# Fast Inverse Reinforcement Learning with Interval Consistent Graph for Driving Behavior Prediction

**Masamichi Shimosaka†, Junichi Sato‡, Kazuhito Takenaka◇, Kentarou Hitomi◇**
Tokyo Institute of Technology†, The University of Tokyo‡, and DENSO CORPORATION◇

**Abstract**

Maximum entropy inverse reinforcement learning (MaxEnt IRL) is an effective approach for learning the underlying rewards of demonstrated human behavior, while it is intractable in high-dimensional state space due to the exponential growth of calculation cost. In recent years, a few works on approximating MaxEnt IRL in large state spaces by graphs provide successful results, however, types of state space models are quite limited. In this work, we extend them to more generic large state space models with graphs where time interval consistency of Markov decision processes are guaranteed. We validate our proposed method in the context of driving behavior prediction. Experimental results using actual driving data confirm the superiority of our algorithm in both prediction performance and computational cost over other existing IRL frameworks.

Inverse reinforcement learning (IRL), inverse optimal control, and imitation learning(Ng and Russell 2000; Abbeel and Ng 2004) are modeling frameworks for acquiring rewards (or cost) of a certain environment by using the optimal path under a possibly different environment as training data. In particular, in human behavior modeling, it is shown that human-centered rewards can be obtained with maximum entropy inverse reinforcement learning (MaxEnt IRL)(Ziebart and others 2008), which allows suboptimal training data (Huang et al. 2015; Vernaza and Bagnell 2012; Dragan and Srinivasa 2012; Walker, Gupta, and Hebert 2014). For instance, Ziebart et al. (Ziebart et al. 2008) modeled the driving behavior of expert taxi drivers and enabled driving behavior prediction based on the experts' very own experience or knowledge. MaxEnt IRL based driving behavior prediction, which balances safety, comfort, and economic performance, is very promising.

Macroscale routing prediction, such as tens of kilometers driving, was dealt with in (Ziebart et al. 2008). Microscale (fine-grained) driving behavior prediction, e.g., acceleration, deceleration, and steering for a few hundred meters, has become increasingly important for further safety in driver assistance or automated driving. Giving emphasis to this point, research has been done on developing fine-grained driving behavior prediction based on MaxEnt IRL (Levine, Popović, and Koltun 2011; Levine and Koltun 2012; Shimosaka, Kaneko, and Nishi 2014). Though these works succeeded in showing the value of MaxEnt IRL based fine-grained modeling, there are still limitations in the general driving behavior problem. (Levine, Popović, and Koltun 2011) and (Shimosaka, Kaneko, and Nishi 2014), for example, modeled two dimensional (position–velocity) state space; however, the model cannot be applied directly to higher dimensionality, e.g., adding steering to state space. This stems from the exponential growth of the computational cost of MaxEnt IRL with respect to the dimensionality of state space, so fine-grained prediction is hard to solve because its state space is likely to become high-dimensional.

Our goal is to achieve fast fine-grained driving behavior prediction. Continuous IRL (Levine and Koltun 2012), as an example of the high-dimensional MaxEnt IRL problem, is impractical in some cases due to its local optimality. In contrast, a discrete approach guarantees global optimality once proper discrete state space is given, hence it is more suitable for driving behavior modeling. In a discrete approach, the calculation cost of MaxEnt IRL is $\mathcal{O}(|S||A|)$, where $|S|$ is the number of states and $|A|$ is the number of actions (Ziebart and others 2008). That is, the key for fast prediction is suppressing the increase of $|S|$ depending on dimensions and preparing a necessary and sufficient action set, $A$, for representing driving behavior. As examples of existing discretization schemes, there are mesh grid representation (Shimosaka, Kaneko, and Nishi 2014) and random graph based representation connected with neighbors (Byravan et al. 2015). In these approaches, however, $A$ for general dynamic systems is not trivial. This is because neighbors on state space defined by Euclidean distance do not necessarily correspond to the transition area of general dynamics defined by its state equation $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)$ under the constraint of control input $\boldsymbol{u}_t$. In other words, this approach could not always hold the assumption that each edge in MDP keeps the same time interval. This problem comes to the surface when state space contains temporal representations, e.g., position and velocity, and state space quantified with conventional discretization approaches does not satisfy time interval consistency in state transition, which is an essential requirement in Markov decision processes (MDPs). Instead of using Euclidean proximity in state space, we employ new metric around "neighbors" with the basis of the state equation of dynamics $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)$. However, the calculation cost increases if a fine graph is generated with the intention of guaranteeing interval consistency with a given time interval; thereby, fast prediction is not accomplished.

In this paper, we focus on the fact that the transition distance on state space varies depending on the time interval in state transition, i.e., the sampling rate, and we propose a novel discretization framework that can control $|S|$ with the sampling rate and $|A|$ and guarantees interval consistency in state transition. To the best of our knowledge, this is the first work that achieves MaxEnt IRL for general state space representation, that is, IRL when state space contains temporal variables.

