Learning Globally Stable Neural Imitation Policies via Reshaped Energy Functions

Haohui Huang, *Member, IEEE*, Wenxu Zheng, Junda Duan, Kailun Huang, Jing Guo, and Chenguang Yang, *Fellow, IEEE*

***Abstract*—Energy function based imitation learning holds tremendous potential for developing point-to-point motion models with stability and accuracy. Nevertheless, there are still hard issues related to learning an appropriate energy function through distilling the motion feature from a non-convex motion. This paper proposes a novel adaptive modulation scheme that aims to learn a neural Lyapunov function, thereby tightly decoupling the trajectory generation and trajectory modulation to alleviate the constraints of traditional neural-learning based methods. Given a neural Lyapunov learning problem, we decouple and modulate the radius and angle of motion position in polar-like space through coordinate transformation to ensure global stability and learning accuracy for point-to-point non-convex motion learning and generalization. The optimization-based scheme guarantees that the neural value along the non-convex trajectory decreases monotonically, leading to smooth motion generation. Both simulations using a public benchmark and real-world evaluations of ultrasound robot scanning demonstrate that the proposed RAM method exhibits superior generalization and higher accurate ability for non-convex motions compared to existing motion learning methods.**

***Index Terms*—Learning from demonstrations; Non-convex motion; Neural networks; Dynamic systems.**

I. Introduction

ROBOTICS technology has made substantial progress in various industries. However, enabling non-professionals to conveniently use robots remains a significant challenge. A widely recognized solution is to impart skills to robots through Learning from Demonstrations (LfD) [1]. By observing human demonstrations, robots can directly replicate human behaviors, thereby learning and mastering various tasks and skills more efficiently. These tasks encompass polishing [2], handwriting [3], ultrasonic detection [4], assembly [5], and more. This article focuses on methods for extracting human skills from demonstrations.

Encoding human motion rules using dynamical systems [6] has been proven to be effective. Dynamical systems are generally divided into two main types: autonomous dynamical systems (ADS) and non-autonomous dynamical systems (N-ADS) [4]. The key distinction between them lies in whether the system's evolution depends on time. In autonomous dynamical systems, evolution depends solely on the internal state and is independent of external time. Conversely, in non-autonomous dynamical systems, both the internal state and external time influence evolution. Dynamical systems are commonly described by differential equations. In non-autonomous systems, time is explicitly included as a variable in these equations, indicating that changes in the state of system depend on both time and the internal state.

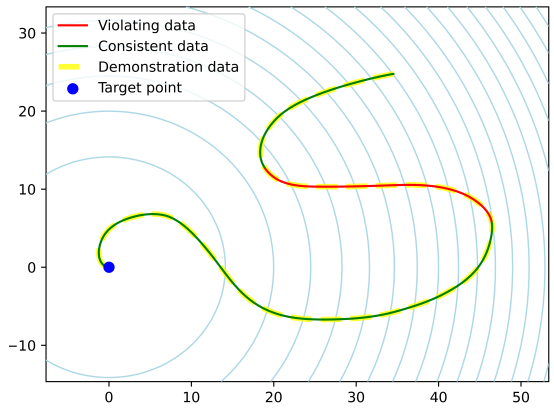

Fig. 1. This illustrates the 'Snake' situation in the LASA dataset when the energy function is $\xi^T\xi$. The green line represents locations that align with the energy gradient, while the red line indicate those that violate it.

Theoretically, any regression technique can be employed to model nonlinear dynamical systems, regardless of whether the system belongs to autonomous or non-autonomous. A typical approach is to use Gaussian Mixture Regression (GMR) [7], [8]. However, some studies have indicated limitations in the generalization capability of the GMR algorithm [9]. Consequently, researchers have proposed refined methodologies [10] within the Gaussian Mixture Regression framework to address this issue. Additionally, Gaussian Process Regression (GPR) [11] is another noteworthy approach. Rooted in Gaussian processes, GPR aims to model the joint distribution of inputs and outputs, allowing for flexible fitting and prediction of nonlinear relationships. Moreover, regression techniques such as Locally Weighted Projection Regression (LWPR), Support Vector Regression (SVR), Reservoir Computing (RC), and Gaussian Process Latent Variable Models (GPLVM) have been employed to estimate complex dynamic systems, achieving varying degrees of success.

Most of the aforementioned regression techniques are based on probability distributions, which provide the system with robust generalization capabilities. However, these algorithms may struggle with large datasets. For instance, while Gaussian process regression may yield satisfactory outcomes with small datasets, the computational load increases dramatically as the

Haohui Huang, Wenxu Zheng, Junda Duan, Kailun Huang, and Jing Guo are with the School of Automation, Guangdong University of Technology, Guangzhou 528300, China.

Chenguang Yang is with the Department of Computer Science, University of Liverpool, L69 3BX Liverpool, U.K (e-mail: cyang@ieee.org).

This work was supported in part by the National Natural Science Foundation of China under Grant 62303120.

dataset size grows. It is widely acknowledged that neural networks offer a viable alternative for regression tasks on large datasets. Common neural network architectures include Multi-Layer Perceptron (MLP) [12], convolutional neural networks (CNN) [13], and Transformers [14]. These methods have been proven effective not only in the field of image processing but also in demonstration learning [3], [15], [16]. These neural networks provide distinct advantages, particularly their end-to-end nature, which allows for flexible adjustments to the network structure at any point. Consequently, the approach proposed here leverages neural networks (NNs) to model the data and enhance algorithmic flexibility.

Previous experience has shown that maintaining the stability of dynamic systems is one of the most challenging and important tasks in applying dynamic systems [17]. Learning dynamic systems through NNs allows us to capture the complex features of the data, but it cannot guarantee the stability of these systems. Therefore, additional strategies must be developed to ensure the stability of the system.

The stability strategies employed by autonomous dynamical systems and non-autonomous dynamical systems typically differ. For most involuntary dynamic systems, the stability of the system can be easily guaranteed by relying on clock signals. A straightforward method for maintaining stability in N-ADS involves leveraging the clock to smoothly transition unstable nonlinear systems into globally asymptotically stable linear systems. Building on this idea, researchers have proposed the Energy-based Stabilizer of Dynamical Systems (ESDS) [18]. Additionally, incremental learning has been achieved through the seamless switching of dynamic systems [19]. However, due to their time dependency, these methods are susceptible to disturbances [17].

Another effective approach for maintaining the stability of dynamic systems is the application of Dynamic Movement Primitives (DMP). DMP utilizes a second-order dynamic system with self-stabilizing properties to construct an "attractor" model. DMP is categorized into Discrete DMP and Rhythmic DMP. Discrete DMP focuses on spatial trajectory planning, while Rhythmic DMP addresses the planning of rhythmic trajectories for continuous periodic motions [20]. Beyond the classical DMP, several enhanced versions exist. For example, the improved DMP proposed in [21] addresses the classical DMP's inability to fit curves when the start and end points are very close together. Some researchers have incorporated regression techniques into the DMP framework [22], enhancing its performance in learning from multiple demonstrations. Although the stability of DMPs can be guaranteed, their special structure separates each motion dimension, which reduces their ability to reproduce and generalize [23].

Another typical approach utilizes the Lyapunov function (LF) as a stability certificate [4]. A seminal work in this area is the Stable Estimator of Dynamical Systems (SEDS) [24], which introduces a quadratic function (QF) as the LF, ensuring the stability of the learned system. However, this approach significantly compromises accuracy and fails to effectively capture the demonstrated preferences. To address this, [17] improves upon the SEDS by employing a weighted sum of asymmetric quadratic functions (WSAQF) as the LF, which enhances accuracy to some extent. Some researchers have proposed using a sum of squares function (SOS) to replace the WSAQF, achieving promising results as well [25], but the function may have multiple minima. This implies that the generated trajectory may converge to an unexpected location. Additionally, with advancements in neural networks, stable dynamical systems have been constructed in an end-to-end manner. For instance, some researchers have utilized input-convex neural networks (ICNNs) as the LF to achieve end-to-end learning [26], [27]. This algorithm has demonstrated strong performance in high-dimensional input scenarios and has been applied to stable video texture generation. However, it struggles with LfD due to the strict constraints imposed by the convex nature of ICNNs. Recently, some scholars have analyzed a novel energy function, NEUM [4], which refers to a neural energy function with the unique-minimum property. The characteristic of this function is that its value increases along the rays emitted from the origin. We call an energy function with such properties a radially increasing energy function. Researchers believe that NEUM is more general than other functions and has achieved promising learning results in robot experiments.

