Learning Variable Impedance Control via Inverse Reinforcement Learning for Force-Related Tasks

Xiang Zhang, Liting Sun, Zhian Kuang, and Masayoshi Tomizuka

***Abstract*—Many manipulation tasks require robots to interact with unknown environments. In such applications, the ability to adapt the impedance according to different task phases and environment constraints is crucial for safety and performance. Although many approaches based on deep reinforcement learning (RL) and learning from demonstration (LfD) have been proposed to obtain variable impedance skills on contact-rich manipulation tasks, these skills are typically task-specific and could be sensitive to changes in task settings. This letter proposes an inverse reinforcement learning (IRL) based approach to recover both the variable impedance policy and reward function from expert demonstrations. We explore different action space of the reward functions to achieve a more general representation of expert variable impedance skills. Experiments on two variable impedance tasks (Peg-in-Hole and Cup-on-Plate) were conducted in both simulations and on a real FANUC LR Mate 200iD/7 L industrial robot. The comparison results with behavior cloning and force-based IRL proved that the learned reward function in the gain action space has better transferability than in the force space. Experiment videos are available at https://msc.berkeley.edu/research/impedance-irl.html.**

***Index Terms*—Compliance and impedance control, learning from demonstration, machine learning for robot control.**

I. Introduction

ROBOT systems are increasingly deployed into various unstructured environments such as factories, warehouses and hospitals. In these environments, robots need to perform complex tasks while interacting with unknown environments in a safe and stable manner. Impedance control, which establishes a virtual mass-spring-damping contact dynamic, has been widely applied to these robot systems to guarantee safe physical interactions. Moreover, many complex manipulation tasks require the robot to change impedance according to the task phases. In practice, a variable impedance skill is needed in such tasks.

In recent years, several learning-based methods have been introduced to obtain variable impedance skills. Examples include learning from demonstrations (LfD) [1]–[3] and deep reinforcement learning (RL) with variable impedance action spaces [4]–[6]. However, the task-specific impedance skills obtained by LfD approaches may fail when the task changes. Besides, designing a suitable reward function is challenging for RL. Therefore, their skill transferability is limited. A more general way of learning variable impedance skills needs to be found to tackle these problems.

The development of the inverse reinforcement learning (IRL) method provides a new insight on learning from demonstration. The goal of IRL is to recover the expert reward or cost function from demonstrations. New policies can then be obtained by reoptimizing this learned reward in new scenarios. Previously, IRL has been applied to several tasks such as plate placing [7] and route planning [8]. However, there is no previous work applying the IRL method to the variable impedance control to our knowledge. Furthermore, the action space for the reward function for this task is still unclear. Should we define rewards directly on the force, or should we look for a reward function in terms of the impedance gain?

This letter proposes an inverse reinforcement learning based approach to recover both the variable impedance policy and the reward function from expert demonstrations. While this learned policy can be utilized to solve the original task, new variable impedance policies can be generated for different task settings by using RL to maximize the learned reward function. Since the learned reward function only depends on the current observation and action, it is agnostic to task settings. Thus, our approach is more general and has better transferability in comparison with previous LfD approaches.

Furthermore, similar to [6], we argue that, for force-related tasks, learning in the impedance gain action space is better than learning in the force action space. The basic idea is that the performance of gain policy is guaranteed by the impedance control law and therefore improves the reward transferability. For validation, we compare our approach with behavior cloning (BC) [9] baseline in two tasks: Peg-in-Hole and Cup-on-Plate (as shown in Fig. 1), and the influence of two action spaces (impedance gain and Cartesian space force) have also been studied. As a result, our approach successfully recovers the expert variable impedance policy and achieves better transfer performance than BC and the force-based IRL in the testing scenarios.

This letter's remainder is organized as follows: Related works are introduced in Section II. Our controller design and learning framework are outlined in Section III. Section IV presents the

Manuscript received October 15, 2020; accepted February 4, 2021. Date of publication February 23, 2021; date of current version March 11, 2021. This letter was recommended for publication by Associate Editor T. Kulvicius and Editor T. Asfour and upon evaluation of the reviewers' comments. *(Corresponding author: Xiang Zhang.)*

Xiang Zhang, Liting Sun, and Masayoshi Tomizuka are with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USA (e-mail: xiang_zhang_98@berkeley.edu; litingsun@berkeley.edu; tomizuka@me.berkeley.edu).

Zhian Kuang is with the Research Institute of Intelligent Control and Systems, Harbin Institute of Technology, Harbin 150001, China, and also with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USA (e-mail: zhiankuang@foxmail.com).

Digital Object Identifier 10.1109/LRA.2021.3061374

Fig. 1. Left: a human expert demonstrates on the robot, Right: the robot accomplishes the task with the learned variable impedance skill.

experiments in simulation and on the real robot. Finally, the conclusion is given in Section V.

II. Related Work

A. Force-Control Related Tasks in Robotics

Force control is essential for many robotics applications, such as the assembly, the surface polishing, and the machining. Previous works in this field can be categorized into two methods: pure force control and position-force control. The pure force control accomplishes tasks by directly controlling the robot's force in the Cartesian space or in the joint space. Examples can be found in the peg-hole assembly task [10] and visual manipulations tasks from [11]. For the position-force control, it can be further separated into the hybrid position/force control [12] and the impedance control [13]. In the first method, the position and force are controlled in two separate channels. Researchers have deployed this method on surface polishing tasks and peg-in-hole [3]. However, the position-force decoupling can only be achieved when the task is well-defined and may not be available for complex tasks [2]. Impedance control mitigates this limit by controlling the force applied by the robot when it deviates from the desired trajectory and has been introduced to the valve turning task [2] and several tools using tasks [14].