Our contributions are summarized as follows. First, we propose a novel state space discretization framework to deal with general state space representation. This guarantees interval consistency in state transition while suppressing an increase in calculation cost depending on state space dimensions. Second, we construct fast and fine-grained driving behavior prediction based on MaxEnt IRL using a graph-based state space generated with the above state space quantization. Third, we demonstrate the superiority of the proposed framework to conventional approaches through an experimental evaluation with real driving data.

## Related Works

### Fast Approximate Solver of Markov Decision Processes

"Prediction" in IRL based driving behavior modeling corresponds to optimal path search with MDPs. MDP is common technique in the robot manipulation domain, whose state space is often high-dimensional; therefore, approximate solvers of MDPs are well-studied.

There are continuous approaches that deal with continuous state space and approximate inner functions. For instance, value function approximation (Taylor and Parr 2009; Konidaris, Osentoski, and Thomas 2011) approximates a value function as a linear combination of basis functions such as RBF and solve the MDP with gradient based approaches. EM-based policy iteration (Hoffman et al. 2009) also calculates policy on the basis of expectation–maximization algorithm. These approaches have the benefit dealing with states as continuous; however, they depend on the initial value due to their gradient method, so their solutions are not stable. This does not matter when we have only to learn a policy (we just have to try several times); however, these approaches are not suitable for online applications including driving behavior prediction, which require immediate path planning to a given reward.

On another front, there are discrete approaches that suppress calculation cost by quantizing state space coarsely. Fitted value iteration(Munos and Szepesvári 2008; Ernst, Geurts, and Wehenkel 2005) or fitted Q iteration (Farahmand et al. 2009) estimate the value function or action value function (Q function) via regression algorithms with finite samples of states, actions, and rewards on continuous state space. Though they deal with continuous state space, they are essentially equal to discrete approaches because they use finite samples, which strongly affect the performance of approximation. Alternatively, hierarchical MDP (Barry, Kaelbling, and Lozano-Pérez 2011) computes hierarchical policies by representing state space hierarchically with coarse upper-level and fine lower-level states when MDPs have a specific structure. This enables global optimum solution to be obtained without searching all lower-level states; however, how to construct hierarchies for general MDPs is not trivial.

For representing driving behavior, which is of particular interest in this paper, predicted paths should satisfy two points; the given reward is globally maximized on them, and they are smooth enough to be used as control input. Even if we obtain continuous and smooth paths with locally approximate MDPs, it is difficult to correct them to be globally optimal. In discrete approaches, in comparison, global optimality is guaranteed in quantized space, but the predicted path is rough, and discretization error remains. However, there are a number of pieces of research on fast locally path smoothing (Zucker et al. 2013; Kalakrishnan et al. 2011; Ohtsuka 2004), so the discretization error can be corrected by using these existing approaches with a given reward. Therefore, discrete approaches that guarantee global optimality under the proper state space are promising for driving behavior prediction.

### MaxEnt IRL for High-dimensional State Space

In the field of behavior modeling, maximum entropy inverse reinforcement learning (MaxEnt IRL) (Ziebart and others 2008), which represents path likelihood as a stochastic model, is an effective and commonly used approach. In return for allowing suboptimal training data, MaxEnt IRL requires the computation of a partition function whose calculation cost increases exponentially depending on the number of dimensions of state space. To solve high-dimensional MaxEnt IRL efficiently, it becomes a key factor how to bring approximation techniques in MDPs into the field of MaxEnt IRL. In this section, we discuss the pros and cons of existing methods for high-dimensional MaxEnt IRL.

Linear dynamics quadratic reward IRL (Ziebart 2010; Ziebart, Dey, and Bagnell 2012) is an efficient approach only when the state equation is linear and the reward is in a quadratic form of state; however, designing features, which is basis of the reward, is strictly limited. Vernaza et al. (Vernaza and Bagnell 2012) leveraged the symmetry of the partition function, which comes from the low dimensional structure of features, to compute the partition function efficiently. This approach guarantees global optimality in continuous space; nonetheless, it also has a hard limitation for feature representation. Levine et al. (Levine and Koltun 2012) addressed continuous state and action space by testing local optimality near the initial action sequence with the Laplace approximation of likelihood. Dragan et al. (Dragan and Srinivasa 2012) reduced calculation cost by locally approximating the reward as a quadratic form. These approximate continuous approaches have the advantage of being able to use the control input of dynamics as actions; however, solutions are not stable due to their local optimality, so these approaches are not suitable for driving behavior modeling.