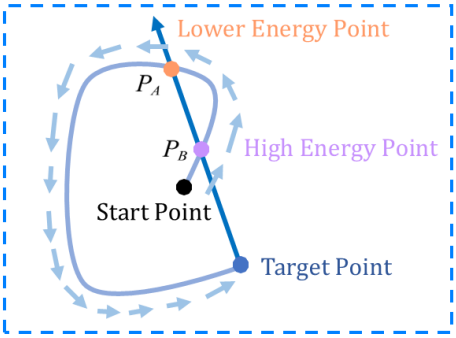

Fig. 2. Examples that are difficult to learn using NEUM. According to our learning objective, the energy at point PA should be lower than that at point PB. However, since both PA and PB lie on a ray emitted from the origin, and NEUM is radially increasing, the energy at PA can never be lower than that at PB. It's a contradiction between trajectory and energy.

However, this study found that NEUM is not suitable for all situations. For example, in the 2D case, NEUM is unable to effectively learn the trajectory depicted in Fig. 2. When we use an energy function to ensure stability, if the learned energy function perfectly fits the data, the energy of the points in the demonstrated trajectory should gradually decrease from the starting point to the target point, with the energy reaching zero at the target point. It is clear that using NEUM does not perform well in learning the example in Fig. 2. This is because, according to our learning objective, the energy at point P_A should be lower than that at point P_B. However, since both P_A and P_B lie on a ray emitted from the origin, and NEUM is radially increasing, the energy at P_A can never be lower than that at P_B. It is a contradiction between trajectory and energy. The characteristic of this type of trajectory is that, starting from the origin, there exists a ray emitted from the origin such that the trajectory first passes through points on the ray that are farther from the origin, and then passes through

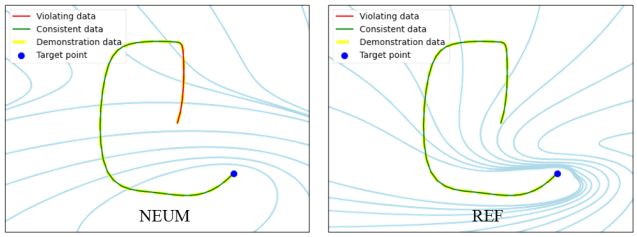

Fig. 3. The results of adapting to the example in Fig. 2 using NEUM and REF(Ours). The results demonstrate that REF achieves precise fitting to the given example, whereas NEUM exhibits significant adaptation deficiencies.

points on the ray that are closer to the origin. We refer to trajectories with such contradictions as Energy Non-Radially Increasing Trajectories. We call the energy function with this property the radial increasing energy function.

In this paper, we introduce a Reshaped Energy Function (REF) to address this issue and compare it with NEUM. To illustrate the differences between the proposed REF method and NEUM, we provide a comparison of their learning performance on the given example, as shown in 3. Analysis indicates that REF can learn some forms of energy functions that NEUM cannot capture, suggesting that our energy function possesses greater expressive capability. The proposed energy function in this paper is a more general form, while QF, WASQF, ICNN, and NEUM are special cases of energy functions. WASQF is a quadratic like function, ICNN is a convex function, and NEUM is a radially increasing function. Specifically, our contributions are as follows:

1) We propose two types of coordinate transformations applicable to the energy function, modulating the equipotential curves of the energy function in terms of length and angle, respectively.
2) We propose a Reshaped Lyapunov Function based on Coordinate Transformations. Through analysis, we demonstrate that our function is more general than most existing Lyapunov functions.
3) We construct a single-step learning model using the REF and FCNN. Unlike ESDS, DMP, etc., our approach is based on autonomous dynamical systems.
4) We conduct simulation tests on the LASA dataset and performed two representative experiments on a real robot. The experimental results indicate that our algorithm achieves a balance between stability and accuracy.

The remainder of this paper is organized as follows. In Section II, we introduce the fundamental issues and key theories of the method. Section III presents the principles of the method and conducts stability analysis on the designed structure. Trustworthy experimental results are provided in Section IV. Finally, Section V concludes the paper.

II. PROBLEM STATEMENT

In this section, we discuss the fundamental formulas of dynamical systems (DS) and methods for maintaining their stability in preparation for the subsequent analysis.

We decompose a complex motion process into several submotions, each ending at single points, which can be learned

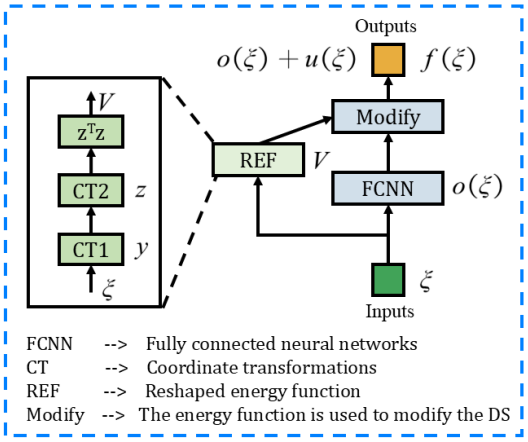

Fig. 4. The overall results of the proposed method and the schematic of the entire dynamic system. REF uses two coordinate transformations to reconstruct a simple energy function. The dynamic system first uses a fully connected neural network (FCNN) for fitting, and then is corrected using a projection algorithm to obtain the final stable dynamic system.

using autonomous nonlinear dynamical systems. Autonomous nonlinear dynamical systems are described as follows:

$$\dot{\xi} = f(\xi) \tag{1}$$

where $\xi \in \mathbb{R}^n$ represents a state variable $\xi = [\xi_1, \xi_2, \ldots, \xi_n]^T$ with n dimensions. Specifically, it can represent the end effector of a robotic arm or the joint angles of a robotic arm. $\dot{\xi} \in \mathbb{R}^n$ is the first-order derivative of the state variable, which can represent the velocity of the robotic arm's end effector or the velocity of the joint angles. $f : \mathbb{R}^n \to \mathbb{R}^n$ is a function encoding the dynamics of the system. Learning this function relationship will yield a velocity function related to position. For example, the simplest autonomous dynamic system $\dot{\xi} = -\xi$ is a contraction vector field with the origin as its target. Simultaneously, the velocity of the system at a certain state is only related to the position of that point.

To learn the dynamical system (DS) representing each subtask, we collect a dataset $D = \{\xi_{t,l}, \dot{\xi}_{t,l}\}_{t=1,l=1}^{T,L}$, where $\xi_{t,l}$ denotes the robot's position at time step t of the lth demonstration trajectory, T represents the duration of each demonstration, and L represents the number of demonstration trajectories. With this dataset, we can construct any neural network and then use gradient descent to learn the DS. The loss function for gradient descent can be expressed using the following formula:

$$Loss = \sum_{l=1}^{L}\sum_{t=1}^{T} \|f(\xi_{t,l}) - \dot{\xi}_{t,l}\|^2 \tag{2}$$

To ensure stability, f is represented as a specially designed neural network, whose structure is illustrated in Fig. 1. Lyapunov stability theory can be utilized to ensure system stability, characterized by the convergence of states to a constant value ξ^*. According to Lyapunov's theory, it is essential to

TABLE I
COMPARISON OF DIFFERENT APPROACHES TO LEARNING STABLE DYNAMICAL SYSTEMS USING DIFFERENT ENERGY FUNCTIONS.