B. Variable Impedance Control

In many complex tasks, variable impedance skills are needed for robots to interact with the environment. Learning from demonstration approaches are utilized to learn the expert variable impedance skills. In [1], human experts control a robot's impedance by a hand-held impedance control interface. The expert data is then recorded by dynamical movement primitives (DMP) and learned by the regression method. Researches in [2] directly estimated the robot impedance from the human demonstrations and then encoded the robot skill by Gaussian Mixture Regression (GMR) with sensed forces. However, these previous works learned expert variable impedance skills on specific tasks which are difficult to transfer to a different task setting. In our approach, both the expert variable impedance skill and the reward function would be recovered, and new skills can be generated by reoptimizing the learned reward function with RL.

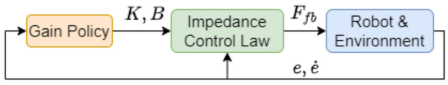
In the field of reinforcement learning, Buchli *et al.* [4] utilized policy improvement with path integrals (PI^2), which is a model-free method to learn the joint space variable impedance skills. However, the joint space impedance they are using limited policy transferability. Rey *et al.* [5] further improved this method by simultaneously learning trajectories and a state-dependent varying stiffness model, which is now represented in the robot end-effector frame. Martin *et al.* [6] compared the RL performance of different action spaces in robot manipulation tasks. They showed the variable impedance control in end-effector space (VICES) has an advantage in constrained and contact-rich tasks. However, the results from RL highly depend on the design of the reward function. With our method, the reward function can be recovered from expert demonstrations and generates new policy for different tasks with RL methods.

C. Inverse Reinforcement Learning

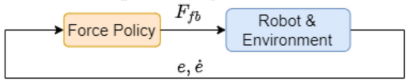
Inverse reinforcement learning is one of the learning from demonstrations methods. Unlike the traditional LfD approaches such as BC, where the main idea is to mimic expert actions, the goal of IRL is to infer the expert's reward function from expert demonstrations. Then the optimal policy is obtained by maximizing this reward function using forward RL. One commonly used framework for IRL is the maximum entropy IRL [8]. This framework assumes the expert trajectories follow a Boltzmann distribution with the cost and updates cost function by maximum likelihood learning. Levine *et al.* [15] further improved this framework to high dimensional and continuous tasks by using the local Laplace approximation of the cost function. However, these methods require the dynamics model for the cost function update, which is challenging to obtain in robot manipulation tasks.

Recently, Finn *et al.* [7] proposed a sample-based IRL approach to recover cost function in high dimensional state-action spaces. In this method, the agent alternates between optimizing the cost function and optimizing policy, which generates trajectories to minimize the cost. Since the optimizer requires no dynamics model, both the cost function and the policy are updated in a model-free way. Later the authors found that their method agrees with the generative adversarial network (GAN) formulation and introduced the GAN-GCL algorithm in [16]. One practical problem for GAN-GCL is that it evaluates the full trajectory and results in high variance estimates. Adversarial inverse reinforcement learning (AIRL) [17] improved the performance by extending the GAN-GCL algorithm to single state-action pairs and achieved superior results in simulation.

Although many IRL algorithms employ entropy regularisation to prevent the simply mimicking the expert policy, there is no previous work focus on the effect of action space selection to the authors' knowledge. In our method, by introducing variable impedance gain action space, we can find more general representation of the expert policy than using force as action and improve the reward function transfer performance in a new task setting.



(a) Impedance gain controller



(b) Force controller

Fig. 2. Controller design.

Algorithm 1: Learn Variable Impedance Skills with AIRL.

Collect expert demonstration trajectories τ_i^D
Initialize impedance gain policy π and reward function r with random weights
for *iteration k = 1 to K* **do**
  Collect trajectories τ_i under policy π
  Update the reward function by minimizing (7)
  Evaluate trajectories τ_i with the reward function
  Update policy π to maximize the reward by TRPO
end

III. Proposed Approach

A. Cartesian Space Impedance Control

Consider the dynamics model of the robot in the Cartesian space:

$$\boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{G}(\dot{\boldsymbol{x}}) = \boldsymbol{J}^{-\boldsymbol{T}}\boldsymbol{\tau} - \boldsymbol{F}_{ext} \quad (1)$$

where $\boldsymbol{M}(\boldsymbol{x})$ is the mass-inertia matrix, $\boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ denotes the Coriolis matrix, $\boldsymbol{G}(\dot{\boldsymbol{x}})$ is the gravity vector, $\ddot{\boldsymbol{x}}$, $\dot{\boldsymbol{x}}$ and $\boldsymbol{x}$ are respectively the Cartesian acceleration, velocity and position of the end-effector, $\boldsymbol{J}$ is the Jacobian matrix and $\boldsymbol{\tau}$, $\boldsymbol{F}_{ext}$ represents the joint space motor torque input and the external force, respectively. Under the impedance control law, the robot will behave as a mass-spring-damping system, which follows the dynamics equation:

$$\boldsymbol{M_d}(\ddot{\boldsymbol{x}} - \ddot{\boldsymbol{x}}_{\boldsymbol{d}}) + \boldsymbol{B_d}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{\boldsymbol{d}}) + \boldsymbol{K_d}(\boldsymbol{x} - \boldsymbol{x_d}) = -\boldsymbol{F}_{ext} \quad (2)$$

where $\boldsymbol{M_d}, \boldsymbol{B_d}, \boldsymbol{K_d}$ are the desired mass, damping and stiffness matrices. By solving (1), (2) and setting $\boldsymbol{M_d} = \boldsymbol{M}(\boldsymbol{x})$, the impedance control law can be written as:

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{J}^{\boldsymbol{T}}\boldsymbol{F} \\ \boldsymbol{F} &= \boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}}_{\boldsymbol{d}} + \boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{G}(\boldsymbol{x}) \\ &\quad -\boldsymbol{B_d}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{\boldsymbol{d}}) - \boldsymbol{K_d}(\boldsymbol{x} - \boldsymbol{x_d}) \end{aligned} \quad (3)$$

This impedance control law can be further separated into two parts: the feed-forward term $\boldsymbol{F_{ff}}$ to cancel the nonlinear robot dynamics and the feedback term $\boldsymbol{F_{fb}}$ which tracks the desired trajectory:

$$\boldsymbol{F_{ff}} = \boldsymbol{M}(\boldsymbol{x})\ddot{\boldsymbol{x}}_{\boldsymbol{d}} + \boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \boldsymbol{G}(\boldsymbol{x}) \quad (4)$$

$$\begin{aligned} \boldsymbol{F_{fb}} &= -\boldsymbol{B_d}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{\boldsymbol{d}}) - \boldsymbol{K_d}(\boldsymbol{x} - \boldsymbol{x_d}) \\ &= -\boldsymbol{B_d}\dot{\boldsymbol{e}} - \boldsymbol{K_d}\boldsymbol{e} \end{aligned} \quad (5)$$

where $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ are the tracking error and the tracking velocity. The stiffness matrix $\boldsymbol{K_d}$ and the damping matrix $\boldsymbol{B_d}$ are also known as the impedance gain matrices, since they map the tracking error and velocity to the feedback force $\boldsymbol{F_{fb}}$. To simplify the notations, we use $\boldsymbol{K}$ (stiffness) and $\boldsymbol{B}$ (damping) to represent $\boldsymbol{K_d}$ and $\boldsymbol{B_d}$ in the rest of letter.

B. Learning Variable Impedance Skills With AIRL

Fig. 2 depicts our controller design. In our approach, the observations from the robot and the environment are the tracking error $\boldsymbol{e}$ and the tracking velocity $\dot{\boldsymbol{e}}$. Our policy takes in the observations and outputs either the impedance gain $\boldsymbol{K}, \boldsymbol{B}$ or the feedback force $\boldsymbol{F_{fb}}$, depending on the action space design. The impedance gain controller then calculates the control inputs by (3) and controls the robot.

We employ AIRL [17] to learn both the expert policy and the reward function and the training procedure is detailed in Algorithm 1. In this adversarial training setting, the discriminator which separates the generator trajectories and the expert trajectories is defined as:

$$D_\theta(o, a) = \frac{exp(r_\theta(o, a))}{exp(r_\theta(o, a)) + \pi(a|o)} \quad (6)$$

where $r_\theta(o, a)$ is the reward function we want to learn and $\pi(a|o)$ is the probability of taking action a at observation o under current policy. The discriminator is updated to minimize this loss [17]:

$$L(\theta) = \sum_{t=0}^{T} -\mathbb{E}_D[logD_\theta(o_t, a_t)] - \mathbb{E}_{\pi_t}[1 - logD_\theta(o_t, a_t)] \quad (7)$$

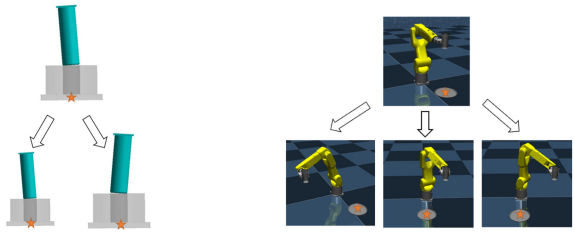
The generator is the variable impedance policy. During the training, the policy is updated to maximize the trajectory reward, which is evaluated by the reward function. In our approach, we use TRPO [18], which is a policy gradient based RL method to update the policy.

Since the environment dynamics are unknown, we applied RL to reoptimize a new policy in a different task setting to test the performance of the learned reward function. In the RL process, the policy update is the same as the IRL but with the fixed learned reward function.

IV. Experiments

To evaluate our proposed approach, we conduct experiments in several robotic tasks, both in simulation and on a real robot. In the simulation, we design two variable impedance control tasks: Peg-in-Hole task and Cup-on-Plate task. For the experiment validation, we collect human expert data for the Cup-on-Plate task by kinesthetic teaching and test learned policies on the real robot.

To validate our approach's generalizability, we compare the performance of our approach with four baseline methods: 1) force-based AIRL, 2) gain-based BC, 3) force-based BC and 4) constant gain. We hypothesize that using the impedance gain as action space improves the transferability to the testing



(a) The Peg-in-Hole task (b) The Cup-on-Plate task

Fig. 3. Training and testing scenarios for (a) the Peg-in-Hole task and (b) the Cup-on-Plate task, orange stars indicate the goal points.