In contrast, discrete approaches approximate solutions by control quantization granularity or the sampling density of state space. While the accuracy of approximation depends on the granularity of state space, they guarantee global optimality in the discrete set of states and need no limitation of features; thereby, these approaches are suitable for driving behavior modeling. In the field of human or vehicle behavior modeling via computer vision, Huang et al. (Huang and Kitani 2014) and Walker et al. (Walker, Gupta, and Hebert 2014) quantized image feature space by using clustering methods. However, clustering based approaches are inadequate for driving behavior modeling because the proper radius or number of clusters is not trivial on physical state space. The approximate MaxEnt IRL proposed in (Huang et al. 2015) avoided the dimensional dependence of calculation cost by leveraging fitted Q iteration (Farahmand et al. 2009) in policy calculation and Monte Carlo sampling in the computation of feature expectation and succeeded in dealing with significant high-dimensional state space. However, its approximation accuracy heavily depends on the number of actions. A lot of (from tens to hundreds) actions are needed for fine-grained driving behavior representation; therefore, it is difficult to apply this approach to such a field. Byravan et al. (Byravan et al. 2015) controlled the number of states not depending on the number of dimensions by representing state space as a coarse graph. Coarse representation of state space with a graph is a promising approach in the field of driving behavior modeling because this can be solved within the existing MaxEnt IRL framework (Ziebart and others 2008) and the dependency of the calculation cost on states and actions is obvious. The existing graph generation algorithm proposed in (Byravan et al. 2015), which connects a graph by using the nearest-neighbor method with Euclidean distance on state space, is useful for the field of robot manipulation; however, this is not applicable to the general dynamics field including time differential variables. In robot configuration space, state transition within a certain Euclidean distance on state space is automatically guaranteed to have time interval consistency. Nonetheless, interval consistency is not guaranteed when state space contains temporal representations because neighbors based on Euclidean proximity on state space do not necessarily correspond to the transition area of general dynamics under the constraint of control inputs.

Thus, MaxEnt IRL via graph-based coarse representation of state space is a promising approach for fast and fine-grained driving behavior prediction; however, there is also a challenge in the graph generation algorithm and a new framework is needed that guarantees time interval consistency in state transition.

## Graph-based MaxEnt IRL

In this section, we explain graph-based MaxEnt IRL (Byravan et al. 2015), which is the base of our approach, and discuss a possible design for a new IRL framework that is adequate for general state space representation.

### Discrete Maximum Entropy Inverse Reinforcement Learning

For states $s \in S$ and actions $a \in A$, a state sequence $\zeta$ is defined as $\zeta = \{(s_0, a_0), (s_1, a_1), \dots\}$. With feature vector $\boldsymbol{f}(s, a)$, given a sequence $\zeta$, we denote $\boldsymbol{f}(\zeta) = \sum_{(s_t,a_t)\in\zeta} \boldsymbol{f}(s_t, a_t)$ as a feature vector for a whole sequence. The overall reward of a sequence is defined as a linear form of $\boldsymbol{f}(\zeta)$ like $\boldsymbol{\theta}^\top \boldsymbol{f}(\zeta)$, where $\boldsymbol{\theta}$ is a weight vector. In MaxEnt IRL, a likelihood of a state sequence is represented as the maximum entropy model like equation 1.

$$P(\zeta|\boldsymbol{\theta}) \propto \exp\left(\boldsymbol{\theta}^\top \boldsymbol{f}(\zeta)\right) \tag{1}$$

Given a training dataset $\mathcal{D} = \{\tilde{\zeta}_i | i = 1, ..., M\}$, the optimal weight $\boldsymbol{\theta}^*$ is calculated by minimizing the objective, which is the sum of the negative log likelihood $L(\mathcal{D}|\boldsymbol{\theta}) = -\sum_i \ln P(\tilde{\zeta}_i|\boldsymbol{\theta})$ and regularization term $\Omega(\boldsymbol{\theta})$ as follows.

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(L(\mathcal{D}|\boldsymbol{\theta}) + \Omega(\boldsymbol{\theta})\right) \tag{2}$$

Here, the gradient of $L(\mathcal{D}|\boldsymbol{\theta})$ is

$$\begin{aligned}\frac{\partial L(\mathcal{D}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \sum_{\zeta} P(\zeta|\boldsymbol{\theta}) \boldsymbol{f}(\zeta) - \mathop{\mathbb{E}}_{i}[\boldsymbol{f}(\tilde{\zeta}_i)] \\ &= \sum_{s\in S, a\in A} D(s)\pi_{\boldsymbol{\theta}}(a|s)\boldsymbol{f}(s,a) - \mathop{\mathbb{E}}_{i}[\boldsymbol{f}(\tilde{\zeta}_i)],\end{aligned} \tag{3}$$

where $D(s)$ is the expected state visitation frequencies of state $s$. In discrete MaxEnt IRL, $D(s)$ can be computed with a dynamic programming based algorithm, and its cost is $\mathcal{O}(|S||A|)$ (Ziebart and others 2008), where $|S|$ is the number of states and $|A|$ is the number of actions. $|S| \propto \mathbb{R}^d$ for $d$ dimensional state space with naive discretization scheme such as grid representation; thereby, the calculation cost increases exponentially depending of the number of dimensions.

### Discrete Representation of State Space with Graph

In a graph $\mathcal{G}$ consisting of a finite set of nodes $\mathcal{V} \subset \mathbb{R}^d$ and edges $\mathcal{E}$, we denote discrete states $s \in \mathcal{V}$ and actions $a \in \mathcal{E}$. $\mathcal{E}(s) \subset \mathcal{V}$ is a set of nodes that can be transited from state $s$. Thereby, $|S| = |\mathcal{V}|$ and $|A| = |\mathcal{E}(s)|$. To learn with discrete MaxEnt IRL, continuous training demonstrations in $\mathbb{R}^d$ are projected on the discrete graph $\mathcal{G}$ (Byravan et al. 2015).