	Unique-minimum	Multiple motions	Single-step learning	Learning energy non-radially increasing trajectories
QF [8]	✓	✓	—	—
WASQF [17]	✓	✓	—	—
SOS [25]	—	✓	—	—
ICNN [26]	✓	✓	✓	—
NEUM [4]	✓	✓	—	—
REF (Ours)	✓	✓	✓	✓

identify an energy function $V(x)$ that has a unique minimum and satisfies the following condition

$$\begin{cases} (a)\, V(\xi-\xi^*)=0, \xi=\xi^* \\ (b)\, V(\xi-\xi^*)>0, \forall \xi\neq\xi^* \\ (c)\, \dot{V}(\xi-\xi^*)=0, \xi=\xi^* \\ (d)\, \dot{V}(\xi-\xi^*)<0, \forall \xi\neq\xi^* \end{cases} \tag{3}$$

where $V:\mathbb{R}^n\to\mathbb{R}$ is a function with scalar output. Without loss of generality, we can define ξ^* as the origin, thus having $\xi^*=0$. We can ensure conditions (a) and (b) in Eq. (3) by specially designing the energy function V; this will be discussed in Section III. For conditions (c) and (d) in Eq. (3), we not only need to consider the expression of the function V but also $f(x)$. Because $\dot{V}(\xi)$ can be determined by the following equation:

$$\dot{V}(\xi)=\dot{\xi}^T\frac{\partial V(\xi)}{\partial \xi}=f(\xi)^T\frac{\partial V(\xi)}{\partial \xi} \tag{4}$$

To achieve Eq.(3)(c), Eq.(4) indicates that V must satisfy $\frac{\partial V(\xi)}{\partial \xi}=0$ for $\xi=0$. To achieve Eq.(3)(d), we first use a neural network representation of $o(\xi)$ as a basis, and then we correct $o(\xi)$ to $f(\xi)$ using a projection algorithm:

$$f(\xi)=o(\xi)-\frac{\text{ReLu}\left(o(\xi)^T\frac{\partial V(\xi)}{\partial \xi}+\rho(\xi)\right)}{\left\|\frac{\partial V(\xi)}{\partial x}\right\|_2^2}\frac{\partial V(\xi)}{\partial \xi} \tag{5}$$

where $\rho(\xi)$ is any positive-definite function and the activation function $\text{ReLu}(x)$ has the form

$$\text{ReLu}(x)=\begin{cases} x, x>0 \\ 0, else \end{cases} \tag{6}$$

The formula indicates that V must satisfy $\frac{\partial V(\xi)}{\partial \xi}\neq 0$ for $\xi\neq 0$. If $\frac{\partial V(\xi)}{\partial \xi}\neq 0$ holds for $\xi\neq 0$, Eq. (3)(d) can be derived based on Eq. (4) and Eq. (5). If $o(x)^T\frac{\partial V(\xi)}{\partial \xi}+\rho(x)<0$, then we have:

$$\dot{V}=o(\xi)^T\frac{\partial V(\xi)}{\partial \xi}<-\rho(\xi) \tag{7}$$

If $o(x)^T\frac{\partial V(\xi)}{\partial \xi}+\rho(x)\geq 0$, then we have:

$$f(\xi)=\left(I-\frac{\frac{\partial V(\xi)}{\partial \xi}\frac{\partial V(\xi)}{\partial \xi}^T}{\|\frac{\partial V(\xi)}{\partial \xi}\|_2^2}\right)o(\xi)-\rho(\xi)\frac{\frac{\partial V(\xi)}{\partial \xi}}{\|\frac{\partial V(\xi)}{\partial \xi}\|_2^2} \tag{8}$$

Substituting Eq. (8) into Eq. (4), we obtain

$$\dot{V}=f(\xi)^T\frac{\partial V(\xi)}{\partial \xi}=-\rho(\xi)\frac{\frac{\partial V(\xi)}{\partial \xi}^T\frac{\partial V(\xi)}{\partial \xi}}{\|\frac{\partial V(\xi)}{\partial \xi}\|_2^2}=-\rho(\xi) \tag{9}$$

In summary, for all $\xi\neq\xi^*$, we have:

$$\dot{V}\leq-\rho(\xi) \tag{10}$$

Moreover, since $\rho(\xi)$ is an arbitrary positive definite function, it follows that $\dot{V}(\xi)<0, \forall\xi\neq\xi^*$.

In fact, an autonomous dynamic system is locally asymptotically stable if there exists a continuously differentiable Lyapunov function that satisfies Eq. (3). Furthermore, if the following conditions are met, it is proven that the system has global stability

$$\lim_{\|\xi\|\to+\infty}V(\xi)=+\infty \tag{11}$$

It is worth noticing that the DS requires globally asymptotically stable. Therefore, it is necessary to simultaneously satisfy Eq. (3) and Eq. (11). Specifically, the analysis of $o(x)$ and $V(x)$ will be introduced in the following sections.

III. Proposed approaches

In this section, we propose a complex energy function reshaped from a quadratic function based on coordinate transformations. First, we introduce the basic form of coordinate transformations and the energy function. Next, we provide a two-dimensional example to illustrate the underlying ideas of the proposed method. Finally, we demonstrate that the energy function satisfies a series of necessary conditions.

A. Basic Forms of Energy Functions

In this paper, a quadratic function is employed as the energy function and two coordinate transformations are utilized to reshape its form. The energy function defined by the equation $V(\xi)=z^Tz$. The variable z in the equation is a function of ξ, obtained through two consecutive coordinate transformations applied to ξ. The coordinate transformations defined by the equation:

$$\begin{cases} z=f_\lambda(y) \\ y=f_R(\xi) \end{cases} \tag{12}$$

where f_R:$\mathbb{R}^n\to\mathbb{R}^n$ and f_λ:$\mathbb{R}^n\to\mathbb{R}^n$ represent two different coordinate transformations. The energy function can then be expressed by the equation:

$$V(\xi)=z^Tz=f_\lambda(f_R(\xi))^Tf_\lambda(f_R(\xi)) \tag{13}$$

The energy function must incorporate learnable parameters, as only energy functions endowed with such parameters are capable of capturing the flow direction of the demonstrated trajectories from the dataset, thereby mitigating the influence

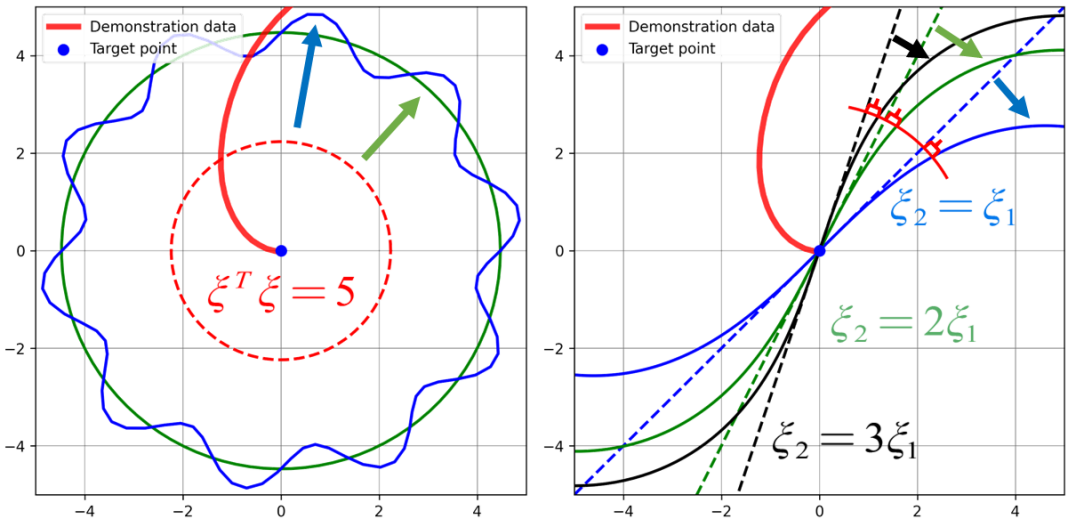

Fig. 5. Left figure: The red dashed line represents a closed circular curve in space when neither of the two transformations is active. When only the scaling transformation is applied, points on the red dashed line are radiated outward, as illustrated by the green and blue solid lines. Right figure: The dashed line represents a straight line in space when neither of the two transformations is active. When only the rotation transformation is applied, points on the dashed line are distorted and consequently move to the positions indicated by the solid line.

of the energy function on the original dynamical system. Therefore, both f_R and f_λ are parameterized. As stated in the Section I, our method is primarily designed to address the issue that the radial increasing energy function fails to learn the radial increasing energy function. Therefore, we first need to construct a radial increasing energy function, which is the role of f_λ. After that, f_R is used to break the constraints of the radial increasing energy function. Since these two components are analogous to scaling and rotation in polar coordinates, we refer to f_λ as the scaling transformation and f_R as the rotation transformation in this paper.