TABLE I
PEG-IN-HOLE SUCCESS RATE IN DIFFERENT TILT ANGLES

Tilt angles	$-6°$	$-4°$	$-2°$(T)	$0°$	$2°$
Gain-AIRL	**91.7%**	**100%**	98.3%	**100%**	**100%**
Gain-BC	56.0%	100%	96.7%	100%	46.7%
Force-AIRL	53.3%	71.7%	**100%**	56.7%	0%
Force-BC	80.0%	95.0%	93.3%	100%	75.0%
Constant-Gain	6.7%	10.0%	36.7%	100%	31.7%
Gain-AIRL-His	**100%**	**91.7%**	**100%**	**100%**	**93.3%**
Gain-BC-His	63.3%	86.7%	90.0%	100%	55.0%
Force-AIRL-His	20.0%	0%	75.0%	21.6%	31.6%
Force-BC-His	86.7%	86.7%	93.3%	83.3%	83.3%

scenarios, either via AIRL or BC, since the impedance control law remains effective when task setting changes. Furthermore, by recovering the expert reward function, learning gain in the impedance control via AIRL is a more general approach than BC.

A. Tasks in Simulator

1) Task Setups: As depicted in Fig. 3, two robotic tasks, Peg-in-Hole and Cup-on-Plate, are conducted in the Mujoco [19] simulation to evaluate the proposed approach. In the Peg-in-Hole task, the diameters of the peg and hole are 25.37 mm and 25.40 mm, respectively. For the Cup-on-Plate task, FANUC LR Mate 200iD/7 L industrial robot is included into simulation and the robot's goal is to place a cup on a plate in a fast and steady manner. In two simulation environments, model parameters such as the mass-inertia matrix $\boldsymbol{M}(\boldsymbol{x})$, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ and the gravity vector $\boldsymbol{G}(\boldsymbol{x})$ are calculated by Mujoco automatically using the simulation model.

2) Observation Spaces: We use the tracking error e and tracking velocity $\dot{e}$ together as observations for two tasks. The goal points are shown in Fig. 3, and the end-effectors are located on the center of mass (COM) of the peg for the Peg-in-Hole task and on the cup for the Cup-on-Plate task.

Moreover, since a single pair of e and $\dot{e}$ doesn't provide acceleration information and may not fully represent the system dynamics. We also employ an augmented observation space which contains a history of e and $\dot{e}$ from the last five time steps for evaluation. In Table I,II and III, evaluations on the augmented observation space are marked with "His" which denotes using a history of e and $\dot{e}$.

3) Action Spaces: There are two options for the action space selection, the force action, and the impedance gain action. For the force action case, our policy outputs the Cartesian space force exerted by the robot. For the impedance gain action space, our policy outputs impedance gains and the control input are obtained by equation (5).

TABLE II
PEG-IN-HOLE SUCCESS RATE IN DIFFERENT MESH SCALES

Mesh scale	0.3	0.5	0.7	0.9	1.0(T)
Gain-AIRL	**91.7%**	**90.0%**	**95.0%**	**100%**	98.3%
Gain-BC	70.0%	80.0%	93.3%	91.7%	96.7%
Force-AIRL	0%	0%	0%	91.6%	**100%**
Force-BC	16.7%	40.0%	75.0%	90.0%	93.3%
Constant-Gain	28.3%	43.3%	11.7%	23.3%	36.7%
Gain-AIRL-His	**86.7%**	**90.0%**	**91.7%**	**100%**	**100%**
Gain-BC-His	33.3%	66.7%	83.3%	83.3%	90.0%
Force-AIRL-His	0%	21.7%	0%	26.7%	75.0%
Force-BC-His	6.7%	50.0%	63.3%	76.7%	93.3%

TABLE III
CUP-ON-PLATE RELATIVE PERFORMANCE DIFFERENCE

	Training	T1	T2	T3	T4
Gain-AIRL	**0.01**	**0.09**	**0.01**	0.19	**0.24**
Gain-BC	0.02	0.17	0.10	0.25	0.36
Force-AIRL	0.1	0.55	0.58	0.32	0.26
Force-BC	0.05	0.40	0.20	0.42	0.58
Constant-Gain	0.14	0.18	0.18	**0.17**	0.39
Gain-AIRL-His	0.04	**0.11**	**0.01**	**0.12**	**0.38**
Gain-BC-His	0.04	0.12	0.12	0.15	0.53
Force-AIRL-His	**0.02**	0.69	0.53	0.51	0.54
Force-BC-His	0.06	0.25	0.29	0.36	0.65

To reduce the dimension of the gain action space, we suppose that the stiffness matrix $\boldsymbol{K}$ and damping matrix $\boldsymbol{B}$ are diagonal. Therefore, our policy outputs diagonal elements of two matrices rather than full matrices. Furthermore, by enforcing diagonal elements to be positive, we can ensure the stiffness matrix $\boldsymbol{K}$ and the damping matrix $\boldsymbol{B}$ are positive definite. To extend our approach to the full matrix case, Cholesky decomposition can be utilized to guarantee $\boldsymbol{K}, \boldsymbol{B} > 0$.

Different impedance gain outputs are utilized for two tasks according to tasks. In the Peg-in-Hole task, the peg velocity is small, and the stiffness plays a main role in the insertion. Thus, the policy outputs a 6-dimensional stiffness, $[K_1, K_2, K_3, K_4, K_5, K_6]$ with fixed damping term, where $K_i (i = 1, 2, 3)$ denote the stiffness for the position error and $K_i (i = 4, 5, 6)$ denote the stiffness for the orientation error. For the Cup-on-Plate task, the tracking velocity is large, and the damping term affects the performance. Therefore, the output of our policy is now $[K_1, \ldots, K_6, d]$, which contains an extra damping factor d. The stiffness and damping matrices can then be obtained by:

$$K = diag(K_1, K_2, K_3, K_4, K_5, K_6) \quad B = d\sqrt{K}$$

We use a 1-dimensional damping factor rather than another 6-dimensional damping for dimension reduction.

