Dynamic Modeling of Robotic Manipulator via an Augmented Deep Lagrangian Network

Shuangshuang Wu∗, Zhiming Li, Wenbai Chen, and Fuchun Sun

Abstract: Learning the accurate dynamics of robotic systems directly from the trajectory data is currently a prominent research focus. Recent physics-enforced networks, exemplified by Hamiltonian neural networks and Lagrangian neural networks, demonstrate proficiency in modeling ideal physical systems, but face limitations when applied to systems with uncertain non-conservative dynamics due to the inherent constraints of the conservation laws foundation. In this paper, we present a novel augmented deep Lagrangian network, which seamlessly integrates a deep Lagrangian network with a standard deep network. This fusion aims to effectively model uncertainties that surpass the limitations of conventional Lagrangian mechanics. The proposed network is applied to learn inverse dynamics model of two multi-degree manipulators including a 6-dof UR-5 robot and a 7-dof SARCOS manipulator under uncertainties. The experimental results clearly demonstrate that our approach exhibits superior modeling precision and enhanced physical credibility.

Key words: deep Lagrangian network; nonconservative dynamics; multi-degree manipulator; inverse dynamic modeling

1 Introduction

A robotic manipulator, as a mechanical device composed of joints, links, and actuators, is often a complex system with high nonlinearity and strong coupling. Discovering its governing equations is crucial in understanding, comprehending, and controlling the behavior occurring in robotic manipulator system. Traditional modeling methods for robotic manipulators typically involve manual mathematical and analytical techniques based on

fundamental principles from mechanics and physics, such as Newtonian, Lagrangian or Hamiltonian mechanics. For example, using Euler-Lagrange equation, the equations of motion for robotic manipulators with flexible joints[1], legged robots[2], and the robotic manipulator with a flying multirotor base[3] are obtained. The manual modeling methods rely on the accurate physical parameters and known structure of the robot which are not easy to get under certain conditions. The complex analytical process, high manual calculation load, and intricate physical parameters pose substantial challenges in achieving the accurate modeling for a robotic system.

With the advances of machine learning, data-driven methods has proven extremely useful in robotic system modeling. The representative methods include unstructured multi-layer perceptron (MLP)[4–6], locally weighted projection regression (LWPR), support vector regression (SVR), and Gaussian processes regression (GPR). However, they possess the same limitation: the lack of physical plausibility, model interpretability, and

● Shuangshuang Wu, Zhiming Li, and Wenbai Chen are with School of Automation, Beijing Information Science and Technology University, Beijing 100192, China. E-mail: ssw_0538_ysu@163.com;li_zhiming1022@163.com;chenwb@bistu.edu.cn.
● Fuchun Sun is with Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. E-mail: fcsun@mail.tsinghua.edu.cn.
∗ To whom correspondence should be addressed.
  Manuscript received: 2023-09-02; revised: 2023-12-04; accepted: 2024-01-09

generalization capability. Moreover, these methods require a high amount of dataset and often are constrained by computational inefficiency, the curse of dimensionality, and a proneness to overfitting.

In recent years, the physics-inspired learning methods are gradually being employed in robot dynamic modeling[7–13]. Reference [11] constructs deep Lagrangian networks (DeLaN) based on Lagrangian mechanics, which can effectively learn the system dynamics equations while ensuring their physical validity. The physics-inspired learning methods do not require specific knowledge of each system to obtain interpretable kinematic and dynamic models, and the learned model guarantee compliance with Lagrangian mechanics[12]. However, physics-inspired deep networks are often outperformed in simulating articulated rigidbodies without any uncertainty[13]. When it comes to real manipulator systems involving undeniable factors such as actuator and inter-joint friction, the modeling performance falls short compared to standard deep networks. Many studies have integrated friction in some specific model into the physics-inspired networks[12, 14–16]. However, the intricate occurrences of friction, hysteresis, and contact are often observed to happen simultaneously, posing challenges in describing them within certain specific models. As feedforward neural networks (FFNN) are able to well perform nonlinear regression, it can locally learn correlations of the torques and system state which are ignored by the network topology of DeLaN[11, 12].

Inspired by the above discussions, in this paper, by combining a DeLaN and a standard FFNN, a novel deep Lagrangian network called as DeLaN-FFNN to learn dynamics models is proposed and applied to model robot manipulator systems. This network can learn both the aspects of manipulator dynamics that adhere to Lagrangian mechanics and the uncertainties that deviate from Lagrangian mechanics. This network is an augmented DeLaN and can be widely applied to model a large class of the physical body system including multi-body robot, legged robot, robot car, and so on. In this paper, we apply the proposed network to model two multi-degree of robotic manipulator systems under unknown disturbances, and the simulation results have shown the validity and superiority.

The rest of this paper is organized as follows: In Section 2, we extensively review Lagrangian dynamics, exploring the integration of Lagrangian

priors with deep learning to produce DeLaN. In Section 3, we introduce the structure of integrated deep Lagrangian network, accompanied by a thorough explanation of its principles and functions. In Section 4, several experiments are conducted to confirm the effectiveness of our method in enhancing the accuracy of robotic arm dynamic modeling while maintaining model rationality.

2　Preliminary

Given that our work is conducted within the framework of DeLaN, it is necessary to provide a concise introduction to it. DeLaN achieves transparency through the integration of prior knowledge and the enforcement of physical plausibility. This transparency facilitates a clearer comprehension of the learned components within the function and how the approximation aligns with the actual model.

2.1　Lagrangian dynamics

The Lagrangian $\mathcal{L}$ in Lagrangian mechanics is a function of generalized coordinates q that describes the dynamics of a system. While the Lagrangian is not unique, any function that produces the correct equations of motion is considered valid. The Lagrangian is generally chosen to be

$$\mathcal{L}(q,\dot{q}) = T(q,\dot{q}) - V(q) = \frac{1}{2}\dot{q}^{\mathrm{T}}H(q)\dot{q} - V(q) \quad (1)$$

where $T(q,\dot{q})$ is the kinetic energy, $V(q)$ is the potential energy, and $H(q)$ is the symmetric and positive definite inertial matrix of the manipulator. By utilizing the calculus of variations, we get the Euler-Lagrange equation with non-conservative forces described by