B. Basic Form of Coordinate Transformations

The scaling transformation f_λ is primarily used to construct the radial increasing energy function. As stated in Section I, an existing radial increasing energy function, namely NEUM, is available. However, the parameters of NEUM are subject to complex constraints, as detailed in [4]. To avoid the cumbersome handling of these constraints, we propose a novel structure to establish the radial increasing energy function:

$$\begin{cases} f_\lambda(y) = \lambda(y) \cdot y \\ \lambda(y) = \omega(\vec{r}_y)^T q(y) \\ q(y) = [q_1(y), ..., q_i(y), ..., q_m(y)]^T \\ q_i(y) = \varrho\left(\alpha_i(\vec{r}_y) r_y + \beta_i(\vec{r}_y)\right) \\ \varrho(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} \\ \vec{r}_y = \frac{y}{r_y} \\ r_y = \|y\| \end{cases} \tag{14}$$

where r_y represents the magnitude of the state y, $\varrho(s) : \mathbb{R} \to \mathbb{R}$ is an activation function known as the tanh function, $q(y)$ is the output of a neural network with n neurons, $q_i(y)$ represents the i-th component of $q(y)$. Additionally, $\omega_i(\vec{r}_y)$, $\alpha_i(\vec{r}_y)$, and $\beta_i(\vec{r}_y)$ represent the i-th component of $\omega(\vec{r}_y)$, $\alpha(\vec{r}_y)$, and $\beta(\vec{r}_y)$. Normally, ω, α, and β, as parameters of the neural network, should be constant values. However, in order to derive the Lyapunov stability conditions, the parameters here need to satisfy $\alpha_i > 0$ and $\omega_i > 0$. It introduces additional constraints in the final optimization, which is not conducive to adjusting the neural network weights. In our approach, we set ω, α, and β as outputs of three fully connected neural networks and can be defined by the equation:

$$\begin{cases} \omega(\vec{r}_y) = f_{\lambda_1}(\vec{r}_y, \Theta_1), \omega \in \mathbb{R}^m \\ \alpha(\vec{r}_y) = f_{\lambda_2}(\vec{r}_y, \Theta_2), \alpha \in \mathbb{R}^m \\ \beta(\vec{r}_y) = f_{\lambda_3}(\vec{r}_y, \Theta_3), \beta \in \mathbb{R}^m \end{cases} \tag{15}$$

where the input $\vec{r}_y \in \mathbb{R}^n$ represents the direction of the state and can be defined by the equation $\vec{r}_y = \frac{y}{r_y}, \forall r_y \neq 0$. Θ_{λ_1}, Θ_{λ_2}, and Θ_{λ_3} are trainable parameters, f_{λ_1} and f_{λ_2} represent the outputs of two arbitrary neural networks activated by the activation function $\delta_f(s) = e^s$, which ensures that $\omega_i(\vec{r}_y) > 0$ and $\alpha_i(\vec{r}_y) > 0$, f_3 represents an arbitrary neural network. Since the inputs to f_{λ_1}, f_{λ_2}, and f_{λ_3} are $\vec{r}_y = \frac{y}{r_y}$, for any $r_y \neq 0$, it holds that $\frac{d\omega_i}{dr_y} = 0$, $\frac{d\alpha_i}{dr_y} = 0$, and $\frac{d\beta_i}{dr_y} = 0$, as the direction and magnitude are decoupled. From the above structure, it can be observed that for any $\alpha_i(\vec{r}_y) > 0$, $\alpha_i(\vec{r}_y) r_y + \beta_i(\vec{r}_y)$ increases as r_y increases, and $\varrho(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$ is an increasing function. Therefore, $q_i(y)$ is an increasing function of r_y, meaning $q_i(y)$ is an increasing function of r_y. Since the weighted sum of increasing functions remains an increasing function, $\lambda(y)$ is an increasing function of r_y, and thus $\frac{d\lambda(y)}{dr_y} > 0$. $f_R(y)$ represents a spatial transformation of the input y, causing points in the y-space to radiate outward along a ray originating from the origin.

To break the constraints of the radial increasing energy function, we apply the rotation transformation between the original input ξ and the input y of the scaling transformation. The rotation transformation performs pairwise rotational adjustments across different dimensions in the ξ-space. The rota-

tion transformation between these two components is defined as $k(\theta, s)$,where θ represent rotation angle and $s = [s_1, s_2]$ represent a two-dimensional input to be modulated. This formula can also be applied to the rotation of points

$$k\left(\theta,\ s\right) = \begin{bmatrix} \cos\left(\theta\right) & \sin\left(\theta\right) \\ -\sin\left(\theta\right) & \cos\left(\theta\right) \end{bmatrix} s \tag{16}$$

To enhance the expressive power of rotational modulation, we parameterize θ and the rotation transformation can be rewritten as follows:

$$\begin{cases} \theta = \ f_\theta\left(r_s\right) \\ r_s = s_1^2 + s_2^2 \end{cases} \tag{17}$$

where θ is represented by an arbitrary neural network. In this formula, we can easily see that any point with the same value of r_s will be rotated by the same angle. To apply this structure to higher-dimensional cases, we need to extend it in a recursive form. This paper uses pseudocode to describe the proposed structure, as shown in Algorithm 1. In the expression, the function $f_{\theta_1}, f_{\theta_2}, ..., f_{\theta_{n-1}}$ is represented by arbitrary neural networks. The entire rotation transformation requires $n-1$ neural networks. Furthermore, the input dimension of each neural network is 1. It is easily find that f_R has an inverse structure, i.e., there exists $\xi = f_R^{-1}(y)$. According to the Inverse Function Theorem, f_R and its inverse function satisfy $J_{f_R^{-1}}(f_R(\xi)) \cdot J_{f_R}(\xi) = I$ where J_{f_R} and $J_{f_R^{-1}}$ represent the Jacobian matrices. This implies that J_{f_R} is of full rank, i.e., $\frac{\partial f_R}{\partial y} \neq 0$. This conclusion will be used in the subsequent theoretical derivations.

Algorithm 1 $f_R(\xi)$

Input: ξ

$i \leftarrow 1$
while $i < n$ **do**
  $s_1, s_2 \leftarrow \xi_i, \xi_{i+1}$
  Calculate r_s from Eq.(17)
  $\theta \leftarrow f_{\theta_i}(r_s)$
  Calculate $k(\theta, s)$ from Eq.(16)
  $\xi_i, \xi_{i+1} \leftarrow k_1, k_2$
  $i \leftarrow i + 1$
end while
return ξ

Output: $f_R(\xi)$

As illustrated in Fig.5 (Left), when neither f_R nor f_λ is active, $V_1(\xi) = \xi^T\xi$. In this case, we plot the contour line of $V_1(\xi) = \xi^T\xi = 5$, which is represented by the red dashed line in the figure. When only f_λ is active, we set the parameters $m = 2$, $\alpha = [1, 1]$, $\beta = [0, 0]$, and $\omega_1 = [1, 1]$. Then, $V_1(\xi) = \lambda(\xi)^2\xi^T\xi$, where $\lambda(\xi) = 2\tanh(5)$. This is equivalent to replacing ξ with $\lambda(\xi)\xi$, meaning that the points on the contour line will radiate outward along the rays originating from the origin by a factor of $2\tanh(5)$, as shown by the green solid line. If we set ω_1 as a function of $\xi/\|\xi\|$, this implies that the degree of radiation varies in different directions, as shown by the blue solid line. As illustrated in Fig.5 (Right), when neither f_R nor f_λ is active, we plot the curves $\xi_2 = \xi_1$, $\xi_2 = 2\xi_1$, and $\xi_2 = 3\xi_1$, represented by the blue dashed line, green dashed line, and black dashed line, respectively. When only f_R is active, $f_R(\xi) = R\xi$, where R represents the rotation matrix. This is equivalent to replacing ξ with $R\xi$, so the points on the original dashed lines will be rotated. The rotated curves are shown by the blue solid line, green solid line, and black solid line, respectively. Here, the rotation angle $\theta = 0.1(\xi_1^2 + \xi_2^2)$. This means that points with the same polar radius will be rotated by the same amount.

C. Stability Proof

In this section, we present the proof of the overall system stability. From Section II, we have established that an appropriate Lyapunov function must satisfy Eq. (2) and Eq. (5). Therefore, the focus of this section is to demonstrate these two conditions. In fact, through the analysis in Section II, Eq. (2) can be simplified to:

$$\begin{cases} (a)\, V\left(\xi\right) = 0, \xi = 0 \\ (b)\, V\left(\xi\right) > 0, \forall \xi \neq 0 \\ (c)\, \frac{\partial V}{\partial \xi} \neq 0, \forall \xi \neq 0 \end{cases} \tag{18}$$

Proposition 1. The energy function $V(\xi)$ represented by $V\left(\xi\right) = z^T z = \left(f_R\left(f_\lambda\left(\xi\right)\right)\right)^T f_R\left(f_\lambda\left(\xi\right)\right)$ satisfies $V\left(\xi\right) = 0, \xi = 0$, $V\left(\xi\right) > 0, \forall \xi \neq 0$ and $\lim\limits_{\|\xi\| \to +\infty} V\left(\xi\right) = +\infty$.