4) Generating Expert Data: Fifty expert trajectories are collected for both tasks by two designed variable impedance controllers. In the Peg-in-Hole task, the expert is a fixed tip stiffness controller. The stiffness of the peg tip can be transferred to the COM frame by:

$$K_{COM} = J_{tip}^T \, K_{tip} \, J_{tip} \tag{8}$$

where K_{tip} denotes the stiffness matrix of the peg tip and J_{tip} is the Jacobian matrix between COM and the tip, which depends on the peg's configuration. Thus, this fixed tip stiffness is a variable stiffness on the COM. The uncorrelated stiffness $diag(K_1, K_2, K_3, K_4, K_5, K_6)$ is solved by:

$$diag(K_1, K_2, K_3, K_4, K_5, K_6)\, e = K_{COM} e \tag{9}$$

where e is the tracking error.

In the Cup-on-Plate task, the variable impedance controller contains three phases:

$$controller\ phases = \begin{cases} accelerating, & \text{if } e_{pos} > e_1 \\ switching, & \text{if } e_1 > e_{pos} > e_2 \\ reaching, & \text{if } e_2 > e_{pos} \end{cases}$$

where $e_{pos} = \sqrt{e_x^2 + e_y^2 + e_z^2}$ indicates the Cartesian space position tracking error and e_1, e_2 are two gain changing points which are 0.4 m and 0.2 m. As shown in Fig. 5(a), our designed expert control law chooses the largest gain to accelerate in the accelerating phase and generally switching to the smaller gain in the switching phase. In the reaching phase, the robot approaches the plate with minimum speed to guarantee safety.

5) Performance Score: In the experiment, we designed performance functions to evaluate different policies. In the Peg-in-Hole task, the performance function outputs a constant penalty if the peg is not aligned with the hole. Otherwise, it penalizes the z direction error, which is the vertical distance to the target point.

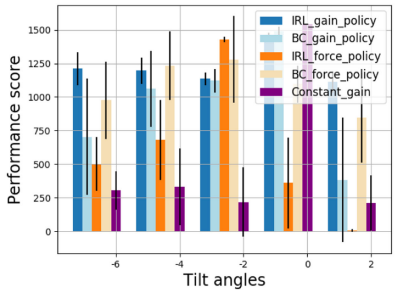
The designed performance function in the Cup-on-Plate task only penalizes the tracking error e when in the accelerating phase and the tracking velocity $\dot{e}$ in the reaching phase. If in the switching phase, the the performance function penalizes both e and $\dot{e}$.

6) Training Details: For the network architectures, our reward function is a 2-layer neural network with 32 units. For the policy, we use a 2-layer gaussian policy with 32 units when using a single pair of e and $\dot{e}$ as observation. The units number increases to 128 for the augmented observation space. The activation function is Tanh for the policy and Relu for the reward function.

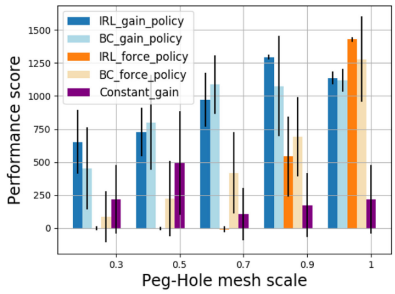
For the Learning hyper-parameters, the batch size and trajectory length of TRPO are 8000 and 200, respectively in the Peg-in-Hole task. For the Cup-on-Plate task, we use a batch size of 10 000 and trajectory length of 500.

7) Evaluations: In evaluation, we set two metrics to compare different approaches, which are the imitation learning performance in the training scenario and the transfer learning performance in the testing scenarios.

Peg-in-Hole task: We first evaluate the imitation learning results of four policies in the training scenario. The mean performance score and the mean success rate of three runs are given in the rightmost column of Fig. 4(b) and marked with (T) in Table I and Table II. We observe that all policies achieve high success rates in accomplishing the Peg-in-Hole task and their performances are similar, which approves these polices successfully imitate the expert in the training.



