Robot Policy Improvement With Natural Evolution Strategies for Stable Nonlinear Dynamical System

Yingbai Hu, *Student Member, IEEE*, Guang Chen, Zhijun Li, *Fellow, IEEE*, and Alois Knoll, *Senior Member, IEEE*

***Abstract*—Robot learning through kinesthetic teaching is a promising way of cloning human behaviors, but it has its limits in the performance of complex tasks with small amounts of data, due to compounding errors. In order to improve the robustness and adaptability of imitation learning, a hierarchical learning strategy is proposed: low-level learning comprises only behavioral cloning with supervised learning, and high-level learning constitutes policy improvement. First, the Gaussian mixture model (GMM)-based dynamical system is formulated to encode a motion from the demonstration. We then derive the sufficient conditions of the GMM parameters that guarantee the global stability of the dynamical system from any initial state, using the Lyapunov stability theorem. Generally, imitation learning should reason about the motion well into the future for a wide range of tasks; it is significant to improve the adaptability of the learning method by policy improvement. Finally, a method based on exponential natural evolution strategies is proposed to optimize the parameters of the dynamical system associated with the stiffness of variable impedance control, in which the exploration noise is subject to stability conditions of the dynamical system in the exploration space, thus guaranteeing the global stability. Empirical evaluations are conducted on manipulators for different scenarios, including motion planning with obstacle avoidance and stiffness learning.**

***Index Terms*—Dynamical system, exponential natural evolution strategies (NESs), imitation learning, policy improvement of robustness and adaptability.**

Manuscript received 23 March 2022; revised 20 May 2022; accepted 10 July 2022. This work was supported in part by the European Union's Horizon 2020 Framework Program for Research and Innovation under Grant 945539 (Human Brain Project SGA3); in part by the Shanghai Municipal Science and Technology Major Project under Grant 2018SHZDZX01; in part by ZJ Lab; in part by the Shanghai Center for Brain Science and Brain-Inspired Technology, in part by the Shanghai Rising Star Program under Grant 21QC1400900; in part by the National Natural Science Foundation of China under Grant 62133013 and Grant U2013601; in part by the Anhui Provincial Natural Science Foundation, Anhui Energy-Internet Joint Program under Grant 2008085UD01; and in part by the Major Science and Technology Projects of Anhui Province under Grant 202103a05020004. This article was recommended by Associate Editor B. Jiang. *(Yingbai Hu and Guang Chen contributed equally to this work.) (Corresponding author: Zhijun Li.)*

Yingbai Hu and Alois Knoll are with the Department of Informatics, Technical University of Munich, 85748 Munich, Germany (e-mail: yingbai.hu@tum.de; knoll@mytum.de).

Guang Chen is with the College of Automotive Engineering, Tongji University, Shanghai 201804, China, and also with the Department of Informatics, Technical University of Munich, 85748 Munich, Germany (e-mail: guangchen@tongji.edu.cn).

Zhijun Li is with the Department of Automation, University of Science and Technology of China, Hefei 230026, China (e-mail: zjli@ieee.org).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TCYB.2022.3192049.

Digital Object Identifier 10.1109/TCYB.2022.3192049

I. Introduction

EVER since the onset of pioneering research into robot learning methods of learning by demonstration have attracted much attention. Robot learning can facilitate applications in industry, manufacturing area, healthcare, etc., because it directly clones motor skills by extracting task-relevant information that can be transferred to the robot [1]–[3]. Generally, traditional imitation methods use supervised learning to obtain the regression parameters by modeling dynamic motion primitives (DMPs) [4] and Gaussian mixture models (GMMs) [5]. However, a major drawback of these methods is that they are not so adaptable and highly dependent on large amounts of data [2]–[6]. Therefore, they tend to be restricted to real-world robotic applications. In a real-world scenario, it is significant to design a more efficient learning policy based on the finite expert data.

Imitation learning has two central properties: 1) policy robustness and 2) adaptability. Among the many research papers contributing to this issue, some focus on parameter learning to improve robustness and adaptability. Khansari-Zadeh and Billard [7] presented a Gaussian model-based stable estimator called SEDS for learning the parameters of the dynamical system, which can ensure global stability at the goal point. Khansari-Zadeh and Billard [8] extended the dynamical system to cover various regression models and proposed a learning strategy for optimizing the valid Lyapunov function, which displays strong robustness in terms of disturbance from a random initial state [9], [10]. Duan *et al.* [11] and Jin *et al.* [12] employed various modification tricks to improve the metrics of a dynamical system subjected to global stability, for example, improving the learning speed using an extreme learning machine or improving accuracy means of manifold immersion and submersion. Although the above works have made some progress in improving the robustness of imitation learning, they suffer from a limited adaptability. Specifically, we want the robot might be perform specific special tasks that are not covered by in expert data. For example, a robot is expected to perform obstacle-avoidance tasks taking the shortest path, in which obstacles are configured along expert trajectories.

One possible solution is to combine policy-based high-level learning strategies with imitation learning [13]. Several learning strategies have been investigated for improving adaptability for different scenarios. In [14] and [15], the path integral policy improvement (PI^2) algorithm was tailored to learn the parameters of robot trajectories, in which reinforcement learning could adapt to learning tasks, such as via-points or obstacle-avoidance tasks. Indeed, these policy improvement methods were explored with the aim of optimizing the parameters with the feedback of a cost function set by users, in which the update rule was in the form of a probability-weighted average [16]. In addition to the aforementioned work, reinforcement learning could also be used to acquire the parameters of variable impedance control [17]–[19], with the robot's stiffness being changed according to the task requirements. In [20], reinforcement learning was proposed for learning the impedance parameters, in which the stiffness of the impedance controller was modeled as a dynamical equation and updated in accordance with rewards from the cost function. Similarly, in [21], a covariance matrix adaptation evolution strategy (CMA-ES) is proposed for learning the variable impedance control of robotic grasping, which provides a theoretical rule for assigning the highest weight to the best population for parameter updating.

Inspired by these works, we propose the exponential natural evolution strategies (NESs) algorithm to learn the stiffness of a stable nonlinear dynamical system. This combines the two properties of robustness and adaptability from kinesthetic teaching. As reported in [22], the NES provides a principled way of formulating the optimization problem based on the natural gradient. The proposed exponential NES is a more efficient learning algorithm than the previously mentioned CMA-ES method, because it obtains all parameter updating for covariance matrix adaptation from a single principle. Moreover, unlike the original NES, the parameter updating depends on the natural coordinate, which can reduce the computation of the inverse Fisher information matrix in NES [23].

In this article, we incorporate algorithms of exponential NES with a stable dynamical system into the imitation learning framework. First, the motions of the robot are encoded using the GMM from kinesthetic teaching, which is a normal behavioral cloning process. Then, the GMM-based stable dynamical system is derived from the Lyapunov stability theorem such that the optimized parameters of the dynamical system can be obtained from the stable condition. This process can improve the robustness of the dynamical system. Finally, the exponential NESs are explored for learning the parameters of the policy, which can further improve the stability and adaptability of the dynamical system for different tasks. The contributions of this article are summarized as follows.

1) Exponential NESs are proposed for learning the parameters of a policy that can improve the robustness and adaptability of the dynamical system, where the low-level learning is for behavioral cloning using GMM, and the high-level learning refers to the parameter learning of policy improvement considering robustness and adaptability.
2) Exponential NESs are also explored for learning the stiffness of the variable impedance control, where the stiffness of the controller can be modified online according to the task's requirements.
3) The proposed method can learn the covariance matrix parameter which is used to modify the exploration noise in the parameter space.
4) Offline learning and online robot experiments are conducted to demonstrate the effectiveness of the proposed scheme.

II. Related Work and Motivation

In this section, we will briefly discuss the motivation of the study and the related work in the areas of imitation learning and policy improvement in learning.

A. Imitation Learning From Demonstrations

Learning from demonstrations is a popular approach in robot motion imitation. Several classical supervised learning methods have been widely applied to behavioral cloning, such as Gaussian mixture regression (GMR) [24], Gaussian process regression (GPR) [25], and hidden Markov models (HMMs) [26]. However, these statistical approaches found locally optimal parameters by maximizing the likelihood such that failed to ensure global stability because they were not the theoretical solution to ensure the stability of the dynamical system. Dynamical system-based learning methods were therefore presented to ensure global stability, such as DMP in [27] and SEDS in [7]. In particular, the time-invariant dynamical system in [7] and the constraints of stability condition according to the Lyapunov stability are taken into consideration in the context of learning.

B. Policy Improvement With Learning

On the other hand, we focus on parameter learning to specify the task using policy improvement methods. The evolution strategies algorithm is one candidate solution for parameter learning, such as CMA-ES and NES. In contrast to PI^2 [28], the CMA-ES algorithm can modify the exploration noise during learning by updating its covariance matrix [21]. However, CMA-ES is not able to measure the proximity between the current policy and the updated policy on the basis of distribution in the learning process. We, therefore, consider exponential NES for this purpose. Generally, the NES learns the parameters of policy by following the natural gradient toward higher expected fitness [29]. Indeed, the traditional NES method involved computing the inverse Fisher information matrix, which can affect the efficiency of the learning process. The proposed exponential NES is a more efficient learning algorithm, because all the parameters updating for covariance matrix adaptation are obtained from a single principle [30].

III. Learning Problem

In this section, we will introduce the first learning problem, using GMR to model trajectories by kinesthetic teaching in Section III-A. To learn different tasks in imitation learning, we formulate the second problem to learn the parameters of the dynamical system in Section III-B. The control flow framework is shown in Fig. 1.