Proof Firstly, $V(\xi)$ are semi-positive definite, which is obvious because our energy functions are derived from a quadratic form. By observing f_λ and f_R, it is straightforward to obtain $f_\lambda(0) = 0$ and $f_R(0) = 0$. For $V(\xi)$, by substituting $\xi = 0$, we can prove that $V(0) = 0$: $V\left(0\right) = z\left(0\right)^T z\left(0\right) = \left(f_\lambda\left(f_R\left(0\right)\right)\right)^T f_\lambda\left(f_R\left(0\right)\right)$. Therefore, $V(0) = 0$ hold if and only if $\xi = 0$. Since $\lim_{\xi \to \infty} f_\lambda\left(\xi\right) = \infty$, $\lim_{y \to \infty} f_R\left(y\right) = \infty$, it is clear that V is radially unbounded.

Proposition 2. The energy function satisfies the condition $\frac{\partial V}{\partial \xi} \neq 0, \forall \xi \neq 0$ if its satisfies the condition $\frac{\partial V}{\partial r_\xi} \neq 0$ where $r_\xi = \|\xi\|$ and $r_\xi \neq 0$.

Proof According to the chain rule of differentiation:

$$\frac{\partial V}{\partial \xi} = \frac{\partial V}{\partial r_\xi}\frac{\partial r_\xi}{\partial \xi} + \left(\frac{\partial \vec{r}_\xi}{\partial \xi}\right)^T \frac{\partial V}{\partial \vec{r}_\xi} \tag{19}$$

where $\vec{r}_\xi = \frac{\xi}{\|\xi\|}$. The variable definitions are as follows:

$$\begin{cases} \frac{\partial r_\xi}{\partial \xi} = \frac{\partial \|\xi\|_2}{\partial \xi} = \frac{\xi}{\|\xi\|_2} = \vec{r}_\xi \\ \frac{\partial \vec{r}_\xi}{\partial \xi} = \frac{1}{\|\xi\|_2} I - \frac{\xi\xi^T}{\|\xi\|_2^3} \end{cases} \tag{20}$$

Note that if $\xi \neq 0$, we have:

$$\frac{\partial \vec{r}_\xi}{\partial \xi}\frac{\partial r_\xi}{\partial \xi} = \left(\frac{1}{\|\xi\|_2} I - \frac{\xi\xi^T}{\|\xi\|_2^3}\right)\frac{\xi}{\|\xi\|_2} = 0 \tag{21}$$

In other words, for any $\xi \neq 0$, vector $\frac{\partial r_\xi}{\partial \xi}$ lies in the null space of matrix $\frac{\partial \vec{r}_\xi}{\partial \xi}$, hence, $\frac{\partial V}{\partial \xi} \neq 0$ if and only if $\left(\frac{\partial \vec{r}_\xi}{\partial \xi}\right)^T \frac{\partial V}{\partial \vec{r}_\xi} = 0$ and $\frac{\partial V}{\partial r_\xi}\frac{\partial r_\xi}{\partial \xi} = 0$. Therefore, we only need to ensure:

$$\frac{\partial V}{\partial r_\xi}\frac{\partial r_\xi}{\partial \xi} = \frac{\partial V}{\partial r_\xi}\vec{r}_\xi \neq 0 \tag{22}$$

As $\frac{\partial V}{\partial r_\xi}\frac{\partial r_\xi}{\partial \xi} = \frac{\partial V}{\partial r_\xi}\vec{r}_{\xi_\xi} \neq 0$ if and only if $\frac{\partial V}{\partial r_\xi} \neq 0$, thus $\frac{\partial V}{\partial r_\xi} \neq 0$ and $\frac{\partial V}{\partial \xi} \neq 0$ are equivalent. This implies that the gradient of any ray starting from the origin will not vanish.

Proposition 3. The energy function $V(\xi) = z^T z = (f_R(f_\lambda(\xi)))^T f_R(f_\lambda(\xi))$ has the property $\frac{\partial V}{\partial \xi} \neq 0$ if $\xi \neq 0$.

Proof According to Eq. (13), we can rewrite $V(\xi)$ as follows:

$$V(\xi) = (f_\lambda(y))^T f_\lambda(y) = \lambda^2(y) \cdot r_y^2 \tag{23}$$

According to the chain rule of differentiation, we have:

$$\frac{\partial V}{\partial r_y} = 2\lambda \frac{\partial \lambda}{\partial r_y} r_y^2 + 2\lambda^2 r_y > 0 \tag{24}$$

Consequently, we have $\frac{\partial V}{\partial r_y} \neq 0$. From Proposition 2, we can deduce that $\frac{\partial V}{\partial y} \neq 0$. Since $\frac{\partial y}{\partial \xi}$ is full rank, we have $\frac{\partial V}{\partial \xi} = \frac{\partial V}{\partial y}\frac{\partial y}{\partial \xi} \neq 0$. Therefore, we conclude that $\frac{\partial V}{\partial \xi} \neq 0$ if $\xi \neq 0$.

Based on the proof above, we can derive Eq. (3) and Eq. (11) for V. Together with the analysis in Section II, this leads to the conclusion that the dynamic system derived using Eq. (5) is stable.

IV. Experimental results

A. Parameter settings

In this section, we validate the effectiveness of the proposed algorithm on the LASA dataset. The LASA dataset comprises a collection of human handwriting data, which includes 26 examples, each consisting of 7 demonstrations. To improve learning efficiency, we downsample the dataset at a ratio of 1:5. We utilize Eq. (13) as the Lyapunov function. f_{λ_1}, f_{λ_2} are defined as multilayer perceptrons (MLPs) with two hidden layers, where each hidden layer contains 10 neurons and the output layer consists of 5 neurons (i.e., $m = 2$), and the output is activated by an exponential function. f_{λ_3} are defined as multilayer perceptrons (MLPs) with two hidden layers, where each hidden layer contains 10 neurons and the output layer consists of 5 neurons. Since there are only two dimensions, the angle modulation has only one function f_{θ_1}. f_{θ_1} is set as an MLP with two hidden layers, each containing 10 neurons. In Eq. (5), $o(\xi)$ is represented as an MLP with two hidden layers, where each hidden layer contains 200 neurons. We constructed the proposed structure using PyTorch and trained it using the Adam optimizer. The number of training epochs was set to 1000, with a learning rate of 0.01. For convenience in modifying the structure, we employed numerical derivatives for optimization. The loss function utilized the built-in MSELoss function from PyTorch.

B. Simulation results

The test results for REF are shown in Fig. 8. In this figure, the shading of the background color indicates the value of the learned energy function. The blue lines represent the predicted values of the learned dynamical system (DS). The black cross marks the target point, the white line indicates the expected trajectory, and the red line represents the trajectory reproduced according to the DS. It is evident that the reproduced trajectory closely aligns with the expected trajectory, demonstrating the effectiveness of our algorithm. Furthermore, the energy function value is higher at the beginning of the demonstration and decreases as it approaches the target value, supporting the correctness of our theory. Even for highly complex tasks such as"BendedLine," "DoubleBendedLine," and "Snake," our model has shown strong performance. Importantly, the same set of hyperparameters was used across all examples in the LASA dataset, indicating the broad applicability of our proposed model.

C. Comparison of different Energy functions

To better illustrate the advantages of REF, we compare the four groups of experiments. The four groups of experiments are trained with different energy functions: REF, NEUM, WASQF, and QF. Fig. 7 shows the comparison of the energy function metrics obtained from the experiments on six challenging tasks for each group:

$$E_1 = 1 + \frac{1}{LN}\sum_{l=1}^{L}\sum_{n=1}^{N} \frac{\frac{\partial V(\xi_{l,n})}{\partial \xi_{t,n}}\dot{\xi}_{l,n}}{\|\xi_{l,n}\|_2 \|\frac{\partial V(\xi_{l,n})}{\partial \xi_{l,n}}\|_2} \tag{25}$$

$$E_2 = 1 + \frac{1}{LN}\sum_{l=1}^{L}\sum_{n=1}^{N} \tanh\left(\frac{10 \cdot \frac{\partial V(\xi_{l,n})}{\partial \xi_{t,n}}\dot{\xi}_{l,n}}{\|\xi_{l,n}\|_2 \|\frac{\partial V(\xi_{l,n})}{\partial \xi_{l,n}}\|_2}\right) \tag{26}$$

The results obtained by testing with the above metrics are shown in the figure. It can be observed that, regardless of the metric used for testing, the structure proposed in this paper is the optimal one. To better illustrate the differences between REF and NEUM, we collect a specific type of trajectory that performs poorly with the NEUM energy function, while a balance between accuracy and stability is achieved with the REF energy function. The results of the experiment are shown in Fig. 9.