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathcal{L}(q,\dot{q})}{\partial \dot{q}} - \frac{\partial \mathcal{L}(q,\dot{q})}{\partial q} = \tau \quad (2)$$

and further

$$\frac{\partial^2 \mathcal{L}(q,\dot{q})}{\partial^2 \dot{q}}\ddot{q} + \frac{\partial \mathcal{L}(q,\dot{q})}{\partial q \partial \dot{q}}\dot{q} - \frac{\partial \mathcal{L}(q,\dot{q})}{\partial q} = \tau \quad (3)$$

where τ is the generalized forces. Substituting $\mathcal{L}$ with the kinetic and potential energy in Eq. (1) yields the second order ordinary differential equation described by

$$H(q)\ddot{q} + \underbrace{\dot{H}(q)\dot{q} - \frac{1}{2}\left(\dot{q}^{\mathrm{T}}\frac{\partial H(q)}{\partial q}\dot{q}\right)^{\mathrm{T}}}_{=C(q,\dot{q})} + \underbrace{\frac{\partial V(q)}{\partial q}}_{=g(q)} = \tau \quad (4)$$

where $c(q,\dot{q})$ represents the centrifugal force and Coriolis force matrix, and $g(q)$ represents gravity vector. The Lagrangian method is a commonly used approach for inverse dynamics analysis, which allows for the establishment of the simplest form of inverse dynamics models.

2.2 Construction of the parametric network

Different deep networks have been proposed to learn the physical dynamics models. Figure 1 provides a set of comparative flowcharts for learning continuous-time inverse models using three distinct types of networks, including one standard network and two physics-inspired deep networks. One simple standard deep network flowchart is given as Fig. 1a. Apparently, this approach represents a more direct and raw form of learning directly from data, disregarding the influence of the underlying physical structure. By contrast, DeLaN is one kind of representative physics-inspired deep network, which integrates specific structures that conform to the Lagrangian mechanics to enhance physical rationality.

Figure 1b is the architecture of DeLaN-structured Lagrangian. It has two independent deep networks to parameterize kinetic and potential energy separately,

and then obtains the Lagrangian and inverse model. DeLaN-Black-Box Lagrangian[13] is another kind of DeLaN whose structure is shown as Fig. 1c, with only one single deep network directly parameterizes the Lagrangian. Note that the foundation of this work is the DeLaN-structured Lagrangian. For the sake of brevity, the term "DeLaN" is used afterwards in this paper to refer to the DeLaN-structured Lagrangian without confusion.

As shown in Fig. 1b, DeLaN approximates the mass matrix $H(q)$ and the potential energy $V(q)$ by utilizing separate feed-forward networks. For $V(q)$, a common deep network has been adopted, thus there is no need to elaborate further. Its approximation is taken as $\hat{V}(q,\phi)$, where ϕ is the respective network parameter. For $H(q)$, compared with the traditional model learning approaches which learns $H(q)$ from data directly, DeLaN exploits the matrix's symmetric positive definite property and gets $H(q)$ through a more rational and scientific approach. The approximated mass matrix $\hat{H}(q)$ can be decomposed as a product of a lower-triangular matrix $\hat{L}(q)$ with its transpose as follows:

$$\hat{H}(q;\theta) = \hat{L}(q;\theta)\hat{L}(q;\theta)^{\mathrm{T}} \tag{5}$$

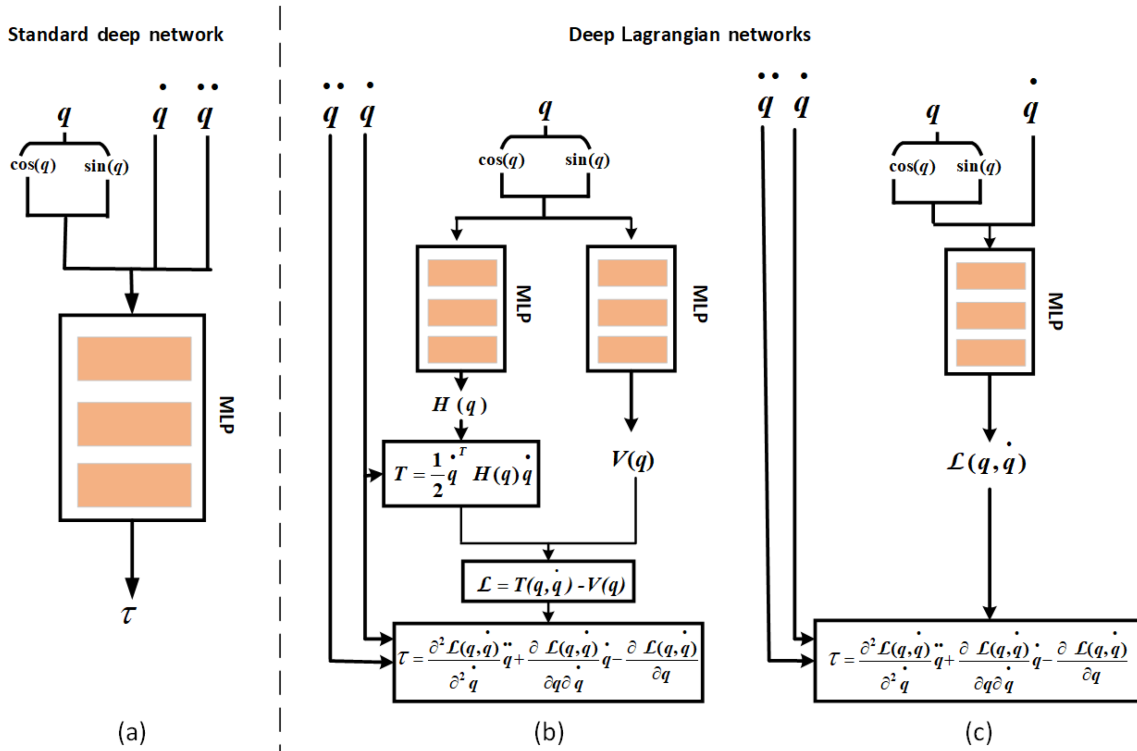where θ is the network parameters of mass matrix, and

Fig. 1 Three flowcharts for a continuous-time inverse model utilizing the deep networks, where (a) is a standard deep network, (b) is a DeLaN in the form of a structured Lagrangian, and (c) is a DeLaN in the form of a Black-Box Lagrangian.