### Challenges Associated with Coarse Graphs

Considering the discussion above, what is important for high dimensional IRL is to suppress the increase of $|S|$ for the number of dimensions $d$. As discussed in section , coarse representation of state space with a graph is a promising approach for driving behavior prediction to suppress the increase of $|S|$; however, the existing graph generation algorithm proposed in (Byravan et al. 2015) is not applicable for driving behavior prediction. Only when the units of variables in state space are equal and there are no dependencies among them can state transition be done in a uniform time interval among nodes that are within a certain definite distance on state space. That is, interval consistency of state transition is automatically guaranteed as long as "neighbors" are connected with edges. Nonetheless, including driving behavior prediction, when state space contains temporal variables such as position and velocity, time interval consistency is not guaranteed only by connecting neighbor nodes in terms of Euclidean proximity because state transition depends on control inputs.

As an alternative to connecting neighbors based on Euclidean distance, we propose an *interval consistent graph*, which guarantees time interval consistency in state transition. Under constraint of control inputs, our graph generation algorithm can balance the trade-off between the node density of the graph and the approximation performance of path discretization. It is not able to directly control $|S|$, which calculation cost depends on; however, $|S|$ can be determined by adjusting the sampling rate and the number of connected edges (which corresponds to the number of actions) as parameters.

## Generating Interval Consistent Graphs

In this section, we introduce an interval consistent graph generation algorithm as a general discretization approach of state space.

### Definition

Assume a general dynamic system whose state equation is $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ denotes a continuous state at time $t$ and $\boldsymbol{u}_t \in \mathbb{R}^{d'}$ denotes a control input, so that $\gamma_{i,\min} < u_{t,i} < \gamma_{i,\max}, i = 1, \ldots, d'$. A graph, $\mathcal{G}$, consists of a finite set of nodes $\mathcal{V} \subset \mathbb{R}^d$ and edges $\mathcal{E}$.

### Guarantee of Interval Consistency in State Transition for Arbitrary Time Interval

It is not trivial to guarantee the interval consistency of state transition under the constraint of control inputs with the conventional approach in (Byravan et al. 2015), which is based on Euclidean proximity on state space, so we focus on the fact that state transition should be based on the state equation of the system.

For notational compactness, define $\mathcal{U} \subset \mathbb{R}^{d'}$ as $\mathcal{U} = \{\boldsymbol{u} \in \mathbb{R}^{d'} \mid \gamma_{i,\min} < u_i < \gamma_{i,\max}, i = 1, \ldots, d'\}$, and let $\Delta t$ denote a constant time interval between time $\tau$ and $\tau+1$ for arbitrary $\tau$. Given $\boldsymbol{x}_\tau$ and $\boldsymbol{x}_{\tau+1}$, we define the *transition possibility* in $\Delta t$ (also called the "one-step transition possibility") between them as the existence of a set $\mathcal{R} \subset \mathcal{U}$ defined in equation 4, i.e., $\mathcal{R} \neq \emptyset$.

$$\mathcal{R}(\boldsymbol{x}_\tau, \boldsymbol{x}_{\tau+1}) = \{\boldsymbol{u}_{\tau+1} \in \mathcal{U} \mid \|\boldsymbol{x}_{\tau+1} - f(\boldsymbol{x}_\tau, \boldsymbol{u}_{\tau+1})\|_2^2 < \varepsilon\},$$

where $\varepsilon > 0$. With this definition, the transition possibility in $2\Delta t$ (two-step transition possibility) between $\boldsymbol{x}_\tau$ and $\boldsymbol{x}_{\tau+2}$ is defined as below.

$$^\exists \boldsymbol{x}_{\tau+1} \in \mathbb{R}^d,\ \mathcal{R}(\boldsymbol{x}_\tau, \boldsymbol{x}_{\tau+1}) \neq \emptyset \text{ and } \mathcal{R}(\boldsymbol{x}_{\tau+1}, \boldsymbol{x}_{\tau+2}) \neq \emptyset$$

As a generalization, the $\kappa$-step transition possibility between $\boldsymbol{x}_\tau$ and $\boldsymbol{x}_{\tau+\kappa}$ is defined as follows.

$$^\exists \boldsymbol{x}_{\tau+1}, \ldots, {}^\exists \boldsymbol{x}_{\tau+\kappa-1},\ {}^\forall k = 1, \ldots, \kappa,\ \mathcal{R}(\boldsymbol{x}_{\tau+k-1}, \boldsymbol{x}_{\tau+k}) \neq \emptyset$$

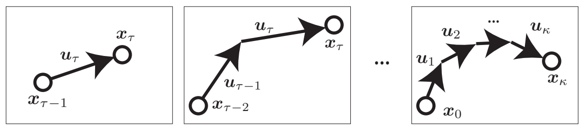

Figure 1: Definition of one, two, and $\kappa$-step state transition.

Figure 1 shows images of transition possibility in one step, two steps and $\kappa$ steps.

