies on segment data and dynamics, while \( R(\xi_{t_i}) \) additionally relies on features.

### Table I: IOC results from data segments

| Trial No. | Intervals of segments \([t_i, t_f]\) | Estimate \( \hat{\omega} \) |
|-----------|------------------------------------|---------------------|
| Trial 1   | \([1, 2], [10, 13], [70, 73], [80, 83]\) | \(1, 2, 1, 1, 1\) |
| Trial 2   | \([10, 30], [50, 51]\) | \(1, 2, 1, 1, 1\) |
| Trial 3   | \([50, 55], [90, 92]\) | \(1, 2, 1, 1, 1\) |
| Trial 4   | \([1, 4]\) | \([7.8, 1.5, 325.1, 1.4, 1]\) |
| Trial 5   | \([1, 4], [10, 13]\) | \(1, 2, 1, 1, 1\) |

In the IOC task, we learn the weight vector \( \omega \) from the segment data of the optimal trajectory in Fig. 2. As shown in Table I, we perform five trials, and for each trial we are given a collection of segments of optimal trajectory, as indicated by the corresponding time intervals (second column). We apply Algorithm 1 to obtain the least-square estimate \( \hat{\omega} \) for each trial, and show the estimation results in the last column in Table I.
The state and input vectors of the UAV are defined as:

\[
x \triangleq \begin{bmatrix} p' & v' & q' & \omega' \end{bmatrix}^T \in \mathbb{R}^{13},
\]

\[
u \triangleq \begin{bmatrix} T_1 & T_2 & T_3 & T_4 \end{bmatrix}^T \in \mathbb{R}^4.
\]

The control objective function of the UAV includes a carefully selected attitude error term. As used in [26], we define the attitude error between UAV’s current attitude \( q \) and the goal attitude \( q^g \) as:

\[
e(q, q^g) = \frac{1}{2} \text{Tr}(I - R(q^g)R(q)),
\]

where \( R(q) \in \mathbb{R}^{3 \times 3} \) is the direction cosine matrix [25] corresponding to the quaternion \( q \). Other error terms that are included in the control objective function are simply the squared distances to their corresponding goals:

\[
e(p, p^g) = || p - p^g ||^2,
\]

\[
e(v, v^g) = || v - v^g ||^2,
\]

where \( p^g \) and \( v^g \) are the goal position and velocity states respectively.

We generate the UAV optimal trajectory by minimizing a given control objective function. The initial set is set as

\[
x_0 = [p_0, v_0, q_0, \omega_0]^T = [-8, -6, 9, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1]^T,
\]

and the goal state is set as

\[
x^g = [p^g, v^g, q^g, \omega^g]^T = [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]^T.
\]

The control objective function is written as the weighted distance to the goal state plus the control effort \( ||u||^2 \), where the features and weights are defined as follows:

\[
\phi = \begin{bmatrix}
|| p - p^g ||^2 \\
|| v - v^g ||^2 \\
\frac{1}{2} \text{Tr}(I - R^g(q^g)R(q)) \\
|| u ||^2
\end{bmatrix},
\]

with corresponding weights:

\[
\omega = [2, 1, 1, 2]^T.
\]

The time horizon is set to \( T = 50 \).

Similar to the previous experiment, we set up four trials, and for each trial we observe different segments of the optimal state trajectories, as listed in the second column in Table III. The result of feature weights estimation is shown in Table II.

As shown in Table III, in different trials, we use different segment of system trajectory to recover the true weight vector \( \omega \). All used segments are data-effective. The proposed method successfully estimates feature weights. The results demonstrate effectiveness of the proposed method for learning an objective function from sparse trajectory segments.

V. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, an inverse optimal control method is developed to learn the objective function only using demonstration segments. The available data is a collection of multiple segments of a system optimal trajectory. We first introduce the concept of data effectiveness to evaluate the contribution of any segment to IOC, and then show that each segment data can be utilized to establish a linear constraint on the unknown objective weights. Along this key idea, the proposed IOC method incrementally incorporates each segment to obtain a least-square estimate of the weights.