$L(q;\theta)$ is the feedforward neural network parameterized by θ. The Lagrangian $\mathcal{L}$ can be approximated as

$$\hat{\mathcal{L}}(q,\dot{q};\theta,\phi) = \frac{1}{2}\dot{q}^{\mathrm{T}}\hat{H}(q;\theta)\dot{q} - \hat{V}(q;\phi) \tag{6}$$

where θ and ϕ are the respective network parameters. By employing this parametrization, the inverse model can be derived. Then the inverse model $\tau = f^{-1}(q, \dot{q}, \ddot{q}; \theta, \phi)$ is approximated by

$$\hat{f}^{-1}(q,\dot{q},\ddot{q};\theta,\phi) = \hat{H}(q;\theta)\ddot{q} + \dot{\hat{H}}(q;\theta)\dot{q} - \frac{1}{2}\left(\dot{q}^{\mathrm{T}}\frac{\partial \hat{H}(q;\theta)}{\partial q}\dot{q}\right) + \frac{\partial \hat{V}(q;\phi)}{\partial q} \tag{7}$$

The input of the network is position-related information of the generalized coordinates, velocity, and acceleration. The issue of angle wrapping at $\pm\pi$ poses challenges in function approximation if continuous revolute joints are without angular limits. Sine/cosine feature transformations are commonly employed as a mitigation strategy for this issue[13]. The network's output is the approximated torque τ as Eq. (7), which enables us to determine the torque of each joint from the joint position-related data corresponding to the desired trajectory. The optimization objective of the Lagrangian dynamics can be expressed as follows:

$$(\theta^*, \phi^*) = \arg\min_{\theta, \phi}\left\|\hat{f}^{-1}(q,\dot{q}, \ddot{q}; \theta, \phi) - \tau_R\right\|_{W_{\tau_R}}^2 \tag{8}$$

where τ_R represents the real torque collected from the physical manipulator, $\|\cdot\|_W$ represents the Mahalanobis norm, and W_{τ_R} represents the diagonal covariance matrix of the generalized forces. It is necessary to normalize the loss function using covariance matrix since the torque magnitude may vary greatly from joint

to joint.

It is worth noting that the optimization problem of Eq. (8) should be solved under the constraints that $\hat{H}$ is positive-definite and the corresponding derivatives can be well approximated. For this issue, Ref. [11] proposed an efficient network which can guarantee the positive-definite constraint for all parameters and analytical derivatives approximation. The detailed network structure are omitted here, please see Fig. 2[11]. By means of this network topology, DeLaN can learn the physical model trained with standard end-to-end optimization techniques.

3 An Augmented Deep Lagrangian Network: DeLaN-FFNN

In a traditional manual dynamics modeling method, the system identification approach and DeLaN typically rely on the Lagrangian equation and the exact knowledge of the kinematics. This promotes extrapolation, but neglects the impact of uncertain physical factors and dynamics on the modeling accuracy of mechanical systems[17]. Anything that is not modeled in physics prior cannot be learned, such as friction[18], hysteresis, and contact[16]. Then a solitary application of Lagrangian mechanics may prove insufficient in describing non-conservative dynamics[19]. We create an augmented deep Lagrangian network model by combining DeLaN that gives the main conservative dynamics model and a standard feedforward network that approximates the existing non-conservative forces. The simple feed-forward network can well learns the uncertainty that is not
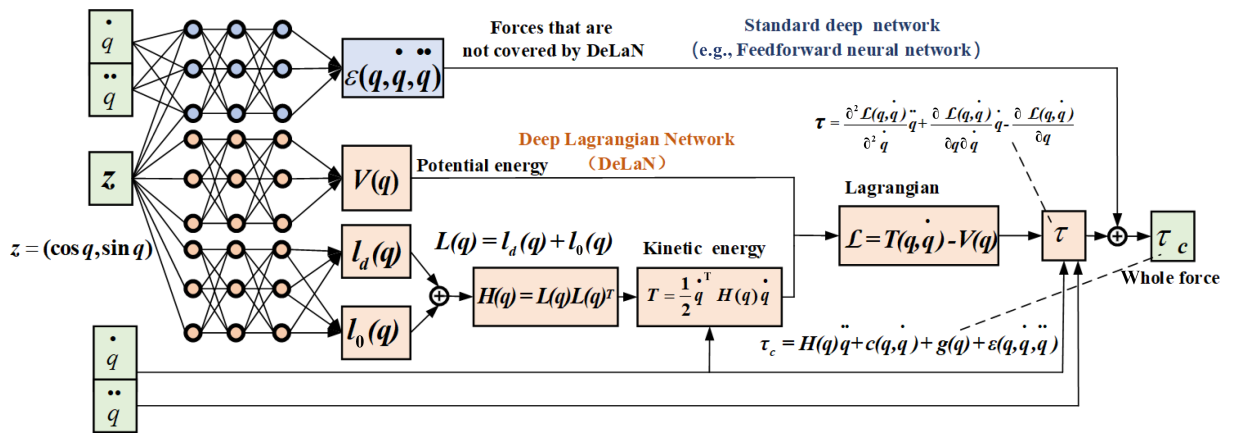

Fig. 2 **DeLaN-FFNN predicts dynamics by decomposing the forces into the forces that comply with Lagrangian mechanics and the forces that do not conform to Lagrangian mechanics. In the network model, the green represents the network's input and output, the orange portion represents DeLaN, and the blue portion represents the standard deep network. This structure of the dynamics network model better aligns with the actual conditions of the manipulator.**

considered by DeLaN, which enhances the modeling accuracy while maintaining the level of physical plausibility.

3.1 Manipulator dynamics model

DeLaNs are designed for continuous control applications and are specifically tailored to model rigid body dynamics[20]. However, in most cases, the robot manipulator system cannot be treated as an ideal rigid body. Building upon Lagrangian mechanics, we incorporate the error terms into the dynamics modeling, the inverse dynamics model of manipulator is described by Refs. [21, 22],

$$H(q)\ddot{q} + C(q, \dot{q}) + g(q) + \varepsilon(q, \dot{q}, \ddot{q}) = \tau_R \qquad (9)$$

where τ_R is the generalized forces, $\varepsilon(q, \dot{q}, \ddot{q})$ represents all the uncertainties that may appear in the manipulator system, mainly include the following factors:

(1) Friction forces: Friction in robot joints is a common nonlinear factor that introduces errors. Friction forces can be categorized into static friction and dynamic friction, and they can be modeled using various approaches, such as Coulomb friction model and viscous friction model[23, 24].

(2) Elastic forces: Robot components such as linkages, transmission systems, or other parts may exhibit elasticity. When a torque is applied, these components undergo elastic deformation, resulting in the generation of opposing forces[25].

(3) Nonlinear dynamic effects: In certain cases, manipulator systems may involve nonlinear dynamic effects, such as nonlinear bending, complex periodic motions, chaos, etc.[26]

(4) External disturbance forces: During operation, manipulators may experience external forces capable of disrupting their motion, such as wind forces, vibrations, or impacts from other objects.