To verify the existence of $\mathcal{R}$ above, given $\boldsymbol{x}_\tau$ and $\boldsymbol{x}_{\tau+\kappa}$, we solve a minimization problem as below.

$$\min_{\boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa} \in \mathcal{U}} \left\| \boldsymbol{x}_{\tau+\kappa} - \tilde{f}(\boldsymbol{x}_\tau, \boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa}) \right\|_2^2 \quad (4)$$

, where $\tilde{f}(\boldsymbol{x}_\tau, \boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa})$ denotes a state after $\kappa$ steps from $\boldsymbol{x}_\tau$ with $\kappa$ inputs $\boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa}$. We solve this problem with sequential Gauss-Newton optimization. In particular, we applied an iterative linear least-square technique to be able to deal with a non-linear state equation. Define extended Jacobian matrix $J$ recursively as a lower triangular matrix like

$$J = \begin{bmatrix} J_{\tau+1,\tau+1} & O & \cdots & O \\ J_{\tau+2,\tau+1} & J_{\tau+2,\tau+2} & \cdots & O \\ \vdots & \vdots & \ddots & \vdots \\ J_{\tau+\kappa,\tau+1} & J_{\tau+\kappa,\tau+2} & \cdots & J_{\tau+\kappa,\tau+\kappa} \end{bmatrix} \quad (5)$$

$$J_{t_1,t_2} = \frac{\partial f(\boldsymbol{x}_{t_1-1}, \boldsymbol{u}_{t_1})^\mathsf{T}}{\partial \boldsymbol{u}_{t_2}} = \begin{cases} \frac{\partial f(\boldsymbol{x}_{t_1-1}, \boldsymbol{u}_{t_1})^\mathsf{T}}{\partial \boldsymbol{u}_{t_1}} & \text{if } t_1 = t_2 \\ \frac{\partial f(\boldsymbol{x}_{t_1-1}, \boldsymbol{u}_{t_1})^\mathsf{T}}{\partial \boldsymbol{x}_{t_1-1}} J_{t_1,t_1-1} & \text{if } t_1 > t_2 \\ O & \text{otherwise.} \end{cases}$$

Let $\boldsymbol{u}_{\tau+1:\tau+\kappa} = [\boldsymbol{u}_{\tau+1}{}^\mathsf{T} \ldots \boldsymbol{u}_{\tau+\kappa}{}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{\kappa d'}$. Given a difference between two points $\delta \boldsymbol{x} = \boldsymbol{x}_{\tau+\kappa} - \tilde{f}(\boldsymbol{x}_\tau, \boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa})$, we obtain $\delta \boldsymbol{u}_{\tau+1:\tau+\kappa} \in \mathbb{R}^{\kappa d'}$ subject to $\delta \boldsymbol{x} = J_{\tau+\kappa} \delta \boldsymbol{u}_{\tau+1:\tau+\kappa}$, where $J_{\tau+\kappa} = [J_{\tau+\kappa,\tau+1} \ldots J_{\tau+\kappa,\tau+\kappa}]$. Then, the $\kappa$–step transition possibility is validated with $\boldsymbol{u}_{\tau+1:\tau+\kappa}$ estimated by iterative calculation. This $\delta \boldsymbol{u}_{\tau+1:\tau+\kappa}$ is calculated like $\delta \boldsymbol{u}_{\tau+1:\tau+\kappa} = J^\dagger_{\tau+\kappa} \delta \boldsymbol{x}$, where $J^\dagger_{\tau+\kappa}$ denotes the pseudo inverse of $J_{\tau+\kappa}$.

### Interval Consistent Graph Generation Algorithm

An interval consistent graph is generated automatically given time interval parameter $\kappa$ and the number of actions $|A|$. First, initial coarse node set $\mathcal{V}_{\text{initial}} \subset \mathbb{R}^d$ is sampled randomly. Then, sequentially search $\kappa$–step transitable nodes $\mathcal{E}(s)$ from state $s$ from $\mathcal{V}$ with the approach mentioned above.If $|\mathcal{E}(s)| < |A|$, generate nodes with sampled control inputs $\boldsymbol{u}_{\tau+1}, \ldots, \boldsymbol{u}_{\tau+\kappa} \sim \mathcal{U}$ and add them to $\mathcal{V}$ and $\mathcal{E}(s)$. The termination condition of graph generation is ${}^\forall s \in \mathcal{V}$, $|\mathcal{E}(s)| = |A|$ or ${}^\forall s$ such that $|\mathcal{E}(s)| < |A|$, and there exists no transitable area in bounded state space. Thereby, $\kappa$-step interval consistency in state transition is guaranteed.

As mentioned above, the number of states $|S|$ can be controlled with $\kappa$ and $|A|$. With a larger $\kappa$, the graph becomes coarser, and the calculation cost decreases; however, prediction performance gets worse. With a larger $|A|$, in comparison, a fine graph is generated, and prediction performance

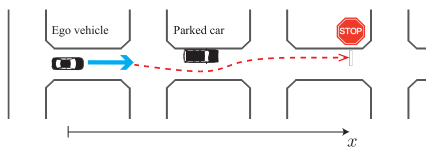

Figure 2: Road segment as modeling target for residential roads.

improves, but the calculation cost increases. with $\kappa$ and $|A|$ varied.