D. Algorithm comparison

We primarily compare our method with two different approaches, namely CLF-DM [17] and NEUM [4]. In this paper, we select two evaluation indicators to assess the proposed system: The Swept Error Area (SEA) and the Velocity Root Mean Square Error (V_{rmse}). According to the guidance from [17] and [4], the number of hidden layer neurons in NEUM is set to 10 for the six most challenging tasks, and to 2 for the remaining tasks. The $\mathcal{L}$ in CLF-DM is set to 10. The results are shown in the Table III. It can be seen that our algorithm achieves better results than other algorithms in the LASA dataset.

E. Robotics experiments

In addition to simulation experiments, we also conducted experiments on the actual robot. Our experiment consists of two steps, as shown in Fig. 10. Firstly, we conduct robot demonstrations through the robot's teaching mode, recording the positions and velocities of the robot's movements. Then, we learn the parameters representing the dynamic system using the aforementioned method. The robot we used is the kinova, which is a seven-axis collaborative robotic arm. Our experimental setup is shown in the figure. We conducted a total of two experiments, the first being an ultrasound

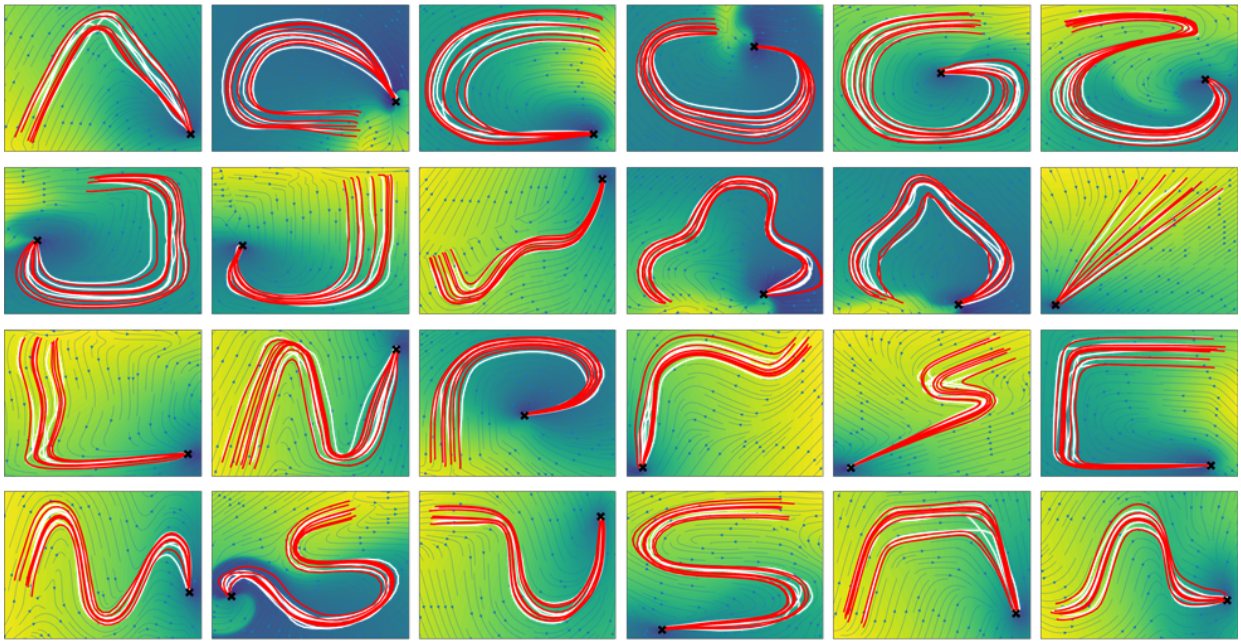

Fig. 6. This is a test using REF and FCNN on the LASA handwritten dataset. The red trajectory is the replicated trajectory, and the white trajectory is the expected trajectory. It can be seen that the learned DS can capture the characteristics of the demonstration very well. The background color is drawn according to the energy function value from high to low.

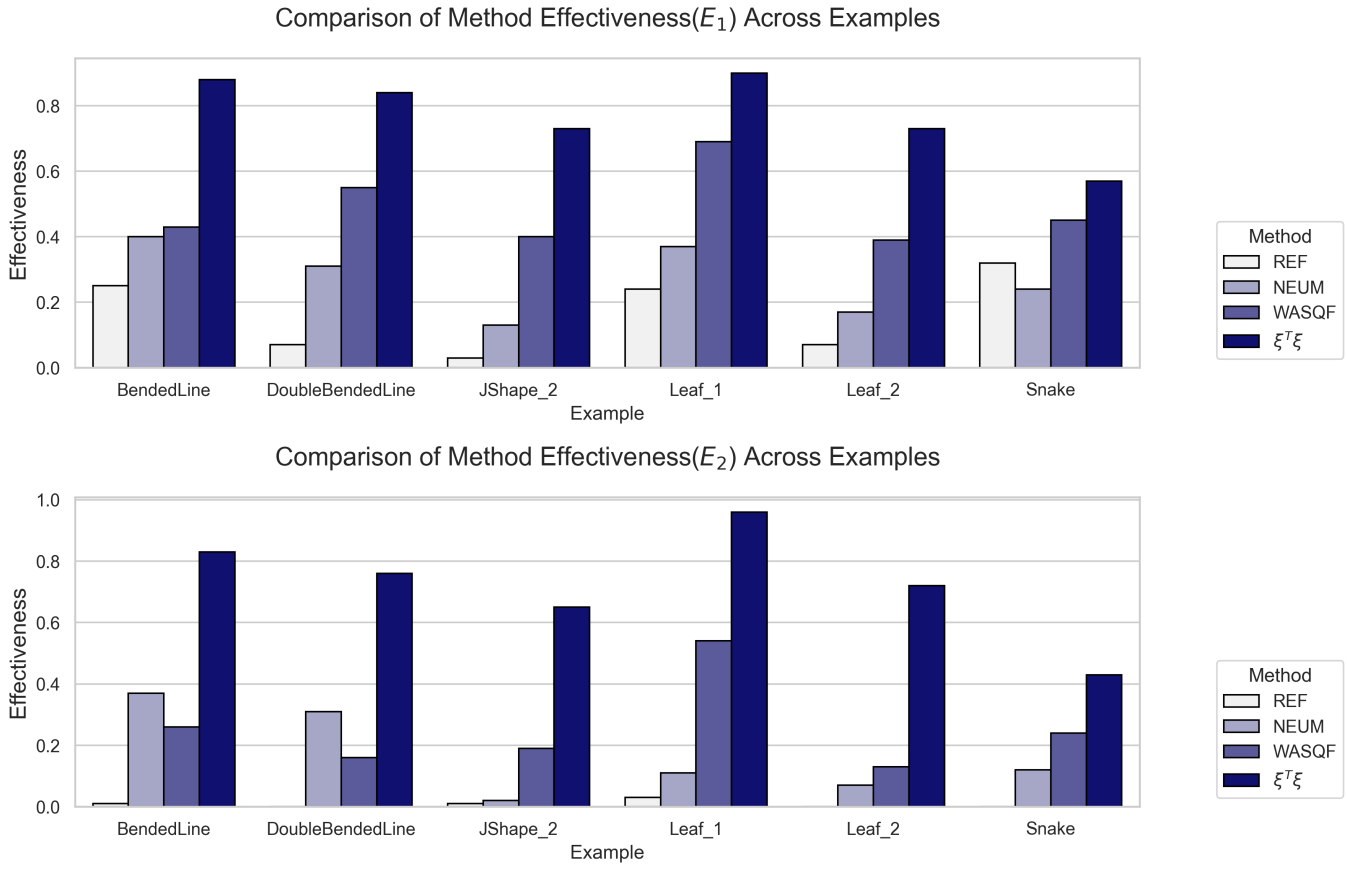

Fig. 7. The optimal results achieved by optimizing the parameters of REF using different objective functions.

scanning experiment, which consisted of two groups: longitudinal scanning for the first group and transverse scanning for the second group. The second experiment was an assembly experiment, also divided into two groups based on the height