(a) Testing scenarios with different tilt angles



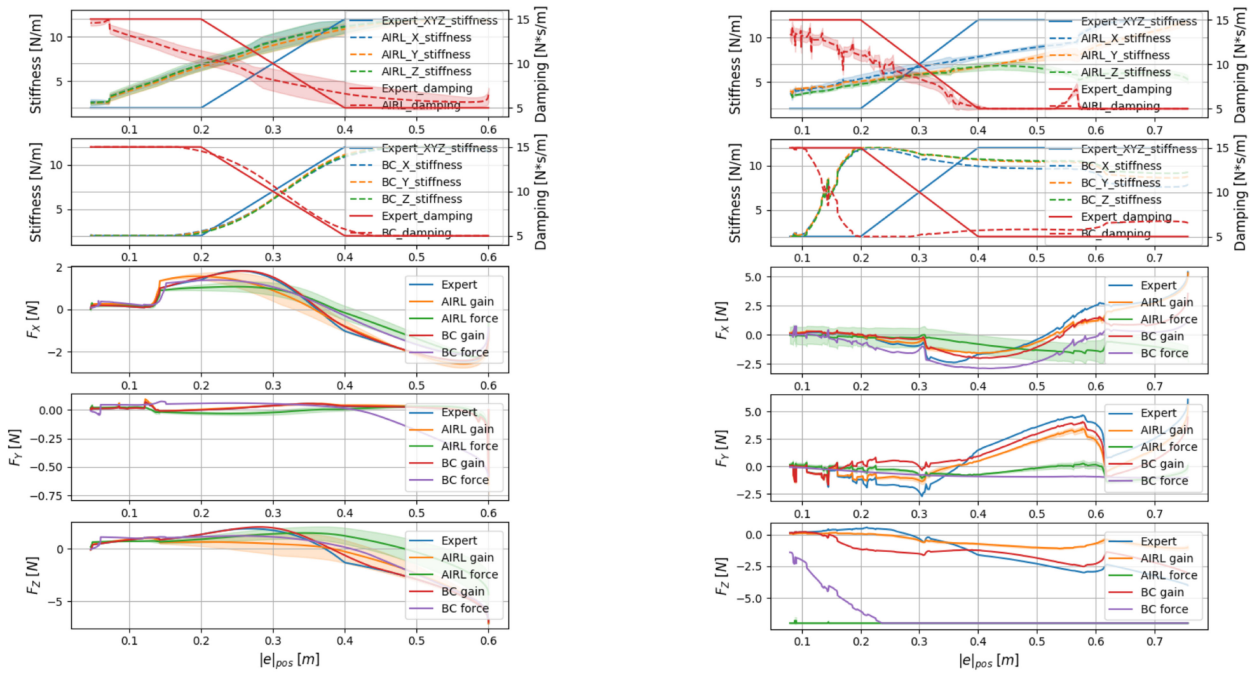
(b) Testing scenarios with different mesh scales

Fig. 4. Transfer performance on the Peg-in-Hole task.

We then transfer the learned reward functions obtained by AIRL to generate new policies in the testing scenario. Since Peg-in-Hole is a difficult task, the initial policy influences the final reoptimizing results. In one setting, we select the three best policies from five random runs to evaluate the transfer performance to mitigate this effect. For BC, we directly transfer the policy obtained in training to the testing scenarios.

In the first testing scenario, we change the tilt angles of the peg. Results are depicted in Fig. 4(a) and Table I. We find that gain-based AIRL achieves the highest success rate and outperforms all the other policies. Two BC policies achieve a high success rate when the tilt angle is close to -2°, but fails to generalize to tilt angles -6° and 2°, where trajectories deviate from expert demonstrations. The force-based AIRL has zero success rate when the tilt angles are 2°. In this case, the force policy imitates the expert action in the training, pushing the peg in the wrong direction from the hole and falls. However, since the sign of the tracking error e changes, the feedback force generated by the impedance control law also reverses the direction. Thus the gain policy is still valid when the tilt angle changes and avoids getting stuck in the middle. For the constant gain baseline, it achieves 100% success rate when the tilt angle is 0°, which is the simplest scenario. However, since it doesn't utilize the variable impedance control gain, its success rate is much lower than our proposed approach when increasing the tilt angle.

For the second testing scenario, we use smaller mesh scales of the peg and hole. Fig. 4(b) depicts the performance scores results. We notice that using gain as action improves the transferability, either for AIRL or BC. The reason is that, as shown



(a) Policy comparisons in the training scenario

(b) Policy comparisons in the testing scenario

Fig. 5. Policy comparisons for Cup-on-Plate task: two top figures of (a) and (b) compare the policy outputs of gain-based AIRL and BC with the expert gain policy; three bottom figures of (a) and (b) show the force applied by policies in x, y, z directions.

in Fig. 2, the feedback force of the gain policy is generated by the impedance control law, which is still effective when task setting changes. However, the force policy doesn't have this advantage and could be infeasible in the new dynamics. The performance score difference is negligible for gain-based AIRL and BC. However, as shown in Table II, the gain-based AIRL achieves higher successful rate when the mesh scale is small.

We also tested our approach with all the baselines on the augmented observation space. As shown in Table I and Table II, our approach still achieves the highest success rate and has similar performance in comparison with the original observation space. However, since the dimension of the augmented observation space is much larger, the force-based AIRL policy is more difficult to train and has even worse generalization results.

Cup-on-Plate task: The first column of Table III shows the relative performance difference with the expert in training. This difference is a normalized performance score difference with the expert, and a larger value indicates a larger performance difference. In the training scenario, all four policies accomplish the Cup-on-Plate task and achieve a similar performance score compared to the expert. We also compare the gains and forces applied by learned policies, and the results are depicted in Fig. 5(a). We observe that both the gain-based AIRL and the force-based AIRL recover the most features of the expert. Since the objective of BC encourages the policy to directly mimic the expert action, two BC baselines achieve the best performance in imitating expert gain and force policies.

In the testing, the robot end-effector is initialized in four different positions. The performance scores in comparison with the expert are given in Table III, and four initial testing points are named from T1 to T4. As a result, gain-based AIRL outperforms other policies in T1, T2, T4 and only falls behind the constant gain baseline in T3.

We explore the underlying reason for this difference by comparing the gains and forces. Fig. 5(b) depicts the policy comparison results for T1. In comparison, the gain-based AIRL recovers the expert gain changing trend and the expert force in x, y directions. However, it chooses small stiffness in z direction in the beginning and applies a much smaller force than the expert. This result may relate to the changes of robot dynamics in the new initial point. Since the mass matrix $M(x)$ in (3) is a function of the current robot configuration, the Cartesian space dynamics is coupled. We observe that, under the new setting, the force applied in y direction will also result in an acceleration in z direction, while this effect is negligible in the training scenario. This dynamics change could influence the way to maximize the reward function and therefore influences the learned policy.