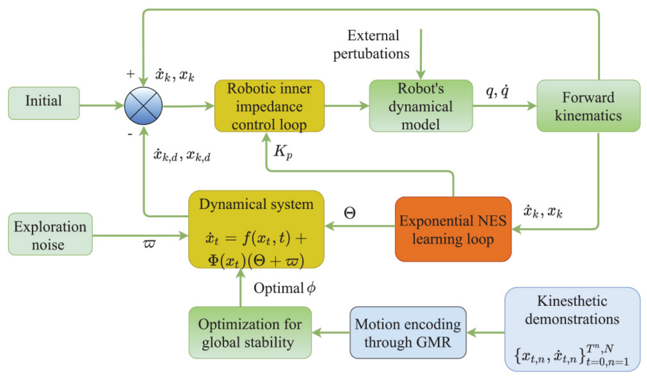
External pertubations
Initial
Robotic inner impedance control loop
Robot's dynamical model
Forward kinematics
$q, \dot{q}$
$\dot{x}_k, x_k$
$\dot{x}_{k,d}, x_{k,d}$
K_p
+
-
Exploration noise
ϖ
Dynamical system
$\dot{x}_t = f(x_t, t) + \Phi(x_t)(\Theta + \varpi)$
Θ
Exponential NES learning loop
$\dot{x}_k, x_k$
Optimal ϕ
Optimization for global stability
Motion encoding through GMR
Kinesthetic demonstrations
$\{x_{t,n}, \dot{x}_{t,n}\}_{t=0,n=1}^{T^n,N}$

Fig. 1. Control flow of the learning-control framework using the proposed methods to derive a stable and robust control policy for the robot system. Θ denotes the policy parameter, K_p is the stiffness of the impedance controller, $\dot{q}, q$ are the joint angle and velocity, and $\dot{x}, x$ are the state variable dynamical system.

A. Nonlinear Dynamical System From Demonstration

To imitate human skills, a data-driven kinesthetic teaching method is explored for learning motor skills, where the robot performs a repetitive task multiple times. Generally, the robot's motion can be encoded by learning methods, such as linear regression, support vector regression, and GMR. In this article, we formulate the robot's motions as an autonomous dynamical system with the state variable x and the output variable $\dot{x}$ as

$$\dot{x} = \hat{f}(x) \tag{1}$$

where x is the end-effector's trajectory in the Cartesian space, and $\dot{x}$ is the velocity; $\hat{f}(x)$ is the nonlinear continuous and continuously differentiable function.

The data points from demonstrations are defined as $(x_{t,n}, \dot{x}_{t,n})(t = 0, \ldots, T)$, where T is the time of the goal point and $N(n = 1, \ldots, N)$ is the number of samples. Each data-point includes a position value $x_{t,n}$ and a velocity value $x_{t,n}$. To encode the dataset of position distribution $p(x_{t,n}, \dot{x}_{t,n})$, the following GMM model is defined as:

$$p\big(x_{t,n}, \dot{x}_{t,n}; \phi\big) = \sum_{k=1}^{K} p(k) p\big(x_{t,n}, \dot{x}_{t,n}|k\big) \tag{2}$$

where K is the number of the Gaussian model; $p(k)$ denotes the prior probability, and $p(x_{t,n}, \dot{x}_{t,n}|k)$ is the conditional probability density function; $\phi_k = \{\lambda_k, \mu_k, \sum_k\}$, where λ_k, μ_k, and $\sum_k$ are prior probability, mean variable, and covariance variable, respectively. The parameters are defined as $\phi = \{\phi_1, \ldots, \phi_K\}$, and the details are expressed as

$$\begin{aligned} p(k) &= \lambda_k \\ p\big(x_{t,n}, \dot{x}_{t,n}|k\big) &= \frac{1}{\sqrt{(2\pi)^d \left|\sum_k\right|}} \\ &\quad \cdot \exp\left(-\frac{1}{2}\big([x_{t,n}, \dot{x}_{t,n}] - \mu_k\big)^{\mathrm{T}} \sum_k^{-1}\big([x_{t,n}, \dot{x}_{t,n}] - \mu_k\big)\right). \end{aligned}$$

For multiple demonstrations from kinesthetic teaching, the GMM encodes the set of trajectories of the robot in the Cartesian space. The k component of the GMM is defined as

$$\mu_k = \begin{Bmatrix} \mu_k^x \\ \mu_k^{\dot{x}} \end{Bmatrix}, \quad \Sigma_k = \begin{pmatrix} \Sigma_k^x & \Sigma_k^{x\dot{x}} \\ \Sigma_k^{\dot{x}x} & \Sigma_k^{\dot{x}} \end{pmatrix} \tag{3}$$

where μ_k and Σ_k are the mean and covariance matrices of the k component GMM. Therefore, the posterior mean estimate is deduced by GMR as

$$\dot{x} = \sum_{k=1}^{K} \frac{p(k)p(x|k)}{\sum_{i=1}^{K} p(i)p(x|i)} \left(\mu_k^{\dot{x}} + \Sigma_k^{\dot{x}x}\big(\Sigma_k^x\big)^{-1}\big(x - \mu_k^x\big)\right). \tag{4}$$

To construct the form of the state equation as (1), the function (4) can be further written as

$$\dot{x} = \hat{f}(x) = \sum_{k=1}^{K} \omega_k(x)(\Lambda_k x + d_k) \tag{5}$$

where

$$\omega_k = \frac{p(k)p(x|k)}{\sum_{i=1}^{K} p(i)p(x|i)} \tag{6}$$

$$\Lambda_k = \Sigma_k^{\dot{x}x}\big(\Sigma_k^x\big)^{-1} \tag{7}$$

$$d_k = \mu_k^{\dot{x}} - \Lambda_k \mu_k^x. \tag{8}$$

It is clear that (5) is a nonlinear dynamical system with nonlinear weighting terms ω_k, which can represent a wide variety of motions.

B. Parameterized Policy Learning for the Dynamical System

The dynamical system-based control law method is highly robust for motor skills learning, because it is a time-variant dynamical system and globally converges to the goal points [7]. However, it cannot perform complex tasks with current parameters, such as via-point or obstacle-avoidance tasks.

To improve the policy parameters from the demonstrations, the evolution strategies learning algorithm is applied to the dynamical system in (5). In this section, a parameterized control policy for the dynamical system in (5) is defined as

$$\dot{x}_t = f(x_t, t) + \Phi(x_t) \cdot (\Theta + \varpi_t) \tag{9}$$

where Θ is the learning parameter; $\Phi(x_t)$ denotes the control matrix; and $\varpi_t \sim \mathcal{N}(0, \Sigma)$ denotes the Gaussian exploration noise.

Considering the special case of a 3-D dynamical system to model the policy parameter, the dynamical system of GMR in (5) is reformulated as

$$\Phi(x_t) = [\Phi_1(x_t), \Phi_2(x_t), \ldots, \Phi_K(x_t)] \tag{10}$$

$$\Phi_k(x_t) = \omega_k \begin{bmatrix} x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_1^t & x_2^t & x_3^t & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

$$\Theta = [\Theta_1, \Theta_2, \ldots, \Theta_K]^T \tag{12}$$

$$\Theta_k = \Big[\Lambda_k^{1,1}, \Lambda_k^{1,2}, \Lambda_k^{1,3}, \Lambda_k^{2,1}, \Lambda_k^{2,2}, \Lambda_k^{2,3} \\ \Lambda_k^{3,1}, \Lambda_k^{3,2}, \Lambda_k^{3,3}, d_k^1, d_k^2, d_k^3\Big]. \tag{13}$$

From the definition in (10)–(13), the learning parameter Θ includes the matrices Λ_k and d_k of the GMR, which is given in the dynamical system in (5). It should be noted that the parameters μ_x^k and Σ_k^x are the components of weighted ω_k, which shows the nonlinear mapping relationship. To avoid nonlinear factors in the learning policy, the policy parameters only have $\mu_k^{\dot{x}}$ and $\Sigma_k^{\dot{x},x}$. Therefore, the learning goal is to find the optimization solution of the mean of the velocity $\mu_k^{\dot{x}}$ and covariance between position and velocity $\Sigma_k^{\dot{x},x}$ in the GMR. The mean and covariance of position are set as fixed parameters. The control matrix $\Phi(x_t)$ contains the nonlinear features of the policy. The features consist of the Gaussian basis functions multiplied by the input variables. The Gaussian basis functions are positioned in the input space during learning.

Generally, the policy parameters Θ are updated such that the cost function of motion trajectories $[x_{t_j}, x_{t_{j+1}}, \dots, x_{t_N}]_\Theta$ from the start time t_i can be minimized

$$\mathrm{S}(\Theta) = \phi\big(x_{t_N}\big) + \int_{t_i}^{t_N} \left(r_{t_i} + \frac{1}{2}\rho_t^T(\Theta)R\rho_t(\Theta)\right) \tag{14}$$

where $\phi(x_{t_N})$ is the terminal cost; r_{t_i} is the immediate cost; $\rho_t(\Theta) = \Phi(x_t)(\Theta + \varpi_t)$ denotes the control cost; and R is the positive constant weight matrix.

In the learning process, the evolution strategies learning method will be explored to minimize the control cost $(1/2)\rho_t^T(\Theta)R\rho_t(\Theta))$. Once optimum control has been obtained, the update item $\delta\Theta_{t_i}$ can be computed at each time step. To obtain a single update vector $\delta\Theta$, a time averaging method can then be used. The details of the update rule for the learning process will be presented in Section IV.

IV. Methodology for Policy Improvement

In this section, the NES algorithm is introduced for learning the parameters Θ in (12). The natural gradient of expected fitness is first deduced to update the policy parameters, and then the updating rule of exponential parameterization is presented.

A. Natural Evolution Strategies

NESs are based on search gradients for updating the policy parameters. The sampled gradient of the expected fitness function is treated as a search gradient. In this article, we aim to minimize the fitness/cost function, which is defined in (14). The search distribution is mainly considered as the multinormal Gaussian distribution with a full covariance matrix.