(5) Sensor errors: In practical manipulator systems, sensors may exhibit noise, drift, or other sources of error. These errors can be reflected in the error terms of the inverse dynamics equations.

(6) Other model errors: Dynamics models are typically created using mathematical models to describe the behavior of the manipulator. However, these models often cannot fully accurately reflect the real-world conditions. Model errors can include errors caused by approximations or unmodeled dynamic characteristics, among other reasons[27].

Apparently, the priors of Lagrangian mechanics are unable to adequately describe the intricate nonlinear phenomena and other uncertainties mentioned above.

3.2 Construction of the parametric network

The structure of the DeLaN-FFNN is shown in Fig. 2. The portion highlighted in orange represents the structure of DeLaN, where the parameters include $H(q)$ and $V(q)$ are to be trained. The portion highlighted in blue represents the structure of FFNN, where and $\varepsilon(q, \dot{q}, \ddot{q})$ is to be trained. FFNN is a standard neural network without a model prior and applied to model the uncertainty term $\varepsilon(q, \dot{q}, \ddot{q})$. The approximated uncertainty term is named as $\hat{\varepsilon}(q, \dot{q}, \ddot{q}; \psi)$ where ψ are the corresponding network weight. Combining DeLaN referred as Eq. (7), FFNN referred as $\hat{\varepsilon}(q, \dot{q}, \ddot{q}; \psi)$, Eq. (9), and

$$\tau_c = \hat{f}^{-1}(q, \dot{q}, \ddot{q}; \theta, \phi, \psi) \qquad (10)$$

the approximated robotic manipulator' inverse model can be described by

$$\hat{f}^{-1}(q, \dot{q}, \ddot{q}; \theta, \phi, \psi) = \hat{H}(q; \theta)\ddot{q} + \hat{C}(q, \dot{q}; \theta) + \\ \hat{g}(q; \phi) + \hat{\varepsilon}(q, \dot{q}, \ddot{q}; \psi) \qquad (11)$$

We define our optimization objective as follows:

$$(\theta^*, \phi^*, \psi^*) = \underset{\theta, \phi, \psi}{\arg\min}\Big(\lambda\hat{\varepsilon}(q, \dot{q}, \ddot{q}; \psi)^2 + \\ \left\|\hat{f}^{-1}(q, \dot{q}, \ddot{q}; \theta, \phi, \psi) - \tau_R\right\|^2_{W_{\tau_R}}\Big) \qquad (12)$$

where λ is a positive value used to adjust the proportion of first loss term in the overall loss value, τ_R, $\|\cdot\|_W$, and W_{τ_R} are as defined for Eq. (8). It is worth noting that when we optimize this model, τ_c (or $\hat{f}^{-1}$) should recover ground truth τ_R, and we maximize the utilization of the output torque τ from DeLaN while minimizing $\varepsilon(q, \dot{q}, \ddot{q})$ as much as possible, specifically, to eliminate the effect of the term $\varepsilon(q, \dot{q}, \ddot{q})$ when τ_R does not contain any uncertainties.

4 Experiment

In the experiments, we employ DeLaN-FFNN, DeLaN, and FFNN models to acquire the inverse dynamics of both a simulated manipulator and a real manipulator. The objective is to test whether DeLaN-FFNN can provide more accurate modeling of the manipulator's dynamics and maintain physical plausibility. As that in Ref. [11], the performances of all the learned model are evaluated using the error on training and test trajectories with the mean squared error (MSE) as the indicator and compared to the other learned models.

The JAX deep learning framework is utilized to construct neural networks, with the automatic differentiation function employed for calculating partial derivatives within the dynamics model[28].

4.1   Simulated manipulator experiments

Simulation environment is built on the Pybullet engine[29]. We utilized the 6-dof UR-5 manipulator simulation model as the modeling object for the dynamic network model. The simulation of the UR-5 manipulator is depicted in Fig. 3a.

4.1.1   Ideal simulation manipulator

We first consider the ideal case which means all the uncertain factors do not exist in the simulated manipulator. The simulation is only based on rigid-body dynamics. We created a trajectory dataset of the UR-5 manipulator including the joint position data q, joint velocities $\dot{q}$ data, joint acceleration $\ddot{q}$, and the joint torque τ by running the manipulator randomly

and getting various multi-segment motions. This torque data is obtained using the inverse dynamics calculation function in Pybullet. The inverse dynamics are computed using the recursive Newton-Euler algorithm (RNEA)[30], which computes the feedforward torque using estimated physical properties of the system, i.e., the link dimensions, masses and moments of inertia. The training dataset consists of a total of 60 667 sets of joint states and joint torques $(q, \dot{q}, \ddot{q}, \tau)$. The following comparisons evaluating the the modeling accuracy of each dynamic network model are all based on this dataset. The simulation environment and three trajectory test examples (0−2) are depicted in Figs. 3b−3d.

In this experiment, for the optimization objective of DeLaN-FFNN, we set the weight parameter $\lambda = 0.0001$. The results of the inverse model are visualized in Fig. 4. Due to DeLaN's capability of learning the underlying physical model, on average, DeLaN
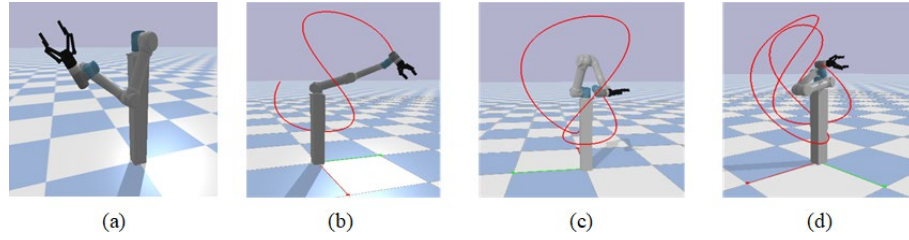