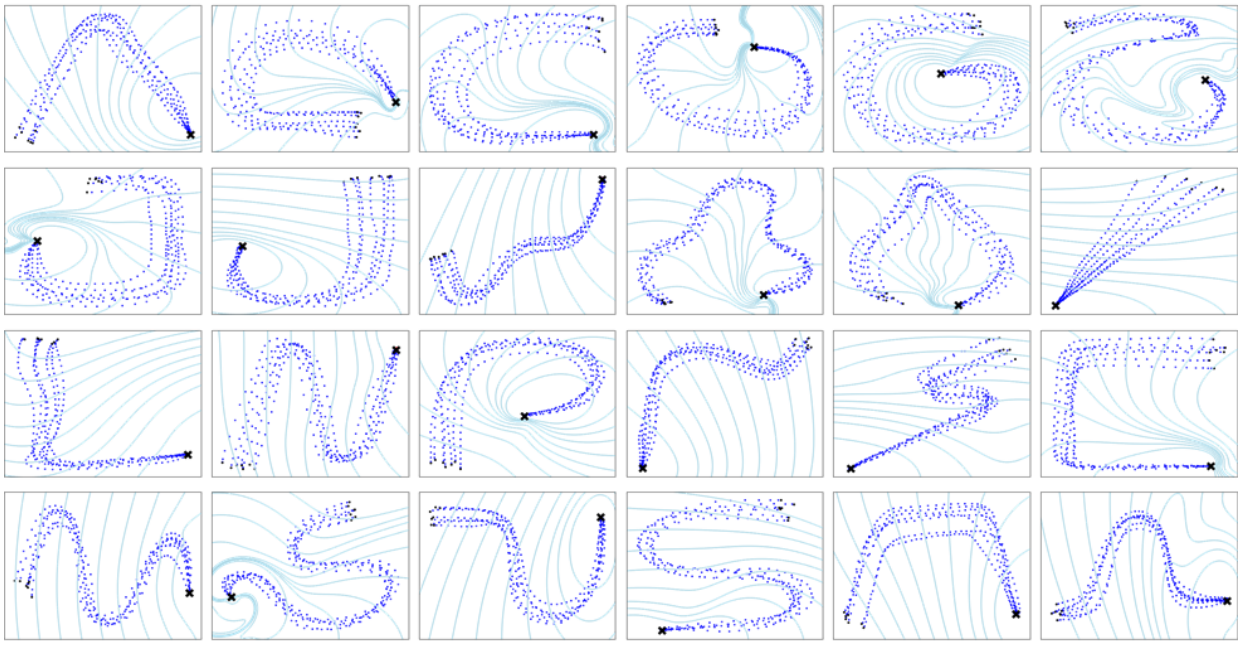

Fig. 8. This is a test using REF and FCNN on the LASA handwritten dataset. The background depicts the equipotential line levels of the energy function learned using REF.

TABLE II
THIS IS THE PERFORMANCE OF DIFFERENT ALGORITHMS ON SIX OF THE MOST CHALLENGING TASKS.

Cases	SEA			VMSE		
	REF	NEUM	WASQF	REF	NEUM	WASQF
BendedLine	59.96(40.63)	211.02(127.24)	59.32(42.01)	2.06(0.69)	2.05(0.59)	1.70(0.60)
DoubleBendedLine	71.22(45.19)	180.53(107.18)	67.76(35.99)	1.51(0.17)	1.51(0.17)	1.41(0.12)
JShape_2	156.56(65.12)	152.20(66.67)	160.96(81.40)	4.34(0.26)	3.63(0.44)	3.76(0.46)
Leaf_1	130.61(48.77)	131.84(81.81)	142.39(53.36)	2.73(0.31)	2.31(0.24)	2.21(0.26)
Leaf_2	74.51(17.52)	80.54(30.84)	87.90(31.73)	2.75(0.42)	2.63(0.33)	2.64(0.33)
Snake	125.61(51.82)	118.58(34.62)	303.27(246.03)	2.46(0.27)	2.28(0.23)	2.53(0.31)

TABLE III
COMPARISON OF SEA AND V_{rmse} INDICATORS OF DIFFERENT ALGORITHMS ON ALL TRAJECTORIES.

Learning method	MEAN-SEA	MEAN-V_{rmse}
REF	71.30[13.90-507.5]	1.44[1.27-10.81]
NEUM	78.43[2.10-754.43]	1.24[1.21-9.17]
WASQF	81.80[5.00-881.44]	1.33[1.2-9.37]

of the trajectories. The experimental results are shown in the figure. According to the results, whether in the ultrasound scanning experiment or the assembly experiment, our reproduced trajectories can closely match the taught trajectories. Additionally, we observed that the learned dynamic system still demonstrates a certain level of generalization performance even in positions without teaching. Furthermore, the energy functions also align with the flow direction of the trajectories. More importantly, our model parameters are consistent with those tested on the LASA dataset. Except for changing the input dimension to 3, we did not alter any other parameters, demonstrating the generalizability of our model.

V. CONCLUSION

This paper introduces a new energy function, REF, and constructs a stable dynamic system using this function. Our Lyapunov function offers a clear geometric interpretation. Furthermore, we demonstrate that our function is more general than existing alternatives. From an experimental perspective, our structure achieves favorable results on metrics such as SEA and VMSE in the LASA dataset. Unlike other models, ours is trained in a single step, providing complementarity among various components compared to multi-step training methods. Additionally, real-world experiments on actual robots confirm the applicability of our model in ultrasound scanning and assembly tasks. Overall, our approach strikes a balance between training accuracy and stability.

REFERENCES

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.
[2] H. Ochoa and R. Cortesão, "Impedance control architecture for robotic-assisted mold polishing based on human demonstration," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 4, pp. 3822–3830, 2021.

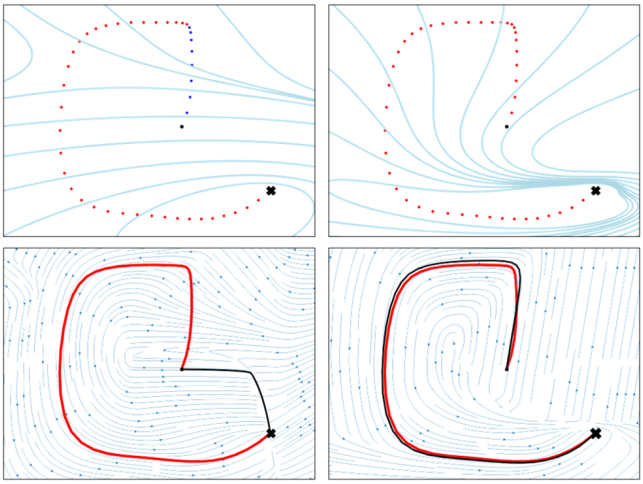

Fig. 9. The upper left figure displays the energy function learned using the NEUM approach, while the upper right figure shows the energy function learned with the REF method. The red dots represent points that satisfy the gradient constraint of the energy function, whereas the blue dots indicate points that do not. The lower left figure illustrates the trajectory reproduction effect learned through the NEUM method, while the lower right figure depicts the trajectory reproduction effect learned with the REF method. In these figures, the red line represents the teaching trajectory, and the black line represents the reproduced trajectory.

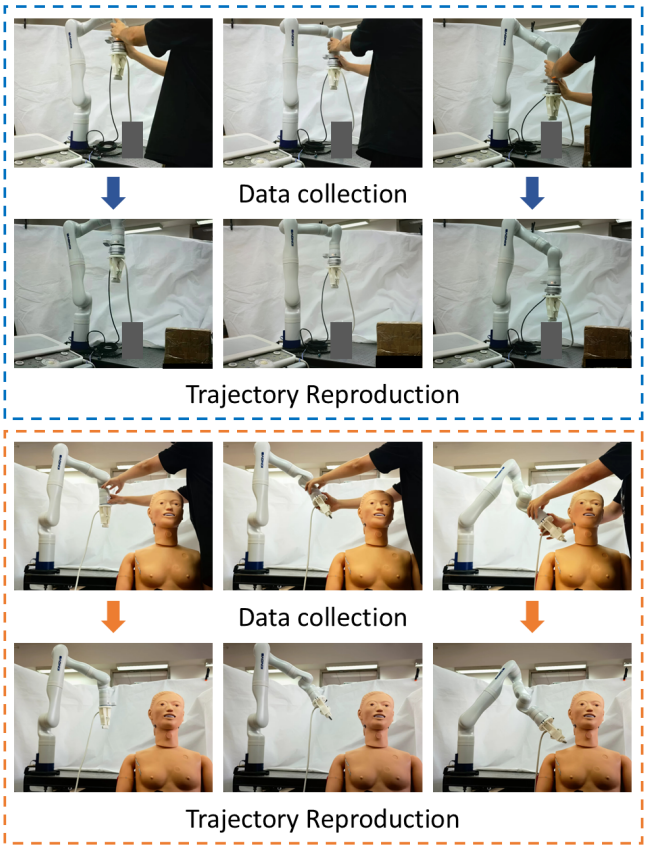

Fig. 10. This is a comic diagram of the proposed algorithm running on an actual robot. We completed two experiments on the actual robot: assembly and ultrasonic detection.