First, we define the mean and covariance of the search distribution as μ_Θ and Σ_Θ and the learning variable $\Theta = \{\mu_\Theta, \Sigma_\Theta\}$. To simplify the distribution expression, we define $CC^T = \Sigma_\Theta$, and then $\xi = \mu_\Theta + Cs$, where variable s is the standard normal distribution $s \sim \mathcal{N}(0, I)$. The sample ξ is then: $\xi \sim \mathcal{N}(\mu_\Theta, \Sigma_\Theta)$. The search distribution of expected fitness function $\bar{S} = -S$ is expressed as

$$\mathbb{J}(\Theta) = \int \bar{S}(\xi)p(\xi|\Theta)d\xi$$
$$p(\xi|\Theta) = \frac{1}{(2\pi)^{m/2}\det(C)}\exp\left(-\frac{1}{2}\left\|C^{-1}(\xi - \mu_\Theta)\right\|^2\right). \tag{15}$$

The gradient of $\mathbb{J}(\Theta)$ is obtained according to the log-likelihood trick

$$\begin{aligned}\nabla_\Theta \mathbb{J}(\Theta) &= \nabla_\Theta \int \bar{S}(\xi)p(\xi|\Theta)d\xi \\ &= \int \bar{S}(\xi)\nabla_\Theta p(\xi|\Theta)d\xi \\ &= \int \bar{S}(\xi)\nabla_\Theta p(\xi|\Theta)\frac{p(\xi|\Theta)}{p(\xi|\Theta)}d\xi \\ &= \int \bar{S}(\xi)\nabla_\Theta \log p(\xi|\Theta)p(\xi|\Theta)d\xi \\ &= \mathbb{E}\big[\bar{S}(\xi)\nabla_\Theta \log p(\xi|\Theta)\big]. \end{aligned} \tag{16}$$

According to the Monte Carlo estimation [31], the function in (16) can be approximated as

$$\nabla_\Theta \mathbb{J}(\Theta) \approx \frac{1}{l}\sum_{i=1}^{l} \bar{S}(\xi_i)\nabla_\Theta \log p(\xi_i|\Theta) \tag{17}$$

where the parameter l represents the population size. The gradient $\nabla_\Theta \mathbb{J}(\Theta)$ offers a search direction in parameter space. Finally, the update rule is written as

$$\Theta^{\text{new}} = \Theta + \eta\nabla_\Theta \mathbb{J}(\Theta) \tag{18}$$

where η is the learning rate parameter.

However, the traditional stochastic search gradient method in (18) is difficult to precisely determine the quadratic optimum. The natural gradient-based method is a good solution and it helps mitigate the slow convergence of the plain gradient in optimization landscapes with ridges and plateaus. Actually, the natural gradient is based on Riemannian geometry, and it learns the information from the manifold of probability distributions. The traditional gradient $\nabla_\Theta \mathbb{J}$ directly follows the steepest descent in the parameter space Θ in the distribution. For the maximum process of $\mathbb{J}$, it will generate a new distribution associated with updating the parameters from the hypersphere of radius c and center Θ, and thereby computing the Euclidean distance of two distributions. However, it creates a new problem that the updating relies on the particular parameterization of the distribution, where the gradients and updates follow the change in parameterization.

The natural gradient algorithm computes the natural distance $D(\Theta||\Theta + \delta\Theta)$ between $P(\xi|\Theta)$ and $P(\xi|\Theta + \delta\Theta)$ using the Kullback–Leibler (KL) divergence. The natural gradient of $\mathbb{J}$ can thus be reformulated as an optimization problem with a KL divergence constraint

$$\max \quad \mathbb{J}(\Theta + \delta\Theta) \approx \mathbb{J}(\Theta) + (\delta\Theta)^T\nabla_\Theta \mathbb{J} \tag{19}$$
$$s.t. \quad \mathrm{KL}(\Theta + \delta\Theta||\Theta) = c \tag{20}$$

where (19) is the Taylor expansion and (20) is the constraint of KL divergence approximated [32]; c is a small increment size. The KL divergence is a measure of the distance of two probabilities. In this constraint, we want to hold new and old policies close in parameter space. The goal of optimization is to find the update direction with the largest ascent of the objective in the KL divergence.

Since the KL divergence can be approximated with second-order Taylor expands using Fisher information matrix, the function (20) is reformulated as

$$\begin{aligned}\mathrm{KL}(\Theta\|\Theta+\delta\Theta) &\approx \mathrm{KL}(\Theta\|\Theta)\\ &\quad+(\nabla_{\Theta+\delta\Theta}\mathrm{KL}(\Theta\|\Theta+\delta\Theta))^T\delta\Theta\\ &\quad+\frac{1}{2}(\delta\Theta)^T F\delta\Theta \qquad (21)\\ &= \mathrm{KL}(\Theta\|\Theta)-\mathbb{E}\big[\nabla_\Theta \log p(\xi|\Theta)\big]^T\delta\Theta\\ &\quad+\frac{1}{2}(\delta\Theta)^T F\delta\Theta\approx\frac{1}{2}(\delta\Theta)^T F\delta\Theta \qquad (22)\end{aligned}$$

with

$$\begin{aligned}F &= \int p(\xi|\Theta)\nabla_\Theta \log p(\xi|\Theta)(\nabla_\Theta \log p(\xi|\Theta))^T d\xi\\ &= \mathbb{E}\big[\nabla_\Theta \log p(\xi|\Theta)(\nabla_\Theta \log p(\xi|\Theta))^T\big]. \qquad (23)\end{aligned}$$

The Lagrange function of optimization in (19) and (20) is written as

$$\mathcal{L}(\delta\Theta,\beta)=\mathbb{J}(\Theta)+\nabla_\Theta\mathbb{J}^T(\Theta)\delta\Theta+\beta\left(\frac{1}{2}(c-\delta\Theta)^T F\delta\Theta\right) \qquad (24)$$

where F denotes the Fisher information matrix, and β is the Lagrange multiplier. From the saddle-point theorem, the optimal solution $\delta\Theta$ satisfies the following condition:

$$\frac{\partial\mathcal{L}}{\partial(\delta\Theta)}=\nabla\mathbb{J}(\Theta)-\beta F(\delta\Theta)=0. \qquad (25)$$

Then, the optimal solution is obtained as

$$\delta\Theta=\beta^{-1}F^{-1}\nabla\mathbb{J}(\Theta). \qquad (26)$$

Finally, when the Lagrange multiplier $\beta > 0$ is given, the direction of the natural gradient is written as

$$\tilde{\nabla}_\Theta\mathbb{J}(\Theta)=F^{-1}\nabla\mathbb{J}(\Theta). \qquad (27)$$

B. Fitness Shaping and Exponential Parameterization

NES adopts rank-based fitness functions (object functions) to keep the method invariant with monotonically increasing transformations of the fitness function. Here, we define the utility-weighted values $\upsilon = [\upsilon_1,\ldots,\upsilon_l]$ with sort ascending to transform the fitness of the population. Invariance against a large set of transformations of the fitness function and/or the underlying search space is a desirable property of evolution strategies. Rank-based fitness shaping makes the algorithm invariant under monotonic transformations of the fitness function, and the natural gradient is invariant under linear transformations of the search space [30]. Therefore, when the same linear transformation is applied to the search space and the initial search distribution, the natural gradient will also be transformed.

First, the population is sorted in descending order, which means that ξ_1 is the best and ξ_l is the worst individual. Then, we replace the fitness function as the utility values, and thus the estimation equation in (17) can be reformulated as

$$\nabla_\Theta\mathbb{J}(\Theta)=\sum_{i=1}^{l}\upsilon_i\nabla_\Theta \log p(\xi_i|\Theta). \qquad (28)$$

C. Exponential NES Update Rule

In traditional CMA-ESs, the policy parameters follow the gradient step of covariance matrix $\delta\Sigma_\Theta$, so the new covariance matrix $\Sigma_\Theta + \delta\Sigma_\Theta$ should hold the positive-definite matrix property. However, since the gradient $\delta\Sigma_\Theta$ can be any symmetric matrix, we cannot guarantee this property. To address this issue, the covariance matrix is represented with an exponential map for symmetric matrices. We first give the exponential map function as

$$\mathcal{S}_m := \{M\in\mathbb{R}^{m\times m}|M^T=M\} \qquad (29)$$

and

$$\mathcal{P}_m := \{M\in\mathcal{S}_m|u^T Mu>0 \text{ for all } u\in\mathbb{R}^m\backslash\{0\}\} \qquad (30)$$

where $\mathcal{S}_m$ is the symmetric vector space, and $\mathcal{P}_m$ is the manifold of a symmetric positive-definite matrix. The exponential map is defined as

$$\exp\ :\mathcal{S}_m\to\mathcal{P}_m,\quad M\to\sum_{n=0}^{\infty}\frac{M^n}{n!}. \qquad (31)$$

The map is a diffeomorphism, and "exp" and its inverse operation $\log : \mathcal{P}_m \to \mathcal{S}_m$ are continuous. The covariance matrix Σ_Θ can be represented as $\exp(\zeta)$ by exp map $\mathcal{P}_m \to \mathcal{S}_m$. The properties of the gradient update are given in Remark 1.

Remark 1: The updating of exp map has the following properties: $\Sigma_\Theta + \delta\Sigma_\Theta$ is the valid covariance matrix due to the vector space updating of $\mathcal{S}_m$; the gradient step is invariant with respect to linear transformation, due to the updating of the ζ of exp operation.