Fig. 3   (a) UR-5 manipulator in the simulation environment. (b−d) Complete trajectory of the end effector of the simulated manipulator in the three test sets are depicted as red lines in figures.
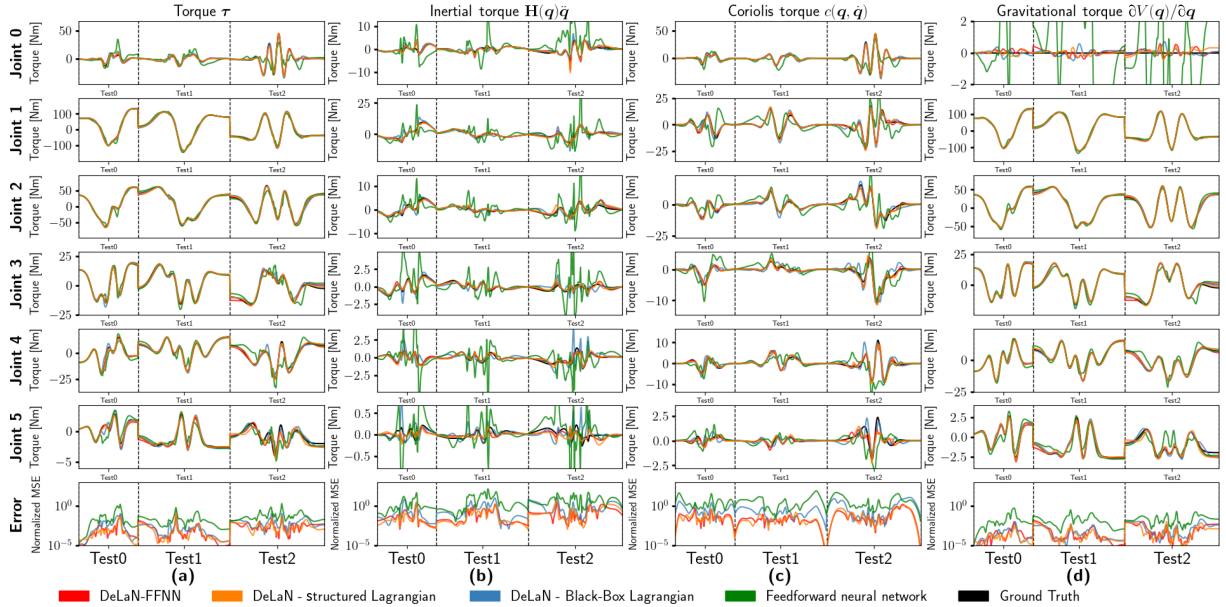

Fig. 4   (a) Learned inverse model using the entire training dataset consists of 60 667 samples. (b) Predicted force decomposition into the inertial force $H(q)\ddot{q}$, (c) the Coriolis and Centrifugal forces $c(q, \dot{q})$, and (d) Gravitational force $g(q)$. The last row represents the normalized MSE of the six joints on the data samples.

achieves a lower MSE compared to FFNN in predicting both the torque τ and the inertial force $H(q)\ddot{q}$, the Coriolis and centrifugal forces $c(q, \dot{q})$, and the gravitational force $g(q)$. Because FFNN is a standard neural network without a model prior, it is unable to learn the underlying physical model of the manipulator, resulting in poorer modeling performance. Among different DeLaN models, DeLaN-structured Lagrangian achieves better modeling performance compared to DeLaN-Black-Box Lagrangian. This is because DeLaN-structured Lagrangian incorporates deeper-level Lagrangian dynamics prior knowledge, allowing it to capture and represent the underlying system dynamics more effectively. Regarding DeLaN-FFNN, it achieves modeling performance that is very close to DeLaN-structured Lagrangian. Compared to DeLaN-Black-Box Lagrangian and FFNN, DeLaN-FFNN achieves a lower MSE. Therefore, it is reasonable to infer that when the actual torque τ_R does not contain uncertainties $\varepsilon(q, \dot{q}, \ddot{q})$ or when uncertainties $\varepsilon(q, \dot{q}, \ddot{q})$ is very small, DeLaN-FFNN is capable of eliminating or approximately eliminating the influence of uncertainties $\varepsilon(q, \dot{q}, \ddot{q})$.

4.1.2 Injecting damping terms into the simulated manipulator

In this experiment, we consider the case that the simulated manipulator may work under a uncertain environment. We configured linear damping, angular damping, and joint damping terms for the simulated manipulator using the changeDynamics function in Pybullet to simulate uncertainties $\varepsilon(q, \dot{q}, \ddot{q})$. Similar to the ideal case, we create a trajectory dataset of the UR-5 manipulator under this condition. We collected a total of 58 494 sets of joint states and joint torques $(q, \dot{q}, \ddot{q}, \tau)$ as the training dataset. The following tests are all based on this dataset.

Firstly, to see the influence of the weight parameter λ in the optimization of DeLaN-FFNN, a series of experiments are conducted by training the model under different values of λ. The result is shown in Fig. 5. When the weight parameter λ is large enough in DeLaN-FFNN, the test error becomes very similar to that of DeLaN-structured Lagrangian. From Fig. 5, we see that as the value of λ in DeLaN-FFNN gradually decreases, the test error will exhibit fluctuating changes. When λ becomes small enough, the test error of DeLaN-FFNN will start to decrease significantly. After reaching a certain point, further decreasing the value of λ will not lead to significant changes in the
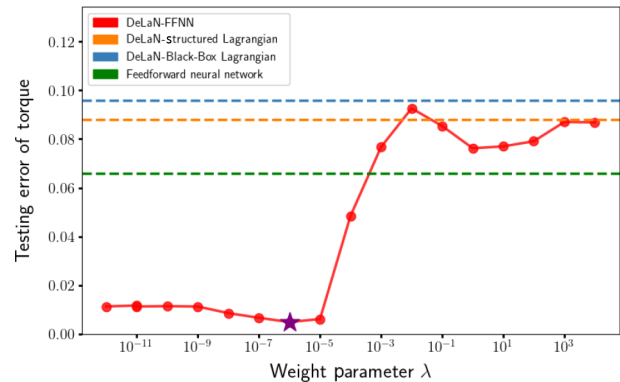

Fig. 5 Trend of test error of DeLaN-FFNN as the weight parameter λ in the optimization objective varies. The value of λ corresponding to the purple pentagon represents the point with the minimum testing error. The test error of other models are fix.

test error of DeLaN-FFNN. It indicates there exist a optimal value or an optimal range of λ for achieving the best learning performance using DeLaN-FFNN. Beyond this range, the impact of λ on improving the model's test error becomes less evident and may even lead to a decrease in modeling accuracy. Therefore, selecting an appropriate λ value during the model training process is crucial in achieving optimal performance.