For gain-based BC, it learns to set a small stiffness and large damping when the error is small. However, the gain switching happens too late, and the performance score is penalized for the final velocity. As depicted in Fig. 5(b), the force-based AIRL applies force in the wrong direction in x, y axes, and fails in the testing scenario. The reason is that the learned reward function for the force action encourages imitation of the expert force actions, which cannot drive the cup to the plate with the new initial point.

The evaluation results using the augmented observation space are shown in Table III. The augmented observation space doesn't improve the performance either for the gain action or for the

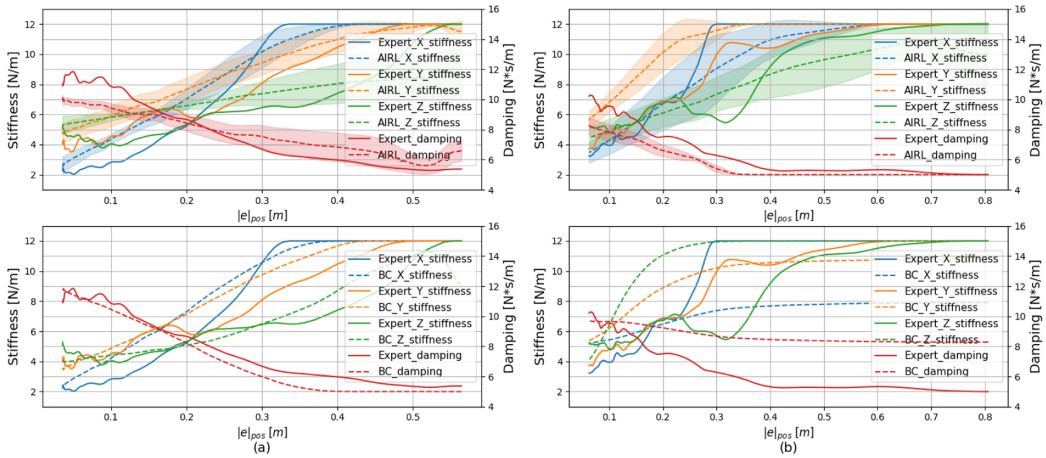


Fig. 6. Policy comparisons for the real robot: (a) gain-based AIRL and BC policies in the training, (b) gain-based AIRL and BC policies in the testing.

force action in the training and testing and our approaches still outperforms all the baselines.

B. Experiments on the Real Robot

We also evaluate our method on the real-world Cup-on-Plate task. In the real-world experiment, expert data is collected by a human expert on the real robot. Then the learned policies are transferred to the real robot for the performance evaluation.

1) Task Setup: The real-world experimental setup consists of the host and target computer, the F/T sensor, and the FANUC LR Mate 200iD robot [20]. We programmed the Cartesian variable impedance control algorithm on the host PC and it controls the real robot system which is connected to the target PC via Simulink Real-Time. The model parameters of the real robot such as the mass-inertia matrix $\boldsymbol{M}(\boldsymbol{x})$, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ and the gravity vector $\boldsymbol{G}(\boldsymbol{x})$ are obtained by the Euler-Lagrange method [21]

2) Collecting Human Expert Data: In the data collection process, the human expert applies force and torque on the end-effector to place cup on the plate. This 6-dimensional Cartesian space force and torque are measured by the F/T sensor and the control input is then calculated with (3). We record both the tracking state $[e, \dot{e}]$ and the human expert force together as the human expert data. The human expert gain is estimated in the data processing. We collect thirty expert trajectories with the same initial point as the training scenario in the simulation.

3) Gain Estimation Using Sliding Window Method: To recover the expert gain policy, we apply a similar method as [2], which uses a short sliding window to estimate stiffness and damping from the force. Each time window contains ten state-force pairs, and expert gain is estimated by solving (5) with Least Square.

4) Evaluation: With real-world human expert data, we learned both a policy and a reward function in the simulation environment by using AIRL. Training settings other than expert data remain the same as the simulation Cup-on-Plate task.

TABLE IV
AVERAGE TRAJECTORY DEVIATION ON THE REAL ROBOT

	Gain-AIRL	Gain-BC	Force-AIRL	Force-BC
Training	12.6 mm	9.5 mm	N/A	35.2 mm
Testing	13.4 mm	48.5 mm	N/A	N/A

TABLE V
FINAL DEVIATION ON THE REAL ROBOT

	Gain-AIRL	Gain-BC	Force-AIRL	Force-BC
Training	12.1 mm	12.0 mm	N/A	70.9 mm
Testing	10.8 mm	17.0 mm	N/A	N/A

We first evaluate the performance of two gain action approaches,which are gain-based AIRL and BC. Fig. 6(a) depicts the real-world expert gains and the recovered gain policies. Although the expert policy is noisy and the Sim-to-Real gap exists, both the gain-based AIRL and BC successfully recover the expert gains. We then collect three trajectories with the learned policy on the real robot. As shown in Table IV and Table V, average deviations with the expert of the gain-based AIRL and BC are close to 10 mm and the final deviations from the target point are both 12 mm.

We also evaluate two gain action approaches in the testing scenario with a different initial position. As shown in Fig. 6(b), the BC policy tends to apply constant damping and x direction stiffness in the testing scenario and causes lagging in comparison with the expert trajectory. However, the gain-based AIRL successfully recovers the expert variable impedance policy. Thus, in the testing, the average deviation from the expert and the final deviation are 13.4 mm and 10.8 mm, respectively, which are much smaller than the gain-based BC and in the same range as in the training scenario.