To reduce the burden of computation of the Fisher matrix, we do not directly compute the global coordinate $\Sigma_\Theta = \exp(\zeta)$, and replace it with a linear transformation to another coordinate system that the current search distribution is the standard normal distribution with zero mean and unit covariance. We first define the current search distribution as $(\mu_\Theta, C) \in \mathbb{R}^m \times \mathcal{P}_m$ and satisfy $CC^T = \Sigma_\Theta$. In the tangent space, we define the updated search distribution as

$$(\delta,M)\mapsto\left(\mu_\Theta^{\text{new}},C^{\text{new}}\right)=\left(\mu_\Theta+C\delta,C\exp\left(\frac{1}{2}M\right)\right). \qquad (32)$$

The coordinate frame is natural, because the Fisher matrix with respect to an orthonormal basis of (δ, M) is the identity matrix. Hence, the distribution $\mathcal{N}(\mu_\Theta, CC^T)$ is converted into $(\delta, M) = (0, 0)$. The trick of obtaining updates in the natural coordinate frame is an option of exponential parameterization, which is used to keep the algorithm invariant under the linear transformation in the searching space. The log density mapping is written in the new coordinate system

$$\begin{aligned}\log(p(\xi|\delta,M)) &= -\frac{m}{2}\log(2\pi)-\mathrm{tr}(C)\\ &\quad-\frac{1}{2}\left\|\exp(-1/2M)C^{-1}(\xi-\mu_\Theta)\right\|^2. \qquad (33)\end{aligned}$$

Considering the population $\xi_i = \mu_\Theta + C \cdot s_i$, where $s_i \sim \mathcal{N}(0, I)$, the gradient is formulated as

$$\begin{aligned}\nabla_\delta \mathbb{J} &= \sum_{i=1}^{l} \upsilon_i \cdot \nabla_\delta|_{\delta=0} \log(p(\xi_i|M=0,\delta)) \\ &= \sum_{i=1}^{l} \upsilon_i \cdot \nabla_\delta|_{\delta=0}\left[-\frac{1}{2}\left\|C^{-1}\cdot(\xi_i-(\mu_\Theta+\delta))\right\|^2\right] \\ &= \sum_{i=1}^{l} \upsilon_i \cdot \nabla_\delta|_{\delta=0} C^{-1}\cdot(\xi_i-(\mu_\Theta+\delta)) \\ &= \sum_{i=1}^{l} \upsilon_i \cdot s_i. \end{aligned} \tag{34}$$

The gradient of M is

$$\begin{aligned}\nabla_M \mathbb{J} &= \sum_{i=1}^{l} \upsilon_i \cdot \nabla_M|_{M=0} \log(p(\xi_i|\delta=0,M)) \\ &= \sum_{i=1}^{l} \upsilon_i \cdot \nabla_M|_{M=0}\Big[-\mathrm{tr}(C) \\ &\qquad -\frac{1}{2}\left\|\exp(1/2M)C^{-1}\cdot(\xi_i-\mu_\Theta)\right\|^2\Big] \\ &= \sum_{i=1}^{l} \upsilon_i \cdot \Big[-I-\left(C^{-1}\cdot(\xi_i-\mu_\Theta)\right)\cdot \\ &\qquad (-1/2I)\cdot\left(C^{-1}\cdot(\xi_i-\mu_\Theta)\right)^T\Big] \\ &= \frac{1}{2}\sum_{i=1}^{l} \upsilon_i \cdot \left(s_i s_i^T - I\right). \end{aligned} \tag{35}$$

From the gradient update of $\nabla_\delta \mathbb{J}$ in (34), it can be seen that μ_Θ depicts the center updating of search distribution. Similarly, σ describes the updating of step size of $\nabla\sigma\mathbb{J}$ as

$$\nabla\sigma\mathbb{J} = \frac{\mathrm{tr}(\nabla_M \mathbb{J})}{m} \tag{36}$$

$$\nabla_B \mathbb{J} = \nabla_M \mathbb{J} - \nabla_\sigma \mathbb{J}. \tag{37}$$

Finally, the updating rule with learning rates is given as

$$\begin{aligned}\mu_\Theta^{\text{new}} &= \mu_\Theta + \eta_\mu \cdot \nabla_\delta \mathbb{J} \\ &= \mu_\Theta + \eta_\mu \cdot \sum_{i=1}^{l} \upsilon_i s_i \end{aligned} \tag{38}$$

$$\begin{aligned}\sigma^{\text{new}} &= \sigma \cdot \exp(\eta_\delta/2 \cdot \nabla_\sigma \mathbb{J}) \\ &= \sigma \cdot \exp\left(\frac{\eta_\delta}{2}\cdot \mathrm{tr}\left(\sum_{i=1}^{l} \upsilon_i (s_i s_i^T - I)\right)\Big/ m\right) \end{aligned} \tag{39}$$

$$\begin{aligned}B^{\text{new}} &= B \cdot \exp\left(\frac{\eta_B}{2}\cdot \nabla_B \mathbb{J}\right) \\ &= B \cdot \exp\left(\frac{\eta_B}{2}\cdot\left(\sum_{i=1}^{l} \upsilon_i (s_i s_i^T - I)\right.\right. \\ &\qquad \left.\left. -\frac{1}{m}\,\mathrm{tr}\left(\sum_{i=1}^{l} \upsilon_i (s_i s_i^T - I)\right)\cdot I\right)\right). \end{aligned} \tag{40}$$

V. Learning Variable Impedance Control of Stable Dynamical System

In this section, we derive the stability conditions of the dynamical system using the exponential NES to learn the stiffness of variable impedance control. To guarantee global stability, the exploration noise of exponential NES is subjected to the stability conditions of the dynamical system.

A. Shaping With Stable Exploration

For imitation learning, the dynamical system in (5) can generate the trajectories according to the reference skills. Generally, an unstable dynamical model will cause unexpected motion, such as deviation from the goal position. Therefore, it is necessary to determine the optimal parameters to regulate the dynamical system.

Theorem 1: If the state trajectories are generated according to the dynamical system in (5), the dynamical model is globally asymptotically stable at the goal point x_g under the sufficient conditions

$$\begin{cases} d_k = -\Lambda_k x_g \\ \Lambda_k + (\Lambda_k)^T \prec 0 \;\; \forall k = 1,\ldots,K. \end{cases} \tag{41}$$

Proof: To obtain the stability conditions of the dynamical system in (5), we first define a candidate Lyapunov function as

$$V(x) = \frac{1}{2}\left(x-x_g\right)^T\left(x-x_g\right). \tag{42}$$

According to the Lyapunov stability theorem of function in (42), we should set the parameters that satisfy the following conditions:

$$\begin{cases} V(x) > 0 \;\; \forall x \in \mathbb{R}^d \backslash \{x_g\} \\ \dot{V}(x) < 0 \;\; \forall x \in \mathbb{R}^d \backslash \{x_g\} \\ V(x_g) = 0 \;\&\; \dot{V}(x_g) = 0. \end{cases} \tag{43}$$

Obviously, $V(x) > 0$ except for goal position $x = x_g$. The time derivative of $V(x)$ is written as

$$\begin{aligned}\frac{d(V)}{dt} &= \frac{dV}{dx}\frac{dx}{dt} \\ &= \frac{1}{2}\frac{d}{dx}\left(\left(x-x_g\right)^T\left(x-x_g\right)\right)\dot{x} \\ &= \left(x-x_g\right)^T\dot{x} \\ &= \left(x-x_g\right)^T\cdot\hat{f}(x) \\ &= \left(x-x_g\right)^T\sum_{k=1}^{K}\omega_k(x)(\Lambda_k x + d_k) \\ &= \left(x-x_g\right)^T\sum_{k=1}^{K}\omega_k(x)\left(\Lambda_k(x-x_g) + \Lambda_k x_g + d_k\right) \\ &= \sum_{k=1}^{K}\omega_k(x)\left(x-x_g\right)^T\Lambda_k\left(x-x_g\right). \end{aligned}$$

Since the parameter ω_k is positive, the condition $\dot{V}(x) < 0$ should be constrained with the condition $\Lambda_k + (\Lambda_k)^T \prec 0$. It is then easy to conclude that the condition $V(x_g) = 0$ & $\dot{V}(x_g) = 0$ is satisfied. The proof of (41) is thus concluded. ■

We still need to obtain the parameters $\phi = \{\phi_1, \dots, \phi_K\}$ with the item $\phi_k = \{\lambda_k, \mu_k, \sum_k\}$ in (5). Common methods use the expectation maximization (EM) algorithm to determine the optimum parameters of the GMM. However, they cannot ensure global stability at the goal position, because they do not consider the constraint in (41) in the optimization. Here, the log-likelihood-based objective function of the optimization problem with constraints is defined as

$$\min \mathcal{F}(\phi) = -\frac{1}{T}\sum_{n=1}^{N}\sum_{t=0}^{T}\log P\big(x_{t,n}, \dot{x}_{t,n}|\phi\big)$$

$$s.t. \begin{cases} d_k = -\Lambda_k x_g \\ \Lambda_k + (\Lambda_k)^T \prec 0 \\ \Sigma_k \succ 0 \\ \sum_{k=1}^{K}\omega = 1, \omega_k \in (0,1). \end{cases} \tag{44}$$

The constraint is from the Lyapunov theory, where $d_k = -\Omega_k x_g$ represents the stable point; $\Lambda_k + (\Lambda_k)^T \prec 0$ denotes the negative-definite matrix; and $\Sigma_K > 0$ and $\sum_{k=1}^{K}\omega = 1, \omega_k \in (0,1)$ are the properties of covariance matrix and weights.