Then the comparative experiments are performed by utilizing DeLaN-FFNN and the other three models to learn the inverse model the UR-5 manipulator. The results can be found in Figs. 6 and 7. From Fig. 7, we see that the modeling performance of DeLaN-structured Lagrangian and DeLaN-Black-Box Lagrangian on three test sets is quite similar. FFNN shows some improvement in modeling accuracy, but it still falls short of achieving a significantly better modeling performance. Apparently, DeLaN-FFNN outperforms the other three dynamic network models and gets a more accurate modeling result. The modeling performance of DeLaN-FFNN on the three test sets is shown in Fig. 6. DeLaN-FFNN demonstrated superior modeling performance for the UR-5 manipulator under uncertainties.

4.2 Real manipulator experiments

To demonstrate that DeLaN-FFNN is more suitable for real environments compared to DeLaN, we conduct the inverse dynamics model learning experiments on a publicly available dataset (http://gaussianprocess.org/gpml/data). This dataset is specifically designed for tackling an inverse dynamics
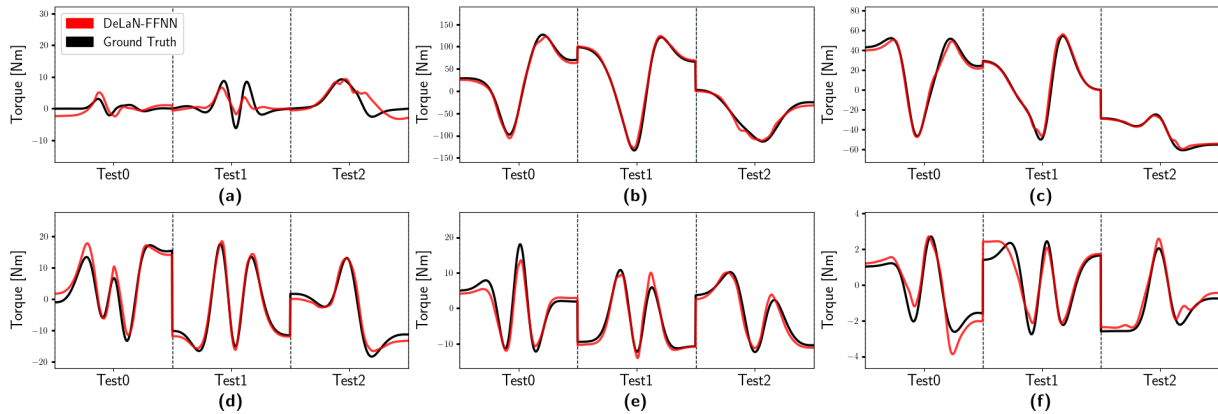
Fig. 6   Correspond to the inverse dynamics modeling of the six joints (0−5) of the manipulator using DeLaN-FFNN. The red color represents the predicted values by DeLaN-FFNN, and the black color represents the ground truth values.
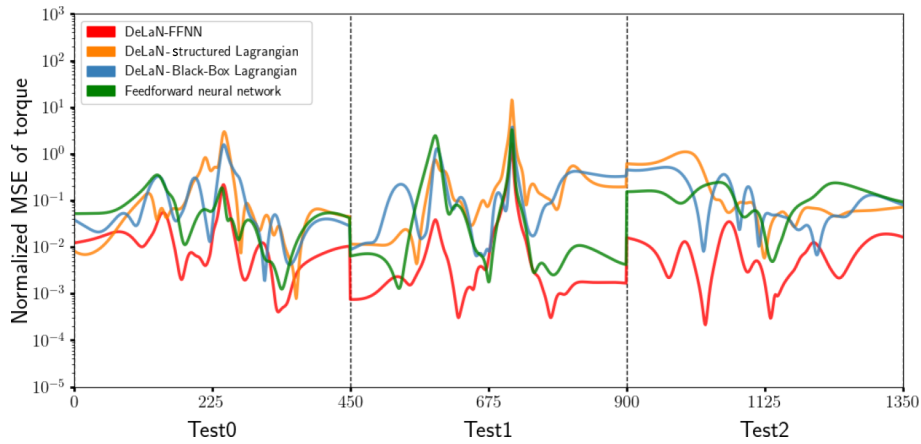

Fig. 7   Normalized MSE of the six joints on the data samples for different models in Test(0−2).

problem associated with a 7 degrees-of-freedom SARCOS anthropomorphic manipulator. The dataset encompasses a 21-dimensional feature space, which includes 7 joint positions, 7 joint velocities, and 7 joint accelerations. Each input instance in the dataset is accompanied by corresponding 7 joint torques. The dataset consists of 44 484 training examples and 4449 test examples[31]. We utilized all 44 484 training data samples for the training. In order to visualize the comparison results effectively, we randomly selected 100 test data samples from the total of 4449 test samples for testing and comparison.The visualization results can be seen in Fig. 8.

We test the modeling performance of each model on the subset of 100 data samples and the entire set of 4449 data samples, and the results of the inverse model are summarized in Table 1. The "Sum" column represents the sum of MSE values for each joint in the corresponding dynamic network model. From Table 1, regardless of the test dataset size, whether it is a subset

of 100 samples or the full 4449 samples, DeLaN-structured Lagrangian and DeLaN-Black-Box Lagrangian exhibit similar MSE on each joint. Therefore, DeLaN-structured Lagrangian and DeLaN-Black-Box Lagrangian achieve comparable modeling results. However, their MSE values are higher than that of FFNN, indicating that their modeling accuracy is not as good as FFNN, which lacks any prior physical constraints. Among all these neural networks, DeLaN-FFNN shows the lowest MSE for each joint, demonstrating superior modeling performance. This demonstrates that DeLaN-FFNN is better suited for dynamic modeling of real physical systems compared to DeLaN.