## Experiment

### Driving Behavior Prediction on Residential Roads

In this paper, we evaluate our proposed method with real driving behavior data on residential roads, where there exist both vehicles and pedestrians. Fatal accidents of pedestrians or cyclists are mainly caused by their sudden appearing in front of cars from behind a parked car or an intersection on residential roads; therefore, not only the deceleration around blind areas but also obstacle avoidance by steering in the right direction are important for preventing such accidents. Since we want to model the speed and steering of cars, state space is defined as $\boldsymbol{x} = [x\ \dot{x}\ \alpha]^\mathsf{T} \in \mathbb{R}^3$, where $x$ is the position of a car and $\alpha$ is the degree of steering. The modeling target is driving behavior in linear road segments as shown in Figure 2. The state equation is regarded as linear because the displacement of the rudder angle is considerably smaller than that of position or steering like

$$\boldsymbol{x}_t = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 0 & 0 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \boldsymbol{u}_t \qquad (6)$$

, where $\boldsymbol{u} = [\ddot{x}\ \dot{\alpha}]^\mathsf{T}$ is the control input and $\Delta t$ is the time interval per one step.

With the existing discretization framework, the number of states $|S|$ increases tens of times that of (Shimosaka, Kaneko, and Nishi 2014), which dealt with position–velocity state space, by adding steering to state space. Three dimensional state space is never low dimensional in the field of MaxEnt IRL; therefore, promoting computation efficiency should be needed even in three dimensional state space. Moreover, the efficiency becomes more and more obvious at a higher dimensionality.

### Designing Feature Descriptors

To generate feature descriptors $\boldsymbol{f}$, we leveraged seven environmental factors in addition to the four used in (Shimosaka, Kaneko, and Nishi 2014), concretely, intersections, the corners of intersections, and the start and goal positions. We prepared three new factors, namely, road width near parked cars and both the endpoints and center point of the parked cars. Given the environmental factors, we represent eight kinds of activities as potentials in position–velocity–steering space. In addition to the five used in (Shimosaka, Kaneko, and Nishi 2014), we used three new activities, namely, suppression of steering far from parked cars, steering degrees, and deceleration around the cars. We generated multiple potential fields with them by varying the means and covariance matrix of Gaussian kernels.

### Experimental Data

We collected experimental data on residential roads with an experimental vehicle we set up and recruited a single expert driver from a taxi company as a test subject. The experimental data was obtained by driving on residential roads in (Shimosaka, Kaneko, and Nishi 2014). The number of trials was 22 (in each trial, the vehicle drove on a segment from the start position to stop line like in Figure 2), and the total travel distance of the dataset was about 6 km.

### Comparative Approaches

**Graph Generation Parameters for Interval Consistent Graph** We set $\kappa = 7$ and $|A| = 64$ as graph parameters in the proposed method. The number of automatically generated nodes, i.e. the number of states $|S|$ in MDPs, was 13,500.

**IRL with Mesh Grid State Space (MG-IRL)** In MG-IRL, state space is represented as a fine-grained mesh grid. We used $0.1$ m for $x$, $0.5$ m/s for $\dot{x}$, and 20 deg. for $\alpha$ as minimum units in the mesh grid. We applied the same bounds of state space to all approaches: $x \in [0\,\mathrm{m}, 100\,\mathrm{m}]$, $\dot{x} \in [0\,\mathrm{m/s}, 8.5\,\mathrm{m/s}]$, and $\alpha \in [-200\,\mathrm{deg.}, 200\,\mathrm{deg.}]$. As a result, the number of states $|S|$ was $100/0.1 \times 8.5/0.5 \times 400/20 = 340,000$. Actions were defined as a minimum set so as to represent dataset adequately in this discretization: positive transition for $x$, three levels $(0, \pm 1$ steps) for $\dot{x}$, and five levels $(0, \pm 1, \pm 2$ steps) for $\alpha$. That is, $|A| = 15$.

**Locally Optimal Continuous IRL (C-IRL)** In C-IRL (Levine and Koltun 2012), a fixed-length control input sequence is locally optimized in continuous space. C-IRL enables us to compute the gradient of a log partition function with Laplace approximation of the likelihood. C-IRL depends a great deal on the initial control input sequence because it guarantees only local optimality. In this experiment, we used zero vector as the initial control sequence, which means linear uniform motion, and set the length of sequence to 200 empirically.

**Graph-based IRL with Naively Connected Graph (GB-IRL)** In GB-IRL (Byravan et al. 2015), state space is represented as a coarse graph. Node set $\mathcal{V}$ is sampled randomly, and then, edge set $\mathcal{E}$ is generated with the k-Nearest Neighbor algorithm. This method, therefore, does not guarantee interval consistency in state transition on the graph. Since it is trivial that state transition is made only in a positive sense for $x$ under this experimental settings, we focus on the nodes whose $x$ value is larger than that of the target node in a k-Nearest Neighbor search. We set the number of states $|S|$ and the number of actions $|A|$ equal to those of the graph generated with our proposed method, that is, $|S| = 13,500$ and $|A| = 64$.