It should be noted that the parameter is added to the exploration noise in (9), which may become unstable due to random noise. To maintain the stability of the dynamical system in the learning process, we need to shape the exploration noise. The noise can be separated into ϖ_k^{Λ} and ϖ_k^{d}, which are added to parameters Λ and d, respectively, resulting in $d_{k,a} = d_k + \varpi_{k,a}^{d}$ and $\Lambda_{k,a} = \Lambda_k + \varpi_{k,a}^{\Lambda}$. The stable conditions of the dynamical system can then be rewritten as

$$d_{k,a} = -\Lambda_{k,a} x_g \tag{45}$$

$$\frac{\Lambda_{k,a} + \big(\Lambda_{k,a}\big)^T}{2} \prec 0$$
$$\forall k = 1, \dots K, \quad a = 1, \dots N_a. \tag{46}$$

The Λ_k matrix can be written as

$$\Lambda_k = \underbrace{\frac{\Lambda_k + (\Lambda_k)^T}{2}}_{(a)} + \underbrace{\frac{\Lambda_k - (\Lambda_k)^T}{2}}_{(b)}. \tag{47}$$

Actually, item (a) in (47) is the symmetric component and (b) is the skew-symmetric matrix, and it can be easily determined that the quadratic form of (b) is 0. Hence, we just need to keep the negative definite of (a). Similarly, the noise of $\varpi_{\Lambda}^{k,a}$ can be constructed as the sum of the skew symmetric and the symmetric noise matrix

$$\varpi_{k,a}^{\Lambda} = \varpi_{k,a}^{\text{skew}} + \varpi_{k,a}^{\text{sym}}. \tag{48}$$

So, we only need to design the symmetric matrix $\varpi_{k,a}^{\text{sym}}$ in the learning process. We sample the Gaussian noise, and obtain the symmetric matrix $(\varpi_{k,a}^{\text{sym}})'$ and construct the matrix $(\Lambda_k + (\Lambda_k)^T)/2 + \eta_{\varpi}(\varpi_{k,a}^{sym})'$, where the parameter η_{ϖ} is decreasing from 1 to 0 until it is negative definite. For condition (45), the exploration noise should satisfy the following:

$$\varpi_{k,a}^{d} = \varpi_{k,a}^{\Lambda} x_g. \tag{49}$$

After obtaining the optimal parameters from (44), the parameters can always preserve the conditions (45) and (46) when the noise is constrained.

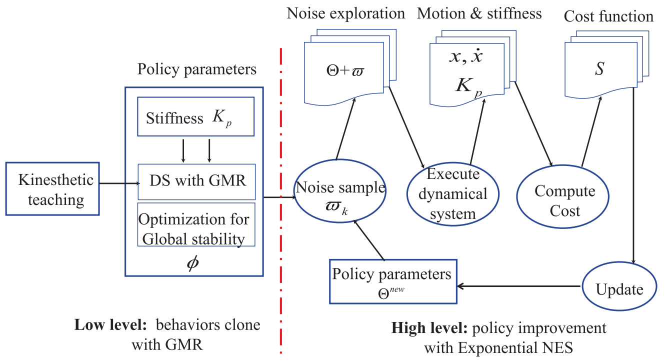

Fig. 2. Learning scheme of the dynamical system using exponential NES.

B. Variable Impedance Control Learning Policy

For the dynamical system, the variable impedance controller is written as

$$U = -K_P(x - x_d) - K_D(\dot{x} - \dot{x}_d) + U_f \tag{50}$$

$$K_D = \text{diag}\Big(2\sqrt{K_P}\Big) \tag{51}$$

where U_f is the feedback feedforward signal; K_P and K_D are the positive-definite matrix which represents the stiffness and damping matrices, respectively; and U is the control input. For Cartesian space tracking tasks, x denotes the trajectory in the task space, giving us the following relationship:

$$K_{P,q} = \mathcal{J}^T K_{P,C}\mathcal{J}, \;\; K_{D,q} = \mathcal{J}^T K_{D,C}\mathcal{J} \tag{52}$$

where $\mathcal{J}$ is the Jacobian matrix of the robot; and $K_{P,q}$ and $K_{P,C}$ denote the joint space and the Cartesian space stiffness matrices, respectively. Similarly, $K_{D,q}$ and $K_{D,C}$ denote the damping matrices in joint space and Cartesian space, respectively.

According to the relationship in (51), we just need to learn the stiffness parameters. The parameterized policy of stiffness learning associated with a 3-D dynamical system can be reformulated as

$$\Phi(x_t) = [\Phi_1(x_t), \Phi_2(x_t), \dots, \Phi_K(x_t)] \tag{53}$$

$$\Phi_k(x_t) = \omega_k \begin{bmatrix} x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_1^t & x_2^t & x_3^t & 0 & 0 & 0 & 1 \end{bmatrix} \tag{54}$$

$$\Theta = [\Theta_1, \dots, \Theta_k, \dots, \Theta_K]^T \tag{55}$$

$$\Theta_k = \Big[\Lambda_k^{1,1}, \Lambda_k^{1,2}, \Lambda_k^{1,3}, \Lambda_k^{2,1}, \Lambda_k^{2,2}, \Lambda_k^{2,3}, \Lambda_k^{3,1}, \Lambda_k^{3,2}, \Lambda_k^{3,3}, \Lambda_k^{4,1}, \Lambda_k^{4,2}, \Lambda_k^{4,3}, d_k^1, d_k^2, d_k^3, d_k^4\Big] \tag{56}$$

where $\Lambda_k^{4,1}$, $\Lambda_k^{4,2}$, $\Lambda_k^{4,3}$, and d_k^4 coincide with the stiffness parameters.

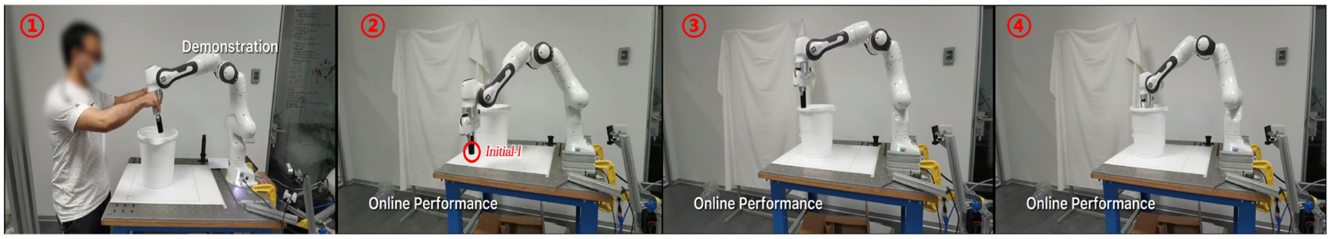

Fig. 3. Physical experiments are conducted using Franka Emika robot, where ① is the expert data from human demonstrations; and ②–④ are the different running phases at the final policy.

To fit the stiffness parameters into the learning process, we encode the K_P with the auxiliary dynamical system. Motivated by [16], the dynamical system of stiffness K_P is defined as

$$\dot{K}_{P,j} = \alpha_{k_p}\left(\left(g_x^j\right)^T\left(\Theta^{K_{P,j}} + \varpi^{K_{P,j}}\right) - K_{P,j}\right) \tag{57}$$

$$\Theta^{K_{P,j}} = \left[\Lambda_j^{4,1}, \Lambda_j^{4,2}, \Lambda_j^{4,3}, d_j^4\right]^T \tag{58}$$

$$g_x^j = \omega_k^j \cdot \left[x_1^d, x_2^d, x_3^d, 1\right]^T \tag{59}$$

where j is the index of the case of 3-D dimension space; $\varpi^{K_{P,j}}$ is the noise added to the dynamical equation of stiffness; the parameter α_{k_p} is a large positive constant scalar so that causes the time derivative of K_P to converge to zero quickly, thus enabling (57) to be rewritten as

$$K_{P,j} = \left(g_x^j\right)^T\left(\Theta^{K_{P,j}} + \varpi^{K_{P,j}}\right). \tag{60}$$

Therefore, the required stiffness can be obtained by learning the parameter $\Theta^{K_P} = [\Theta^{K_{P,1}}, \Theta^{K_{P,2}}, \ldots]$. The learning scheme is shown in Fig. 2.

Finally, for learning stiffness tasks, the details of the cost function in (14) are given as

$$\begin{aligned}
\phi(x_{t_N}) &= \left\|x_g - x_{t_N}\right\|^2 + \sum_{\substack{n=1\\ j=1:t_N}}^{N_{via}} \min \left\|x_n^{\text{via}} - x_{t_j}\right\|^2 \\
r_{t_j} &= \alpha_1\left\|\ddot{x}_{t_j}\right\| + \alpha_2\left\|K_{P,t_j}\right\| + R\cdot\frac{1}{2}\rho_t^T(\Theta)R\rho_t(\Theta) \\
&= \alpha_1\left\|\dot{x}_{t_j} - \dot{x}_{t_{j-1}}\right\| + \alpha_2\left\|K_{P,t_j}\right\| \\
&\quad + R\cdot\frac{1}{2}\rho_t^T(\Theta)R\rho_t(\Theta) \\
R &= \alpha_3 \cdot I
\end{aligned} \tag{61}$$

where α_1, α_2, and α_3 are the weights of the various components of the cost function, which is designed for users according to the tasks; N_{via} is the number of via-points, that is, the set points through which the dynamical system should pass in the learning tasks. In the special case of trajectory learning, we want the robot to go through points that did not appear in the primary experience. Finally, the details of the NES learning dynamical system are given as Algorithm 1.

VI. Experimental Demonstration

In the following, we describe the application of the proposed algorithm in imitation learning. Two scenarios are demonstrated: the first case is motion learning with an obstacle; the second is stiffness learning of a variable impedance controller.

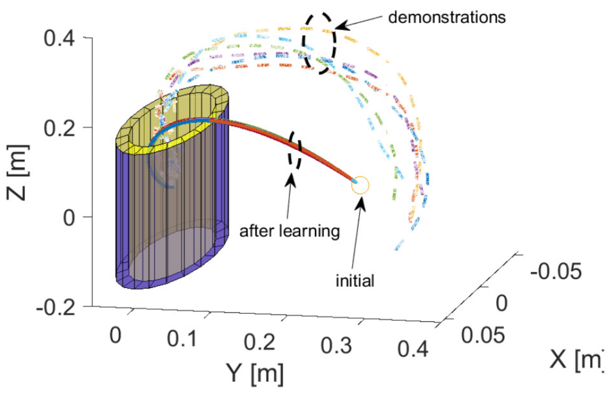

Fig. 4. Robot is controlled to repeat the motion six times from human demonstration. The initial position of the reproduction is selected randomly and learned by policy improvement.