In Figs. 8a−8g, DeLaN-FFNN is capable of accurately modeling the inverse dynamics of the seven joints of the SARCOS manipulator. In Fig. 8h, we have visually compared the modeling accuracy of various neural networks for the SARCOS manipulator's dynamics, On average, when compared to DeLaN,
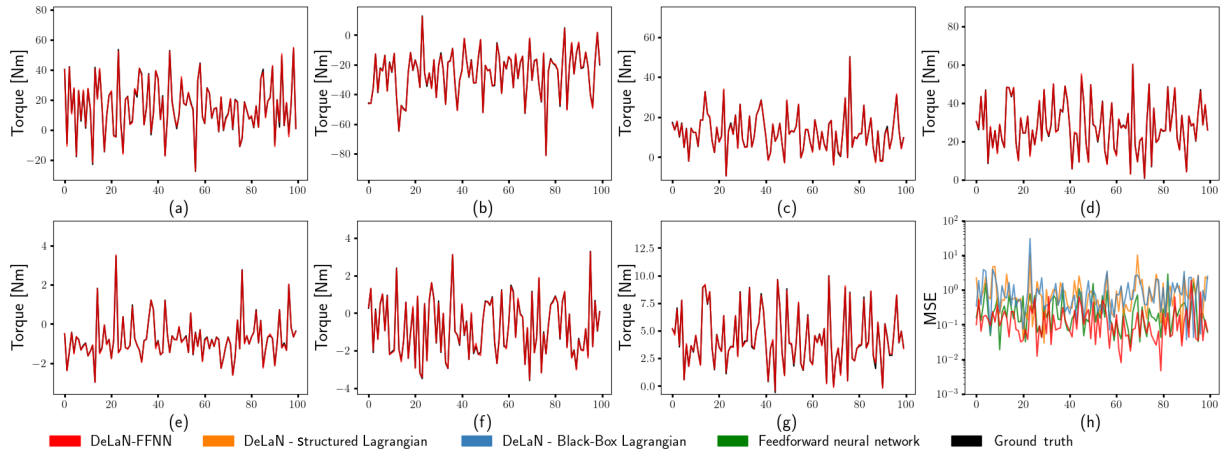
Fig. 8 **(a−g) correspond to the inverse dynamics modeling of the seven joints (0−6) of the manipulator using DeLaN-FFNN. (h) represents the MSE of the seven joints in a specific data sample** $(q, \dot{q}, \ddot{q}, \tau)_i$ **among the 100 test data samples.**

Table 1 **Data in the table represents the MSE between the actual torque values and the predicted torque values for each joint of a model.**

Value of samples	Model	Joint 0	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Sum
100	DeLaN-structured Lagrangian	5.182×10^0	1.312×10^0	3.857×10^{-1}	9.593×10^{-1}	5.091×10^{-3}	1.256×10^{-2}	2.152×10^{-2}	7.878×10^0
	DeLaN-Black-Box Lagrangian	5.980×10^0	1.754×10^0	4.853×10^{-1}	7.888×10^{-1}	7.834×10^{-3}	1.802×10^{-2}	2.992×10^{-2}	9.063×10^0
	Feedforward neural network	1.368×10^0	4.914×10^{-1}	1.748×10^{-1}	2.133×10^{-1}	3.745×10^{-3}	1.011×10^{-2}	1.415×10^{-2}	2.276×10^0
	DeLaN-FFNN	7.230×10^{-1}	2.291×10^{-1}	1.074×10^{-1}	1.484×10^{-1}	1.314×10^{-3}	3.572×10^{-3}	1.136×10^{-2}	1.224×10^0
4449	DeLaN-structured Lagrangian	7.708×10^0	1.677×10^0	4.717×10^{-1}	1.115×10^0	6.611×10^{-3}	1.696×10^{-2}	2.696×10^{-2}	1.102×10^1
	DeLaN-Black-Box Lagrangian	6.761×10^0	1.682×10^0	5.006×10^{-1}	1.121×10^0	7.663×10^{-3}	1.973×10^{-2}	2.947×10^{-2}	1.012×10^1
	Feedforward neural network	1.539×10^0	5.708×10^{-1}	1.763×10^{-1}	2.469×10^{-1}	3.180×10^{-3}	9.605×10^{-3}	1.425×10^{-2}	2.560×10^0
	DeLaN-FFNN	7.620×10^{-1}	3.148×10^{-1}	1.008×10^{-1}	1.984×10^{-1}	1.626×10^{-3}	5.015×10^{-3}	9.165×10^{-3}	1.392×10^0

FFNN achieved a lower MSE. Therefore, in the real manipulator applications, FFNN has demonstrated better modeling performance compared to DeLaN-Black-Box Lagrangian and DeLaN-structured Lagrangian. Additionally, among all the networks, DeLaN-FFNN achieved the lowest MSE, indicating that its modeling accuracy surpasses that of FFNN. DeLaN-FFNN has demonstrated the most precise modeling results and is considerably better suited for modeling the dynamics of real physical systems.

5   Conclusion

This work proposes a novel Deep Lagrangian Network based on the Deep Lagrange Network. While maintaining the advantage of preserving the physical rationality of DeLaN, this network incorporates a standard deep network to approximate the nonconservative dynamics that can not be fully represented by the DeLaN. Then by applying this network, we learn the inverse model of two multi-degree of manipulators: UR-5 and SARCOS manipulator under uncertainties from the trajectory data. The experiments evaluate the model learning performance for these two manipulators and the results show that DeLaN-FFNN outperforms existing approaches. In future work, this network should be applied to model a wider system class and a real physical robot system. The application of the proposed network in model based control is also to be studied.

References

[1]   M. Spong, K. Khorasani, and P. Kokotovic, An integral manifold approach to the feedback control of flexible joint robots, *IEEE J. Robot. Auto.*, vol. RA-3, no. 4, pp. 291–300, 1987.

[2]   S. V. Shah, S. K. Saha, and J. K. Dutt, Modular framework for dynamic modeling and analyses of legged robots, *Mech. Mach. Theory*, vol. 49, pp. 234–255, 2012.

[3]   M. Jafarinasab, S. Sirouspour, and E. Dyer, Model-based motion control of a robotic manipulator with a flying multirotor base, *IEEE-ASME Trans. Mechatron.*, vol. 24, no. 5, pp. 2328–2340, 2019.

[4]   K. Hitzler, F. Meier, S. Schaal, and T. Asfour, Learning and adaptation of inverse dynamics models: A comparison, in *Proc. 2019 IEEE-RAS 19th Int. Conf. Humanoids Robot. (Humanoids)*, Toronto, Canada, 2019, pp. 491–498.

[5]   M. Jansen, Learning an accurate neural model of the dynamics of a typical industrial robot, https://www. semanticscholar.org/paper/Learning-an-Accurate-Neural-Model-of-the-Dynamics-a-Jansen/28f6446fd74986bad-9b179e21d6645faf5b88e22, 1994.

[6]   D. Nguyen-Tuong, J. Peters, M. Seeger, and B. Schlkopf, Learning inverse dynamics: A comparison, in *Proc. 16th Eur. Symp. Artificial Neural Networks*, Bruges, Belgium, 2008, pp. 13–18.

[7]   S. Greydanus, M. Dzamba, and J. Yosinski, Hamiltonian neural networks, in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, Vancouver, Canada, 2019, pp. 15379–15389.