For future research, we will extend the proposed method to a model-free IOC method. By say model free, it means that the dynamics model of the optimal control system is not known, and thus requires additional techniques for model approximation. The motivation here is that the assumption of a known dynamics model is sometimes challenging to fulfill since obtaining such dynamical model often requires expert knowledge. Data-driven methods would be considered as one of the possible options to recover the dynamics model from given data (e.g. states and input observations). Moreover, the estimation of weights vector with noisy data would also be one of the future research directions.

REFERENCES

[1] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in International Conference on Machine Learning, ACM, 2004, p. 1.

[2] M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in IEEE International Conference on Robotics and Automation. IEEE, 2015, pp. 2641–2646.

[3] J. Mainprice, R. Hayne, and D. Berenson, “Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces,” IEEE Transactions on Robotics, vol. 32, no. 4, pp. 897–908, 2016.

[4] A. Y. Ng, S. J. Russell, et al., “Algorithms for inverse reinforcement learning,” in International Conference on Machine Learning, 2000, pp. 663–670.

[5] K. Mombaur, A. Truong, and J.-P. Laumond, “From human to humanoid locomotion—an inverse optimal control approach,” Autonomous Robots, vol. 28, no. 3, pp. 369–383, 2010.

[6] W. Jin, D. Kulić, J. F. Lin, S. Mou, and S. Hirche, “Inverse Optimal Control for Multistage Cost Functions,” IEEE Transactions on Robotics, pp. 1–1, 2019.

[7] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, “Maximum margin planning,” in International Conference on Machine Learning. ACM, 2006, pp. 729–736.

[8] ———, “Online subgradient methods for structured prediction,” Robotics Institute, p. 55, 2007.

[9] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, “Maximum Entropy Inverse Reinforcement Learning,” in AAAI, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.
where $K = D - CA^{-1}B$ is the Schur complement of the block matrix with respect to $A$, one can obtain

$$A(\xi_{t:t+1})^{-1} = \begin{bmatrix} A(\xi_{t:t})^{-1} & A(\xi_{t:t})^{-1}V(\xi_{t:t}) \\ 0 & I \end{bmatrix}.$$

(45)

Given (43) and (45) and $V(\xi_{t:t+1}) \triangleq \text{col} \{0, \frac{\partial f'}{\partial x_{t+1}}\}$, one has

$$E(\xi_{t:t+1}) = B(\xi_{t:t+1})A(\xi_{t:t+1})^{-1}V(\xi_{t:t+1})$$

$$= \begin{bmatrix} B(\xi_{t:t})A(\xi_{t:t})^{-1}V(\xi_{t:t}) \frac{\partial f'}{\partial x_{t+1}} \\
\frac{\partial f'}{\partial x_{t+1}} \end{bmatrix}.$$

(46)

which completes the proof. ■

A2. Proof of Lemma 7

Proof. Consider $R(S)$ in (26b), the optimization problem in (28a) is equivalent to

$$\min_{\omega} \omega' W_N \omega \quad \text{s.t.} \quad \omega' e_1 = 1. \quad (47)$$

If $W_N > 0$, the solution $\hat{\omega}$ to (47) is

$$\hat{\omega} = [e_1' 0] W_N^{-1} e_1$$

$$= \begin{bmatrix} W_N^{-1} e_1 \\
e_1' W_N^{-1} e_1 \end{bmatrix} \quad (48)$$

which completes the proof. ■

APPENDIX: PROOF OF Lemmas

A1. Proof of Lemma 6

According to the definitions (12)-(14), one has the following updates

$$A(\xi_{t:t+1})^{-1} = \begin{bmatrix} A(\xi_{t:t})^{-1} \cdot V(\xi_{t:t}) \\
0 & I \end{bmatrix},$$

(43)

$$B(\xi_{t:t+1})^{-1} = \begin{bmatrix} B(\xi_{t:t}) \cdot 0 \\
0 & \frac{\partial f'}{\partial x_{t+1}} \end{bmatrix}. \quad (44)$$

Based on the fact

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}BK^{-1}CA^{-1} & -A^{-1}BK^{-1} \\
-K^{-1}CA^{-1} & K^{-1} \end{bmatrix}$$

$$= \begin{bmatrix} A(\xi_{t:t})^{-1} & A(\xi_{t:t})^{-1} \cdot V(\xi_{t:t}) \\
0 & I \end{bmatrix}.$$