A. Motion Learning With Obstacle Avoidance

The robot is first driven to perform the sampling tasks by human demonstration and executes the final learning results online as shown in Fig. 3. In our case, the robot executes a reaching motion into a bucket and collects the data from five repetitions, as shown in Fig. 4. Then, the first learning strategy associated with the GMR-based nonlinear dynamical system is used to clone the motor skills. To test adaptability and stability, we randomly set the initial position and reproduce the motor behavior with the condition constraints of the nonlinear dynamical system. The cost function is detailed

$$\begin{aligned}
\phi(x_{t_N}) &= \left\|x_g - x_{t_N}\right\|^2 \\
r_{t_j} &= \alpha_1\left\|\dot{x}_{t_j} - \dot{x}_{t_{j-1}}\right\| + \alpha_2\left\|\dot{x}_{t_j}\right\| \\
&\quad + \alpha_3 * I * \frac{1}{2}\rho_t^T(\Theta)R\rho_t(\Theta) \\
\alpha_1 &= 0.00001, \alpha_2 = 0.1,\ \ \alpha_3 = 0.001.
\end{aligned} \tag{62}$$

The iteration number is 200, and the number of roll-outs at each iteration is 10. When the robot collides with the bucket, the motion will stop and step into the next roll-out. The learning process is shown in Fig. 5. It can be seen that the robot reaches the goal position without collision and obtains the shortest possible path under the condition constraints in (44).

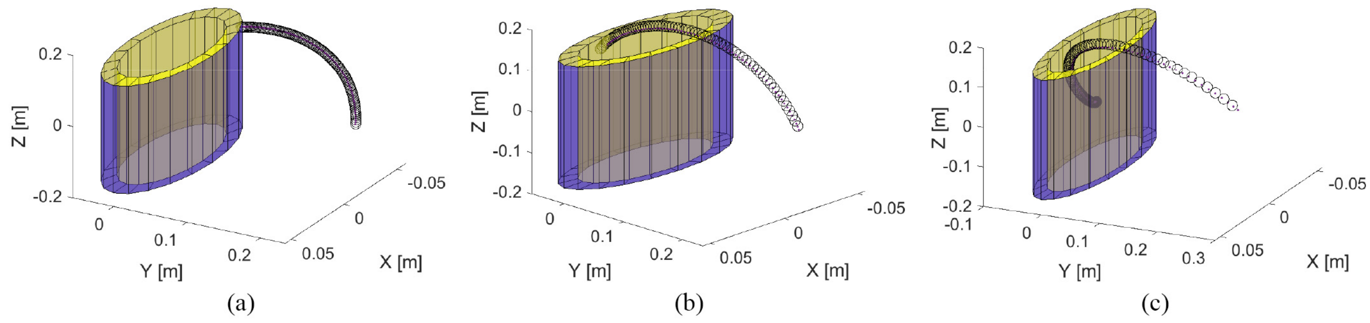

Fig. 5. Learning process of the dynamical system for obstacle avoidance. The first roll-outs collide with the bucket, the 300 roll-outs reach into the bucket but still collide with the bucket, and the 2000 roll-outs reach the goal without collision and obtain the shortest possible path under the condition of (44). (a) 1 roll-outs. (b) 300 roll-outs. (c) 2000 roll-outs.

Algorithm 1: Learning With Exponential NES

```
Input: Dataset $\{x_{t,n}, \dot{x}_{t,n}\}$, $\eta_\sigma$, $\eta_\delta$, $\eta_B$, $m$, $\mu_\Theta^{init}$,
    $\Sigma_\Theta^{init} = C^T C$
$\sigma \leftarrow \sqrt[m]{|\det(C)|}$
$B \leftarrow C/\sigma$
Low-level Learning
    Regression with GMR from human demonstration
    and get parameter $\phi_k = \{\lambda_k, \mu_k, \sum_k\}$ ;
    Obtain the optimal parameter $\phi = \{\phi_1, \cdots, \phi_K\}$ by
    optimization in (44);
end;
High-level Learning
    while until stopping criterion do
        for i=1 ··· l do
            Sampling: $\varpi_i \sim \mathcal{N}(0, I)$; /* To
                guarantee the stability of
                dynamical system, we shape
                the noise as in (48) (49)  */
            $\xi_i \leftarrow \mu_\Theta + \sigma B^T \varpi_i$;
            Evaluation of the cost function $S(\xi_i)$;
        end
        Sort $\{\varpi_i, \xi_i\}$ with respect to $\bar{S}(\xi_i)$;
        Compute gradients:
        $\nabla_\delta \mathbb{J} \leftarrow \sum_{i=1}^{l} \upsilon_i \cdot \varpi_i$;
        $\nabla_M \mathbb{J} \leftarrow \sum_{i=1}^{l} \upsilon_i \cdot (\varpi_i \varpi_i^T - I)$;
        $\nabla_\sigma \mathbb{J} \leftarrow \mathrm{tr}(\nabla_M \mathbb{J})/m$;
        $\nabla_B \mathbb{J} \leftarrow \nabla_M \mathbb{J} - \nabla_\sigma \mathbb{J} \cdot I$;
        Update mean: $\mu_\Theta \leftarrow \mu_\Theta + \eta_\delta \cdot \sigma B \cdot \nabla_\delta \mathbb{J}$;
        Update covariance matrix:
        $\sigma \leftarrow \sigma \cdot \exp(\eta_\sigma/2 \cdot \nabla_\sigma \mathbb{J})$;
        $B \leftarrow B \cdot \exp(\eta_B/2 \cdot \nabla_B \mathbb{J})$;
    end
end;
```

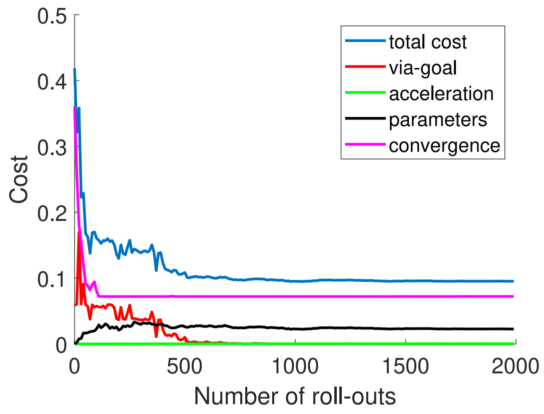

Fig. 6. Obstacle-avoidance case: learning process of cost value.

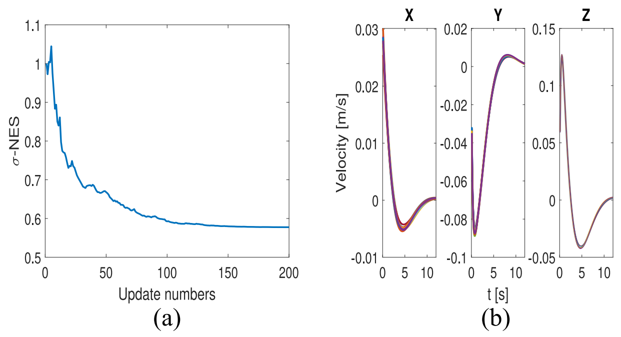

Fig. 7. Obstacle-avoidance case: σ and velocity. (a) Learning process of σ along the update numbers. (b) Velocity of 10 samples at final iteration.

Fig. 6 depicts the learning results of the cost value, with all items, including total cost, via-goal cost, acceleration cost, control cost of parameters Θ, and convergence cost of velocity. It can be seen that all the cost items converge to a small stable value and achieved the optimization results. Fig. 7(a) describes the learning results of the parameter σ in (39). Fig. 7(b) shows the velocity trajectories at the final policy. Moreover, due to the dynamical system constraints, the motion can globally converge to the goal position. Consequently, the learning strategy can limit the state of the dynamical system under the condition constraints, which is important in a real system considering the safe state. It should be noted that the learning method can modify the exploration noise by updating the covariance matrix, which is a convenient way of shaping the noise in a large exploration space.

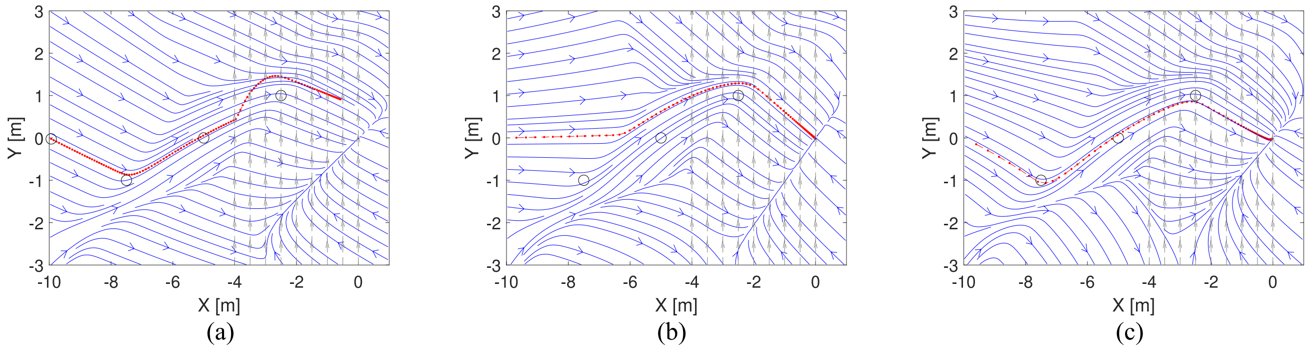

Fig. 8. Learning process of the dynamical system for stiffness learning. The first roll-outs deviate from the original trajectories with the disturbance, while the 200 roll-outs gradually converge to the original trajectories along with the iterations but still with a high error rate, and 1000 roll-outs coincide with the original trajectories and adaptation toward via-points under the condition of (44). (a) 1 roll-outs. (b) 200 roll-outs. (c) 1000 roll-outs.