**Approximate IRL with Fitted Q Iteration (FQI-IRL)** FQI-IRL, proposed in (Huang et al. 2015), is an approxi-

Table 1: Experimental results.

| Method | $\mathrm{MHD}_{50}$ | $\mathrm{MHD}_{90}$ | Time[s] |
|---|---|---|---|
| MG-IRL | $0.058 \pm 0.013$ | $0.135 \pm 0.037$ | $786.6 \pm 48.7$ |
| C-IRL | $0.156 \pm 0.055$ | $0.306 \pm 0.090$ | $17.4 \pm 10.7$ |
| GB-IRL | $0.113 \pm 0.018$ | $0.206 \pm 0.036$ | $39.3 \pm 2.1$ |
| FQI-IRL | $0.149 \pm 0.041$ | $0.315 \pm 0.069$ | $10.0 \pm 0.7$ |
| **Proposed** | $\mathbf{0.032 \pm 0.014}$ | $\mathbf{0.105 \pm 0.063}$ | $\mathbf{32.1 \pm 2.6}$ |

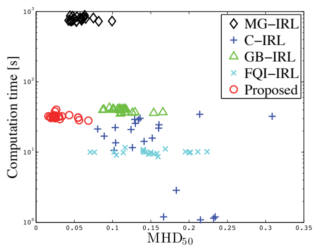

Figure 3: Scatter plot of $\mathrm{MHD}_{50}$ vs. computation time for four approaches.

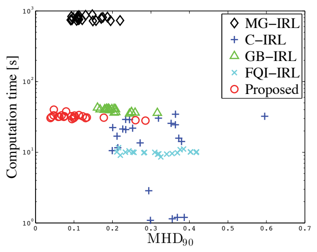

Figure 4: Scatter plot of $\mathrm{MHD}_{90}$ vs. computation time for four approaches.

mate approach for high-dimensional IRL that estimates policy with fitted Q iteration and calculates the gradient of a log partition function with Monte Carlo sampling. Though FQI-IRL treats continuous state space, action space must be quantized, so it was quantized in the same way as MG-IRL, i.e. $|A| = 15$. In fitted Q iteration, we use Gaussian kernels similar to (Huang et al. 2015) and set the deviation to $0.01$ for normalized state space. The number of Monte Carlo sampling was $100$.

## Evaluation Metric

As the evaluation metric for prediction performance, we used the modified Hausdorff distance (MHD) (Atev and others 2006), which is a generalized distance metric between point sequences. $\mathrm{MHD}_{50}$ and $\mathrm{MHD}_{90}$ represent the median and 90 percentile of MHD, respectively. We also used time spent on optimal path prediction as an indicator for computational cost. In MG-IRL, GB-IRL, and FQI-IRL, we measured the time spent on policy calculation in MDPs. In addition, we measured that on the optimization of control input in C-IRL because C-IRL does not calculate policy and optimized control input directly.

## Results and Discussions

Table 1 shows the means and standard deviations of $\mathrm{MHD}_{50}$, $\mathrm{MHD}_{90}$, and computational time for each approach. We also show the relationships MHD between computational time in Figure 3 and Figure 4. The horizontal axes indicate $\mathrm{MHD}_{50}$ and $\mathrm{MHD}_{90}$, and the vertical axes indicate computational time. Each point on these figures corresponds to each trial. The result implies that our approach achieves high performance with low computational cost.

For C-IRL, prediction finished faster than MG-IRL on average; however, the mean of prediction performance was worst for all approaches, and both performance and computational time were unstable due to its local optimality.

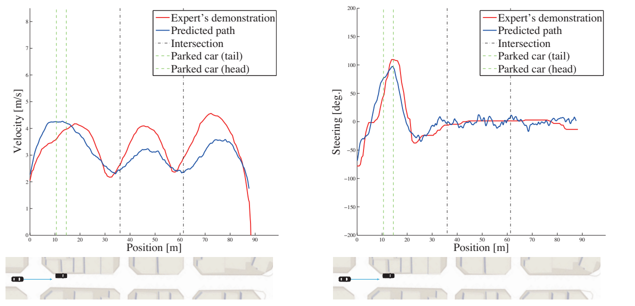

Figure 5: Example of prediction result when car is parked on the road.

GB-IRL achieved faster prediction than did MG-IRL and were more stable than C-IRL thanks to the coarse graph representation and its global optimality guaranty. Nonetheless, prediction performance was inferior to MG-IRL due to the lack of interval consistency guarantee in state transition. The computational time for FQI-IRL was steady and much faster than MG-IRL; however, the prediction performance was worse than MG-IRL and GB-IRL, which were homogeneously trained on the whole state space, due to its sampling based policy calculation. Here, the proposed approach predicted over 20 times faster than did MG-IRL with no drop in prediction performance. The difference in computational time was about 20 times in three dimensional state space, and the effectiveness of the proposed method will become more and more obvious when applied in higher dimensional state spaces.

Figure 5 shows a prediction result of driving behavior in the presence of a parked car on a road. On the left side of Figure 5, the horizontal axis and vertical axis indicate position and velocity, respectively, and on the right side, they indicate position and steering, respectively. The red line shows the demonstrated driving behavior of the expert driver, and the blue one shows the average of 100 sampled paths under the policy calculated with a learned reward function. The black chain line indicates the positions of intersections, and the green dashed line shows the tail and head points of the parked car. It is clear that our model predicted the steering behavior of the expert driver around a parked car from Figure 5.