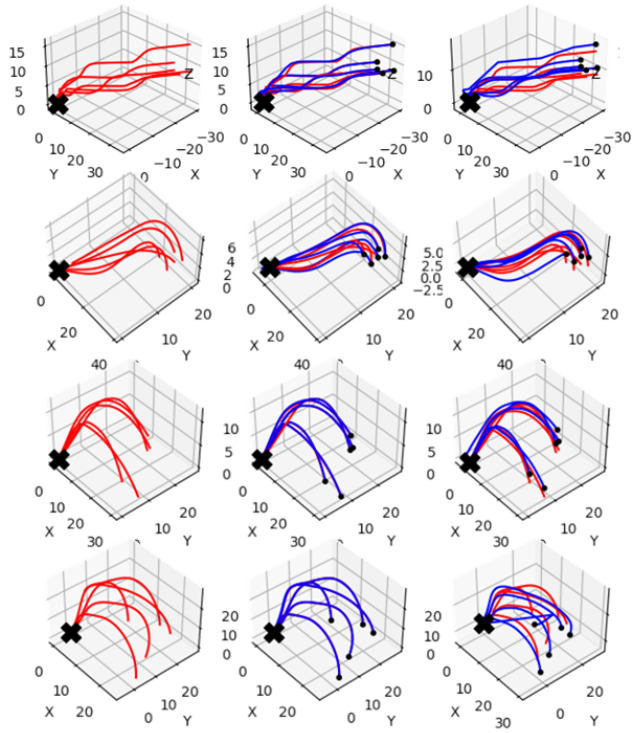

Fig. 11. These are four experiments we conducted on the Rokae robot, listed from top to bottom as two ultrasound robot experiments followed by two assembly experiments. From left to right, the four columns represent: (1) Collected trajectories, (2) Reproduced trajectories, and (3) Generalized trajectories

[3] Y. Zhang, L. Cheng, H. Li, and R. Cao, "Learning accurate and stable point-to-point motions: A dynamic system approach," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1510–1517, 2022.

[4] Z. Jin, W. Si, A. Liu, W.-A. Zhang, L. Yu, and C. Yang, "Learning a flexible neural energy function with a unique minimum for globally stable and accurate demonstration learning," *IEEE Transactions on Robotics*, 2023.

[5] Y. Ma, D. Xu, and F. Qin, "Efficient insertion control for precision assembly based on demonstration learning and reinforcement learning," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4492–4502, 2020.

[6] M. Brin and G. Stuck, *Introduction to dynamical systems*. Cambridge university press, 2002.

[7] M. Hersch, F. Guenter, S. Calinon, and A. Billard, "Dynamical system modulation for robot learning via kinesthetic demonstrations," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1463–1467, 2008.

[8] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[9] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3339–3344, IEEE, 2014.

[10] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning physical collaborative robot behaviors from human demonstrations," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 513–527, 2016.

[11] J. Kocijan, *Modelling and control of dynamic systems using Gaussian process models*. Springer, 2016.

[12] R. Kruse, S. Mostaghim, C. Borgelt, C. Braune, and M. Steinbrecher, "Multi-layer perceptrons," in *Computational intelligence: a methodological introduction*, pp. 53–124, Springer, 2022.

[13] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on convolutional neural networks (cnn) in vegetation remote sensing," *ISPRS journal of photogrammetry and remote sensing*, vol. 173, pp. 24–49, 2021.

[14] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[15] Y. Hristov, D. Angelov, M. Burke, A. Lascarides, and S. Ramamoorthy, "Disentangled relational representations for explaining and learning from

demonstration," in *Conference on Robot Learning*, pp. 870–884, PMLR, 2020.
[16] R. Pérez-Dattari and J. Kober, "Stable motion primitives via imitation and contrastive learning," *IEEE Transactions on Robotics*, 2023.
[17] S. M. Khansari-Zadeh and A. Billard, "Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.
[18] M. Saveriano, "An energy-based approach to ensure the stability of learned dynamical systems," in *2020 IEEE International conference on robotics and automation (ICRA)*, pp. 4407–4413, IEEE, 2020.
[19] M. Saveriano and D. Lee, "Incremental skill learning of stable dynamical systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6574–6581, IEEE, 2018.
[20] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *The International Journal of Robotics Research*, vol. 42, no. 13, pp. 1133–1184, 2023.
[21] D.-H. Park, H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoids 2008-8th IEEE-RAS international conference on humanoid robots*, pp. 91–98, IEEE, 2008.
[22] A. Hewitt, C. Yang, Y. Li, and R. Cui, "Dmp and gmr based teaching by demonstration for a kuka lbr robot," in *2017 23rd International Conference on Automation and Computing (ICAC)*, pp. 1–6, IEEE, 2017.
[23] E. Gribovskaya and A. Billard, "Learning nonlinear multi-variate motion dynamics for real-time position and orientation control of robotic manipulators," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pp. 472–477, IEEE, 2009.
[24] E. Gribovskaya and A. Billard, "Learning nonlinear multi-variate motion dynamics for real-time position and orientation control of robotic manipulators," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pp. 472–477, IEEE, 2009.
[25] J. Umlauft, A. Lederer, and S. Hirche, "Learning stable gaussian process state space models," in *2017 American Control Conference (ACC)*, pp. 1499–1504, IEEE, 2017.
[26] J. Z. Kolter and G. Manek, "Learning stable deep dynamics models," *Advances in neural information processing systems*, vol. 32, 2019.
[27] A. Abyaneh, M. S. Guzmán, and H.-C. Lin, "Globally stable neural imitation policies," *arXiv preprint arXiv:2403.04118*, 2024.

Haohui Huang received the Ph.D. degree in Control Science and Engineering from South China University of Technology, Guangzhou, China, in 2020. He was a postdoctoral with School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, from 2020 to 2022. He is currently a lecture with the School of Automation, Guangdong University of Technology, Guangzhou, China. His research interests include intelligent control, imitation learning, and embodied intelligence.

Wenxu Zheng is currently working toward the B.E. degree in the School of Automation, Guangdong University of Technology, Guangzhou, China. His research interests include motion planning, compliant control and deep learning algorithms.

Junda Duan received the Bachelor's degree in Automation from Jiujiang University, Jiujiang, China, in 2024. He is currently a graduate student with the School of Automation, Guangdong University of Technology, Guangzhou, China. His research interests include robot imitation learning, motion control, and intelligent control.

Kailun Huang is currently pursuing the B.S. degree in Automation from Guangdong University of Technology, Guangzhou, China. His research interests include imitation learning, and embodied intelligence.

Jing Guo received the B.S. and M.S. degrees from the Guangdong University of Technology, in 2009 and 2012, respectively, and the Ph.D. degree from LIRMM, CNRS-University of Montpelier, France, in 2016. He was a Research Fellow with the National University of Singapore (NUS), from 2016 to 2018. He is an Associate Professor with the Guangdong University of Technology. His current research interests include robotic control and learning, haptic bilateral teleoperation, and surgical robotics. He has served as a Guest Editor for IEEE ROBOTICS AND AUTOMATION LETTERS and Frontiers in Robotics and AI.

Chenguang Yang (Fellow, IEEE) received the B.Eng. degree in measurement and control from Northwestern Polytechnical University, Xian, China, in 2005, and the Ph.D. degree in control engineering from the National University of Singapore, Singapore, in 2010. He performed postdoctoral studies in human robotics at the Imperial College London, London, U.K from 2009 to 2010. Professor Yang holds the Chair in Robotics with Department of Computer Science, University of Liverpool, UK. He is leading the Robotics and Autonomous Systems Group. He was leading Robot Teleoperation Group at Bristol Robotics Laboratory. He is a member of European Academy of Sciences and Arts (EASA). He was awarded UK EPSRC UKRI Innovation Fellowship and individual EU Marie Curie International Incoming Fellowship. He was President of Chinese Automation and Computing Society in the UK. He is the Corresponding Co-Chair of IEEE Technical Committee on Collaborative Automation for Flexible Manufacturing. As the lead author, he won the IEEE Transactions on Robotics Best Paper Award (2012) and IEEE Transactions on Neural Networks and Learning Systems Outstanding Paper Award (2022). His research interest lies in robot control and learning, human robot interaction and intelligent system design.