B. Learning Stiffness of Variable Impedance Control

In this case, the robot should pass through the via-points, and the variable impedance controller is explored to control the robot with the external disturbance. The external force field is set as $F_{\text{ext}} = [0, 8]^T$, which means the force is only applied to the y-axis. The cost function is detailed

$$\begin{aligned} \phi(x_{t_N}) &= \left\|x_g - x_{t_N}\right\|^2 + \sum_{\substack{n=1\\ j=1:t_N}}^{N_{\text{via}}} \min \left\|x_n^{\text{via}} - x_{t_j}\right\|^2 \\ r_{t_j} &= \alpha_1 \left\|\dot{x}_{t_j} - \dot{x}_{t_{j-1}}\right\| + \alpha_2 \left\|K_p\right\| \\ &\quad + \alpha_3 * I * \frac{1}{2}\rho_t^T(\Theta) R \rho_t(\Theta) \\ \alpha_1 &= 0.001, \alpha_2 = 0.0002, \alpha_3 = 0.000001. \end{aligned} \tag{63}$$

The learning process is shown in Fig. 8, where the trajectory deviates from the original trajectory in the first roll-outs due to the force field (gray dotted line) and gradually converges to the original trajectory, passing through the desired via-points (4 via-points and goal point) along with the iterations after 200 roll-outs, and slightly deviating from the original trajectory and passing more precisely through the via-points after 1000 roll-outs. Fig. 9 shows the learning process cost value, where all cost items, including total cost, via-points cost, acceleration cost, parameters cost, and stiffness cost. We obtain similar results as the obstacle-avoidance case that all the cost items converge quickly, especially the total cost decreasing rapidly and achieving the optimization policy. Specifically, the via-points cost, convergence cost, and total cost performance mean that the final policy can pass through a set of points with high accuracy under perturbation. Fig. 10(a) also converged reasonably quickly and reached a steady state after 80 iterations. The performance and convergence behavior of the policy suggests that the proposed policy is a good match for our case.

Figs. 10(b) and 11(a) show the position and velocity trajectories at the last iteration, respectively. It can be seen that the trajectories pass through the via-points at the final policy with external disturbance and globally converge to the goal position under the constraints of the dynamical system (shown in streamlines). The results of stiffness learning are shown in Fig. 11(b). At the beginning phase, the stiffness level increases

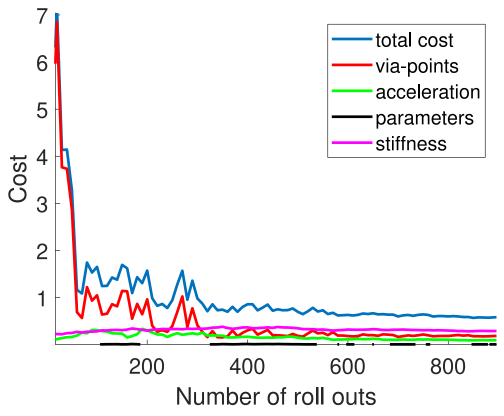

Fig. 9. Disturbance case: learning process of cost value.

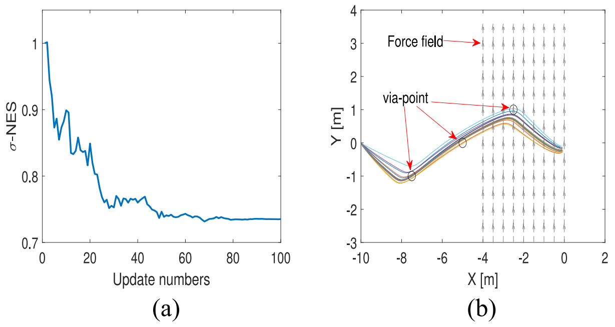

Fig. 10. Disturbance case: σ and trajectory. (a) Learning process of σ along the update numbers. (b) Learning trajectory at the final policy of variable impedance control.

to resist external perturbations and then decreases as the robot gradually approaches the target position.

C. Autonomous Motion Generation and Comparative Experiment

In this case, the robot is performed to grasp the object in terms of the new initial position and new goal position. Fig. 12 depicts the learning results of the dynamical system. Fig. 12(a) and (b) shows the robot can converge to the goal position from

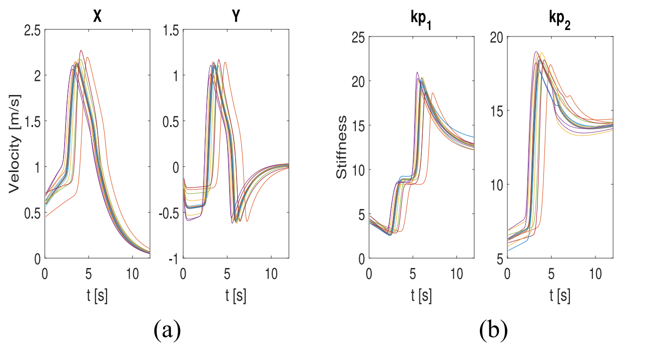

Fig. 11. Disturbance case: cost value and σ. (a) Velocity at the final policy. (b) Stiffness value at the final policy.

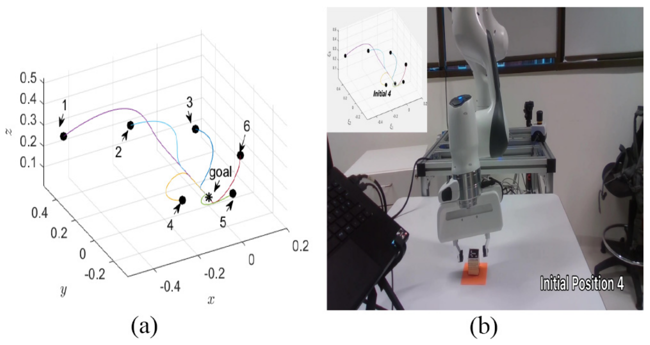

Fig. 12. Autonomous motion planning from the random initial position to the goal position. (a) Motion generation from the random initial position. (b) Demonstration with the robot.

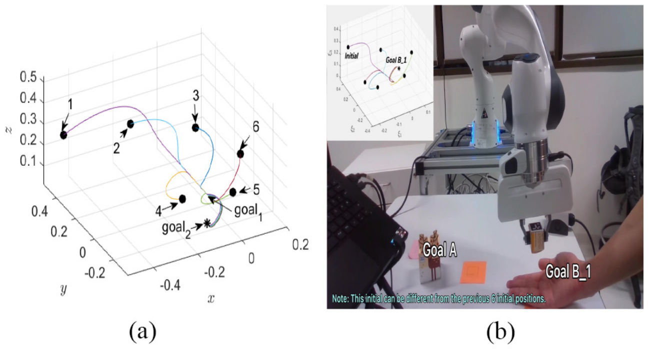
goal1
goal2
Goal A
Goal B_1
Note: This initial can be different from the previous 6 initial positions.
(a)
(b)

Fig. 13. Autonomous motion planning from goal$_A$ position to goal$_{B1}$ position. (a) Motion generation from the new starting position to the new goal position. (b) Demonstration with the robot.

the random initial position, which demonstrates the global convergence properties. Fig. 13 shows that the robot can adapt to the new goal position from the new initial position.

To demonstrate the proposed approach, we conduct a comparative experiment for the via-point task. From Fig. 14, it can be seen that our method has a smoother trajectory and smaller error than the traditional method in [21], and keep a better imitation shape at the final policy. From Fig. 15, we observe that our method converges after 1000 roll-outs (100 updates) and [21] converges after 2000 roll-outs (200 updates). Therefore, it can be concluded that our method has a faster convergence speed than [21].

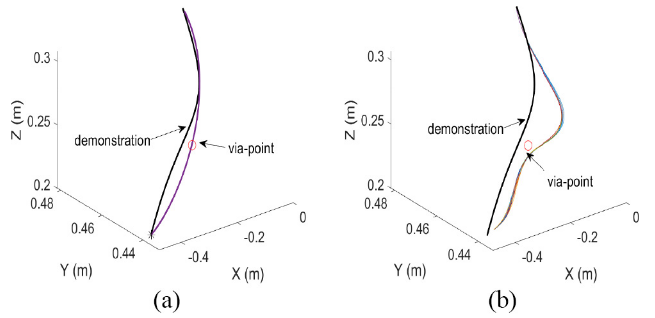

Fig. 14. Comparative experiment: trajectory after learning for via-point task at the final policy. (a) Trajectory learning with our method. (b) Trajectory learning with method in [21].

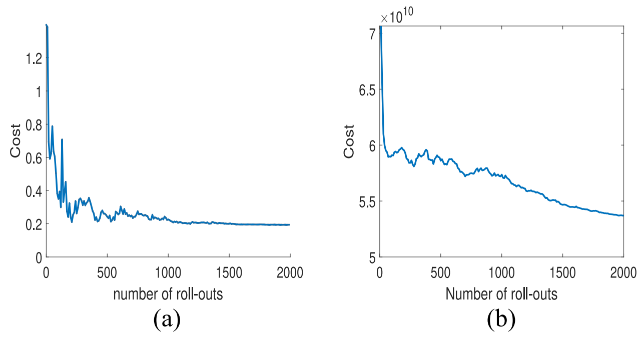

Fig. 15. Comparative experiment: cost value for via-point task at the final policy. (a) Total cost value with our method. (b) Total cost value with method in [21].