[8]   M. Raissi, Deep hidden physics models: Deep learning of nonlinear partial differential equations, *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 932–955, 2018.

[9]   D. Y. Zhong, B. Dey, and A. Chakraborty, Symplectic ode-net: Learning hamiltonian dynamics with control, arXiv preprint arXiv:1909.12077, 2019.

[10]   S. Saemundsson, A. Terenin, K. Hofmann, and M. Deisenroth, Variational integrator networks for physically structured embeddings, arXiv preprint arXiv:1910.09349, 2020.

[11]   M. Lutter, R. Christian, and J. Peters, Deep Lagrangian Networks: Using physics as model prior for deep learning, arXiv preprint arXiv: 1907.04490, 2018.

[12]   M. Lutter, K. Listmann, and J. Peters, Deep Lagrangian Networks for end-to-end learning of energy-based control for under-actuated systems, in *2019 IEEE/RSJ Int. Conf. Intel. Robots Sys. (IROS)*, Macau, China, 2019.

[13]   M. Lutter, and J. Peters, Combining physics and deep learning to learn continuous-time dynamics models, *Int. J. Robot. Res.*, vol. 42, no. 3, pp. 83–107, 2023.

[14]   X. Li, W. Shang, and S. Cong, Offline reinforcement learning of robotic control using deep kinematics and dynamics, *IEEE-ASME Trans. Mechatron.*, 2023.

[15]   J. K. Gupta, K. Menda, Z. Manchester, and M. J. Kochenderfer, A general framework for structured learning of mechanical systems, arXiv preprint arXiv: 1902.08705, 2019.

[16]   M. Lutter, J. Silberbauer, J. Watson, and J. Peters, A differentiable Newton Euler algorithm for multi-body model learning, arXiv preprint arXiv:2010.09802, 2020.

[17]   L. Liu, G. Zuo, J. Li, and J. Li, Dynamics modeling with realistic constraints for trajectory tracking control of manipulator, in *Proc. 2022 IEEE Int. Conf. Real-time Comput. Robot. (RCAR)*, Guiyang, China, 2022, pp. 99–104.

[18]   C. Ritter, Deep learning of inverse dynamic models, Master thesis, Technical University of Darmstadt, Darmstadt, Germany, 2018.

[19]   Z. Liu, B. Wang, Q. Meng, W. Chen, M. Tegmark, and T. Y. Liu, Machine-learning nonconservative dynamics for new-physics detection, *Phys. Rev. E*, vol. 104, no. 5, p. 055302, 2021.

[20]   M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, Lagrangian neural networks, arXiv preprint arXiv: 2003.04630, 2020.

[21]   C. Huang and W. Yang, Inverse dynamics modeling of robots based on sparse spectral gaussian process regression, *J. Phys. Conf. Ser.*, vol. 2010, no. 1, p. 012136, 2021.

[22]   D. Nguyen-Tuong and J. Peters, Model learning for robot control: A survey, *Cognit. Process.*, vol. 12, no. 4, pp. 319–340, 2011.

[23]   A. Wahrburg, J. Bös, K. D. Listmann, F. Dai, B. Matthias, and H. Ding, Motor-current-based estimation of cartesian contact forces and torques for robotic manipulators and its application to force control, *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 879–886, 2017.

[24]   B. Bona and M. Indri, Friction compensation in robotics: An overview, in *Proc. 44th IEEE Conf. Decision Control*, Seville, Spain, 2005, pp. 4360–4367.

[25]   A. Albu-Schäffer, Control of robots with elastic joints using the example of DLR lightweight arms, Ph.D. dissertation, Technical University of Munich, Munich, Germany, 2002.

[26]   D. Q. Cao and R. W. Tucker, Nonlinear dynamics of elastic rods using the Cosserat theory: Modelling and simulation, *Int. J. Solids Struct.*, vol. 45, no. 2, pp. 460–477, 2008.

[27]   P. K. Khsla, Categorization of parameters in the dynamic robot model, *IEEE Trans. Robot. Autom.*, vol. 5, no. 3, pp. 261–268, 1989.

[28]   J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, and S. Wanderman-Milne, JAX: Composable Transformations of Python+ NumPy Programs (v0. 2.5), Software available from https://github.com/google/jax, 2018.

[29]   E. Coumans and Y. Bai, PyBullet quickstart guide,

https://usermanual.wiki/Doc-ument/PyBullet20Qu-ickstart20Guide, 2023.

[30] J. Y. Luh, M. W. Walker, and R. P. Paul, On-line computational scheme for mechanical manipulators, *J. Dyn. Sys., Meas., Control*, vol. 102, no. 2, pp. 69–76, 1980.

[31] S. Vijayakumar and S. Schaal, Locally weighted projection regression: An $O(n)$ algorithm for incremental real time learning in high dimensional space, https://www.semanticscholar.org/paper/Locally-Weighted-Projection-Regression-:-An-O(n)-in-Vijayakumar-Schaal/749c81cc86c3a946424332f3866474b4a1e8014f3, 2000.

Shuangshuang Wu received the BEng degree in automation and the PhD degree in control science from Yanshan University, China, in 2014 and 2020, respectively. From 2018 to 2019, she was a visiting researcher with School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ, USA. From 2020 to 2022, she worked as a postdoctoral researcher with Department of Computer Science and Technology in Tsinghua University. She is currently a lecturer with School of Automation, Beijing Information Science and Technology University, China. Her current research interests include time-delay systems, and intelligent modeling and control.

Wenbai Chen received the BS degree from Northwestern University, China in 1997, the MS degree from Yanshan University in 2004, and the PhD degree from Beijing University of Posts and Telecommunications in 2011. He is currently a professor with School of Automation, Beijing Information Science and Technology University. His current research interests include intelligent robot, artificial intelligence, sensor fusion, machine learning, and wireless sensor network.

Zhiming Li received the BEng degree in automation from Lanzhou Jiaotong University, China, in 2022. He is currently a master student in control science and engineering at Beijing Information Science and Technology University, China. His main research interests include deep learning, robot modeling, and control.

Fuchun Sun received the BS and MS degrees from Naval Aeronautical Engineering Academy, Yantai, China, in 1986 and 1989, respectively, and the PhD degree from Tsinghua University, Beijing, China, in 1998. From 1998 to 2000, he was a postdoctoral researcher with Department of Automation, Tsinghua University, where he is currently a professor with Department of Computer Science and Technology. His current research interests include intelligent control, neural networks, fuzzy systems, variable structure control, nonlinear systems, and robotics. He is a fellow of IEEE.