For the force-based AIRL, it cannot finish the Cup-on-Plate task and finally drives the robot out of the feasible workspace, both in the training and testing. Its failure is caused by the large sim-to-real gap on the force action space. As shown is Fig. 7, the velocity responses of the same force action in simulation and real-robot is too different which makes the learned policy in

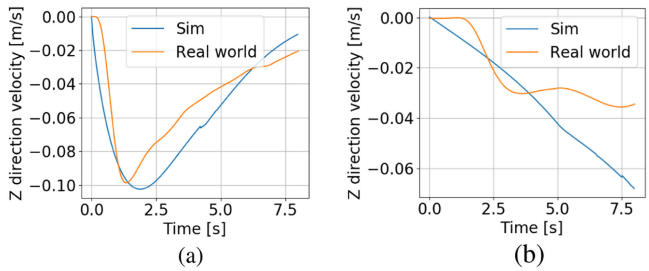


Fig. 7. Velocity response of the robot in simulation and real-world to same (a) gain action and (b) force action.

simulation hard to directly transfer to the real-world. However, for the gain policy, the feedback impedance control gain prevents the trajectory from deviating too much from the target point and therefore result in a much smaller sim-to-real gap for policy transfer. For the force-based BC, it can reach the target point in the training but with a much larger final deviation than two gain based methods. Moreover, it also fails in the testing scenario.

V. Conclusion

In this letter, we introduced an IRL based approach to recover variable impedance policies and reward functions from expert demonstrations. While the learned policy can be utilized in the original task, new impedance policies can be generated by optimizing the learned reward function for different task settings. We also explored the effect of action spaces selection on recovering expert rewards. Benefiting from the feedback control law, we argue that using gain as action can improve the reward transfer performance.

Experiments are conducted to evaluate our approach. In the simulation, the gain-based AIRL successfully imitates the expert demonstrations during the training and has better transfer learning results than all the baselines. For the real robot experiment, although the sim-to-real gap exists, our approach successfully recovers the expert trajectory in the training and outperforms BC in a different initial position.

Although our approach achieves an improved generalization results than traditional LfD methods, there still are some limitations. First, we employ a simplified impedance control law as in the previous references [4], [6], which doesn't measure the external force and results in a coupled Cartesian space dynamics. In the future work, we would like to decouple the dynamics by adding the external force into the impedance control law. Moreover, in our approach, we assume the waypoints are fixed and given in the task. In the future, we plan to include waypoints of the expert trajectory as one of the policy outputs and image of the workspace can be utilized as an input. In this way, our method can be extended to handle tasks with time-varying goal points.

References

[1] L. Peternel, T. Petrič, and J. Babič, "Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 1497–1502.
[2] F. J. Abu-Dakka, L. Rozo, and D. G. Caldwell, "Force-based learning of variable impedance skills for robotic manipulation," in *Proc. IEEE-RAS 18th Int. Conf. Humanoid Robots (Humanoids)*, 2018, pp. 1–9.
[3] T. Tang, H.-C. Lin, Y. Zhao, Y. Fan, W. Chen, and M. Tomizuka, "Teach industrial robots peg-hole-insertion by human demonstration," in *Proc. IEEE Int. Conf. Adv. Intell. Mechatronics*, 2016, pp. 488–494.
[4] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal, "Learning variable impedance control," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 820–833, 2011.
[5] J. Rey, K. Kronander, F. Farshidian, J. Buchli, and A. Billard, "Learning motions from demonstrations and rewards with time-invariant dynamical systems based policies," *Auton. Robots*, vol. 42, no. 1, pp. 45–64, 2018.
[6] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Macau, China, 2019, pp. 1010–1017, doi: 10.1109/IROS40897.2019.8968201.
[7] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 49–58.
[8] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. 23rd AAAI Conf. Artif. Intell.*, Chicago, IL, USA,Jul. 13–17, 2008, pp. 1433–1438.
[9] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Proc. Adv. Neural Inform. Process. Syst.*, 1989, pp. 305–313.
[10] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 819–825.
[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, 2016.
[12] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *ASME. J. Dyn. Sys., Meas., Control.*, vol. 103, no. 2, pp. 126–133, Jun. 1981, doi: 10.1115/1.3139652.
[13] N. Hogan, "Impedance Control: An Approach to Manipulation: Part II–Implementation," *ASME. J. Dyn. Sys., Meas., Control.*, vol. 107, no. 1, pp. 8–16, Mar. 1985, doi: 10.1115/1.3140713.
[14] Y. Li, G. Ganesh, N. Jarrassé, S. Haddadin, A. Albu-Schaeffer, and E. Burdet, "Force, impedance, and trajectory learning for contact tooling and haptic identification," *IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1170–1182, Oct. 2018.
[15] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012.
[16] C. Finn, P. Christiano, P. Abbeel, and S. Levine, "A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models," *CoRR*, vol. abs/1611.03852, 2016.
[17] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *CoRR*, vol. abs/1710.11248, 2017.
[18] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.
[19] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.
[20] Z. Kuang, X. Zhang, L. Sun, H. Gao, and M. Tomizuka, "Feedback-based digital higher-order terminal sliding mode for 6-dof industrial manipulators," 2021, *arXiv:2102.03531*.
[21] Z. Kuang, H. Gao, and M. Tomizuka, "Precise linear-motor synchronization control via cross-coupled second-order discrete-time fractional-order sliding mode," *IEEE Trans. Mechatron.*, vol. 26, no. 1, pp. 358–368, Feb. 2021.