D. Discussion

With imitation learning, classical approaches based on behavioral cloning can imitate the motor skills, but lack adaptability and robustness. Based on Section VI-A, the robot can avoid the obstacle and follow the streamlines to match the expert trajectories of the dynamical system. Based on Section VI-B, we design the new cost function to counter the external disturbance and learn the variable impedance control. The experiment indicates that the proposed learning methods can improve the adaptability and robustness of the dynamical system. It is interesting to note that the cost function can be adjusted by users according to the task requirements. This means that the proposed method can be applied much more broadly than those presented in this article and only requires the design of a reasonable cost function. For instance, it will be interesting to explore a method for learning puncturing skills in surgery.

In addition, although exploration of a large policy space will benefit the learning performance in the virtual environment, the system may be unstable or become a singularity, leading to security issues. Indeed, the constraints of the dynamical system address the problem: the robot's trajectories follow the streamline, running in a safe state when exploring policy space.

VII. Conclusion

This article focuses on improving the adaptability and robustness of robot learning. We propose a policy improvement-based hierarchical learning strategy to imitate and motor skills from human demonstration. The low-level learning method only focuses on behavioral cloning, while the high-level one aims to enhance adaptability and robustness through policy improvement. To obtain the optimal policy parameters, the exponential NES method is presented for learning the parameters of the dynamical system. In the experiment section, we design two scenarios that are not covered by the expert data, with which to demonstrate the proposed methods. Our experiments indicate that our approach can successfully avoid obstacles and counter disturbances through learning stiffness. The first limitation is that the Gaussian component of GMM and length of demonstrations should not be too large, which will slow down the runtime. Another one is that the exploration noise should not be set too large in learning, because the robot should run in the workspace.

References

[1] M. Deng, Z. Li, Y. Kang, C. L. P. Chen, and X. Chu, "A learning-based hierarchical control scheme for an exoskeleton robot in human–robot cooperative manipulation," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 112–125, Jan. 2020.

[2] X. Wu, J. Liu, C. Huang, M. Su, and T. Xu, "3-D path following of helical microswimmers with an adaptive orientation compensation model," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 823–832, Apr. 2020.

[3] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *J. Mach. Learn. Res.*, vol. 22, pp. 1395–1476, Jan. 2021.

[4] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," 2021, *arXiv:2102.03861*.

[5] E. Pignat and S. Calinon, "Bayesian Gaussian mixture model for robotic policy imitation," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4452–4458, Oct. 2019.

[6] G. Chen, K. Chen, L. Zhang, L. Zhang, and A. Knoll, "VCANet: Vanishing-point-guided context-aware network for small road object detection," *Autom. Innov.*, vol. 4, pp. 400–412, Sep. 2021.

[7] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 943–957, Oct. 2011.

[8] S. M. Khansari-Zadeh and A. Billard, "Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robot. Auton. Syst.*, vol. 62, no. 6, pp. 752–765, 2014.

[9] Y.-H. Liu, Y. Liu, Y.-F. Liu, C.-Y. Su, Q. Zhou, and R. Lu, "Adaptive approximation-based tracking control for a class of unknown high-order nonlinear systems with unknown powers," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 4559–4573, Jun. 2022.

[10] X. Wu, D.-X. Liu, M. Liu, C. Chen, and H. Guo, "Individualized gait pattern generation for sharing lower limb exoskeleton robot," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 4, pp. 1459–1470, Oct. 2018.

[11] J. Duan, Y. Ou, J. Hu, Z. Wang, S. Jin, and C. Xu, "Fast and stable learning of dynamical systems based on extreme learning machine," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 6, pp. 1175–1185, Jun. 2019.

[12] S. Jin, Z. Wang, Y. Ou, and W. Feng, "Learning accurate and stable dynamical system under manifold immersion and submersion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3598–3610, Dec. 2019.

[13] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[14] F. Stulp, E. A. Theodorou, and S. Schaal, "Reinforcement learning with sequences of motion primitives for robust manipulation," *IEEE Trans. Robot.*, vol. 28, no. 6, pp. 1360–1370, Dec. 2012.

[15] A. Duan *et al.*, "Learning to sequence multiple tasks with competing constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2019, pp. 2672–2678.

[16] 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.

[17] L. Kong, W. He, C. Yang, Z. Li, and C. Sun, "Adaptive fuzzy control for coordinated multiple robots with constraint using impedance learning," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 3052–3063, Aug. 2019.

[18] 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.

[19] X. Yu *et al.*, "Bayesian estimation of human impedance and motion intention for human–robot collaboration," *IEEE Trans. Cybern.*, vol. 51, no. 4, pp. 1822–1834, Apr. 2021.

[20] F. Stulp and O. Sigaud, "Robot skill learning: From reinforcement learning to evolution strategies," *Paladyn J. Behav. Robot.*, vol. 4, no. 1, pp. 49–61, 2013.

[21] Y. Hu, X. Wu, P. Geng, and Z. Li, "Evolution strategies learning with variable impedance control for grasping under uncertainty," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 7788–7799, Oct. 2019.

[22] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi, "Bidirectional relation between CMA evolution strategies and natural evolution strategies," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2010, pp. 154–163.

[23] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 949–980, 2014.

[24] F. J. Abu-Dakka, L. Rozo, and D. G. Caldwell, "Force-based variable impedance learning for robotic manipulation," *Robot. Auton. Syst.*, vol. 109, pp. 156–167, Nov. 2018.

[25] N. Jaquier, D. Ginsbourger, and S. Calinon, "Learning from demonstration with model-based Gaussian process," in *Proc. Conf. Robot Learn.*, 2020, pp. 247–257.

[26] A. K. Tanwani and S. Calinon, "Learning robot manipulation tasks with task-parameterized semitied hidden semi-Markov model," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 235–242, Jan. 2016.

[27] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, Feb. 2013.

[28] Z. Li, T. Zhao, F. Chen, Y. Hu, C.-Y. Su, and T. Fukuda, "Reinforcement learning of manipulation and grasping using dynamical movement primitives for a humanoidlike mobile manipulator," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 1, pp. 121–131, Feb. 2018.

[29] S. Yi, D. Wierstra, T. Schaul, and J. Schmidhuber, "Stochastic search using the natural gradient," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 1161–1168.

[30] T. Glasmachers, T. Schaul, S. Yi, D. Wierstra, and J. Schmidhuber, "Exponential natural evolution strategies," in *Proc. 12th Annu. Conf. Genet. Evol. Comput.*, 2010, pp. 393–400.

[31] M. Y. Ata, "A convergence criterion for the Monte Carlo estimates," *Simul. Model. Pract. Theory*, vol. 15, no. 3, pp. 237–246, 2007.

[32] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 4, 2007, pp. 317–320.

Yingbai Hu (Student Member, IEEE) received the M.Sc. degree in control engineering from the South China University of Technology, Guangzhou, China, in 2017. He is currently pursuing the Ph.D. degree in computer science with the Technical University of Munich, Munich, Germany.

He is also a member of the Informatics 6-Chair of Robotics, Artificial Intelligence and Real-time Systems, Technical University of Munich. His main research interests include neural network optimization and control in robotics, robot learning, surgical robotics, and reinforcement learning.

Guang Chen received the M.Eng. degree from Hunan University, Changsha, China, in 2011, and the Ph.D. degree from the Technical University of Munich, Munich, Germany, in 2016.

He is a Research Professor with Tongji University, Shanghai, China, and a Senior Research Associate (guest) with the Technical University of Munich. His research interests include bioinspired vision, machine learning, and computer vision with applications in autonomous systems.

Dr. Chen was awarded the Program of Shanghai Rising Star 2021 and Shanghai S&T 35U35 2021. He serves as an associate editor for several international journals. He is the Program Chair of IEEE MFI 2022.

Zhijun Li (Fellow, IEEE) received the Ph.D. degree in mechatronics from Shanghai Jiao Tong University, Shanghai, China, in 2002.

From 2003 to 2005, he was a Post-doctoral Fellow with the Department of Mechanical Engineering and Intelligent systems, The University of Electro-Communications, Tokyo, Japan. From 2005 to 2006, he was a Research Fellow with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, and Nanyang Technological University, Singapore. He was the Vice Dean of the School of Information Science and Technology, University of Science and Technology, Hefei, China, in 2019, where he has been a Professor with the Department of Automation since 2017. His current research interests include wearable robotics, teleoperation systems, nonlinear control, and neural network optimization.

Prof. Li is serving as an Editor-at-Large for *Journal of Intelligent and Robotic Systems* and an associate editor for several IEEE TRANSACTIONS. Since 2016, he has been the Co-Chair of IEEE SMC Technical Committee on Bio-Mechatronics and Bio-Robotics Systems and IEEE RAS Technical Committee on Neuro-Robotics Systems.

Alois Knoll (Senior Member, IEEE) received the Diploma (M.Sc.) degree in electrical/communications engineering from the University of Stuttgart, Stuttgart, Germany, in 1985, and the Ph.D. degree (*summa cum laude*) in computer science from the Technical University of Berlin (TU Berlin), Berlin, Germany, in 1988.

He served on the faculty of the Computer Science Department, TU Berlin until 1993. He joined the University of Bielefeld, Bielefeld, Germany, as a Full Professor and the Director of the Research Group Technical Informatics until 2001. Since 2001, he has been a Professor with the Department of Informatics, Technical University of Munich (TUM), Munich, Germany. He was also on the board of directors of the Central Institute of Medical Technology, TUM. From 2004 to 2006, he was the Executive Director of the Institute of Computer Science, TUM. His research interests include cognitive, medical and sensor-based robotics, multiagent systems, data fusion, adaptive systems, multimedia information retrieval, model-driven development of embedded systems with applications to automotive software and electric transportation, as well as simulation systems for robotics and traffic.