## Conclusion

In this paper, we aimed for a fine-grained, fast driving behavior prediction framework based on IRL for high dimensional problems. To deal with an exponential increase in calculation cost depending on the number of dimensions, we proposed a novel inverse reinforcement learning that relies on coarse graphs nodes guaranteeing interval consistency in state transition. We conducted an experiment in three dimensional state space with actual driving data including avoidance behavior with steering. The results confirmed the effectiveness of the proposed framework when compared with the results obtained with conventional state-of-the-art IRL based approaches. Our future work includes online implementation with palatalization using GPGPUs.

## References

Abbeel, P., and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proc. of ICML*.

Atev, S., et al. 2006. Learning traffic patterns at intersections by spectral clustering of motion trajectories. In *Proc. of IROS*, 4851–4856.

Barry, J. L.; Kaelbling, L. P.; and Lozano-Pérez, T. 2011. DetH*: Approximate Hierarchical Solution of Large Markov Decision Processes. In *Proc. of IJCAI*, 1928–1935.

Byravan, A.; Monfort, M.; Ziebart, B.; Boots, B.; and Fox, D. 2015. Graph-based inverse optimal control for robot manipulation. In *Proc. of IJCAI*, 1874–1880.

Dragan, A., and Srinivasa, S. 2012. Formalizing assistive teleoperation. In *Proc. of RSS*.

Ernst, D.; Geurts, P.; and Wehenkel, L. 2005. Tree-based batch mode reinforcement learning. *JMLR* 6:503–556.

Farahmand, A.; Ghavamzadeh, M.; Szepesvari, C.; and Mannor, S. 2009. Regularized fitted q-iteration for planning in continuous-space markovian decision problems. In *Proc. of ACC*, 725–730.

Hoffman, M.; de Freitas, N.; Doucet, A.; and Peters, J. 2009. An expectation maximization algorithm for continuous markov decision processes with arbitrary rewards. In *Proc. of AISTATS*, 232–239.

Huang, D.-A., and Kitani, K. M. 2014. Action-reaction: Forecasting the dynamics of human interaction. In *Proc. of ECCV*, 489–504.

Huang, D.-A.; massoud Farahmand, A.; Kitani, K.; and Bagnell, J. 2015. Approximate maxent inverse optimal control and its application for mental simulation of human interactions. In *Proc. of AAAI*, 2673–2679.

Kalakrishnan, M.; Chitta, S.; Theodorou, E.; Pastor, P.; and Schaal, S. 2011. STOMP: Stochastic Trajectory Optimization for Motion Planning. In *Proc. of ICRA*.

Konidaris, G.; Osentoski, S.; and Thomas, P. 2011. Value function approximation in reinforcement learning using the fourier basis. In *Proc. of AAAI*, 380–385.

Levine, S., and Koltun, V. 2012. Continuous inverse optimal control with locally optimal examples. In *Proc. of ICML*, 41–48.

Levine, S.; Popović, Z.; and Koltun, V. 2011. Nonlinear inverse reinforcement learning with Gaussian processes. In *Advances in Neural Information Processing Systems 24*. 19–27.

Munos, R., and Szepesvári, C. 2008. Finite-time bounds for fitted value iteration. *JMLR* 9:815–817.

Ng, A. Y., and Russell, S. J. 2000. Algorithms for inverse reinforcement learning. In *Proc. of ICML*, 663–670.

Ohtsuka, T. 2004. A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica* 40(4):563 – 574.

Shimosaka, M.; Kaneko, T.; and Nishi, K. 2014. Modeling risk anticipation and defensive driving on residential roads with inverse reinforcement learning. In *Proc. of ITSC*, 1694–1700.

Taylor, G., and Parr, R. 2009. Kernelized value function approximation for reinforcement learning. In *Proc. of ICML*, 1017–1024.

Vernaza, P., and Bagnell, D. 2012. Efficient high dimensional maximum entropy modeling via symmetric partition functions. In *Proc. of NIPS*, 575–583.

Walker, J.; Gupta, A.; and Hebert, M. 2014. Patch to the future: Unsupervised visual prediction. In *Proc. of CVPR*, 3302–3309.

Ziebart, B. D., et al. 2008. Maximum entropy inverse reinforcement learning. In *Proc. of AAAI*, 1433–1438.

Ziebart, B. D.; Maas, A. L.; Dey, A. K.; and Bagnell, J. A. 2008. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Proc. of UbiComp*, 322–331.

Ziebart, B. D.; Dey, A.; and Bagnell, J. A. D. 2012. Probabilistic pointing target prediction via inverse optimal control. In *Proc. of IUI*.

Ziebart, B. D. 2010. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. Ph.D. Dissertation, Carnegie Mellon University.

Zucker, M.; Ratliff, N.; Dragan, A.; Pivtoraiko, M.; Klingensmith, M.; Dellin, C.; Bagnell, J. A. D.; and Srinivasa, S. 2013. CHOMP: Covariant Hamiltonian Optimization for Motion Planning. *International Journal of Robotics Research*.