A Survey on Learning an Autonomous Dynamic System for Human-Robot Skills Transfer from Demonstration

https://doi.org/10.1016/j.rcim.2025.103092

A Survey on Learning an Autonomous Dynamic System for Human-Robot Skills Transfer from Demonstration

Jiayun Fu[a,1], Haotian Huang[a], Zhehao Jin[b], Andong Liu[a,*], Wen-An Zhang[a], Li Yu[a], Weiyong Si[c] and Chenguang Yang[d]

[a]*The College of Information Engineering, Zhejiang University of Technology, Hangzhou, 310032 China*
[b]*The School of Mechanical and Aerospace Engineering, Nanyang Technological University, 639798 Singapore*
[c]*The School of Computer Science and Electronic Engineering, University of Essex, CO4 3SQ Colchester, U.K*
[d]*The Department of Computer Science, University of Liverpool, L69 3BX Liverpool, U.K*

ARTICLE INFO

Keywords:
Imitation Learning
Human-Robot Skills Transfer
Dynamic system
Robot learning
Lyapunov function

ABSTRACT

Autonomous dynamic systems (ADS) have become a key area of research in the field of robotics, aiming to enable robots to acquire human-like operational skills and perform complex tasks in dynamic environments without external intervention. Despite significant progress, current technologies have yet to enable robots to fully achieve autonomous skill transfer in real-world applications. The prevailing approach to bridge this gap is Learning from Demonstration (LfD), where robots learn by observing and imitating expert demonstrations. Dynamic systems-based methods, particularly those utilizing Lyapunov stability theory, have shown great potential in effectively encoding human motor skills, ensuring the stability, accuracy, and generalization of learned behaviors during the learning process. This survey provides an overview of the recent advancements in dynamic systems for skill transfer, focusing on methods that enable robots to replicate human actions, as demonstrated by experts. We present a classification of existing dynamic systems approaches, highlight landmark studies, and discuss their key features, advantages, and limitations. This paper also explores the applications of these methods and identifies major challenges that remain in both theoretical and practical aspects of robot skill learning.

1. Introduction

In the context of robotics and manufacturing automation, enabling robots to autonomously execute complex tasks with human-like precision and adaptability is a central objective. Recent advancements in robotic technologies have expanded their application to various dynamic and unstructured environments, including homes [1], offices [2, 3] and hospitals [4]. As the demand for robots to perform increasingly sophisticated and human-like tasks rises, Learning from Demonstration (LfD) has emerged as a key strategy for achieving this goal. LfD enables robots to acquire new skills by observing and imitating expert demonstrations [5, 6], offering significant potential for enhancing operational flexibility, efficiency, and seamless integration of robots into diverse industrial environments.

Encoding human motion skills through dynamic systems (DS) learning has been widely demonstrated to be an effective method for capturing human actions. This approach not only describes and models complex motion patterns but also enhances the adaptability and flexibility of robots in variable environments. These systems mainly leverage Lyapunov stability theory (or Contraction and Koopman theory [7, 8]) to ensure the integration of accuracy, stability, and generalization during the learning process. If it is shown to be strictly decreasing along the trajectory of the system, the stability of the system with respect to a fixed point [9] (or differential stability in trajectory tracking [10, 11]) is proven. An intuitive explanation of the dynamic system is shown in Fig. 1.

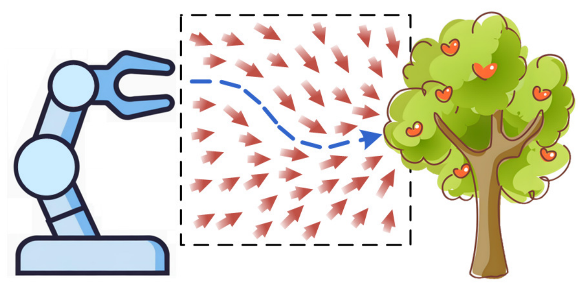

Figure 1: The intuitive understanding of dynamic systems. The blue trajectory shows the movement of the robot's end effector when starting from its current state. The evolution of this trajectory is defined by the DS (1), which is illustrated with a vector field of red arrows in the rest of the space.

The design of these certificate functions is highly valuable for control system designers, as they can prove that even complex and nonlinear control systems can maintain safety and stability. Although certificate theories like the Lyapunov stability theory have existed for over a century, general numerical methods for constructing certificates have only emerged in the past decade or so. Even then, many of the proposed methods are still computationally challenging(e.g., relying on solving high-dimensional partial differential equations (PDEs) numerically [9] and solving high-order dynamic systems [12–14]). Without efficient and

*Corresponding author
lad@zjut.edu.cn (A. Liu)
ORCID(s): 0000-0003-2445-8457 (A. Liu)
[1]First author

general methods, learning Lyapunov functions requires considerable effort in manually designing them for specific dynamic systems. Even in the best case, this manual tuning requires a great deal of intuition and luck to determine the appropriate function form (e.g., a polynomial of fixed degree) and parameters (e.g., polynomial coefficients) for the Lyapunov function.

Precisely, the key to learning dynamic systems for encoding human motion skills lies in finding a Lyapunov function that is both safe and stable to guide the learning process. In recent years, various new techniques for automatically learning Lyapunov functions have emerged. For some simple linear systems, Lyapunov functions can be constructed using Sums of Squares (SOS) methods [12]. For slightly more complex nonlinear systems, approaches such as weighted sum of asymmetric quadratic functions (WSAQF) [15] or simple feedforward networks (Extreme Learning Machines [16]) can approximate Lyapunov functions. However, these methods often have limitations and are challenging to extend to more complex or higher-dimensional systems. To address these challenges, research in control theory, machine learning, and robotics has increasingly turned to neural networks for learning and approximating Lyapunov functions in recent years [17–19]. However, the challenge of designing Lyapunov functions is not the only obstacle in complex robotic dynamic systems. As the field progresses, research is increasingly focusing on demonstration-based learning methods, seeking to generate verifiable control strategies alongside the learned Lyapunov stability certificates. These methods not only aim to capture expert demonstration behaviors but also ensure that the learned dynamic systems possess stability and adaptability.

In recent years, research on ADS has gained widespread attention and made significant progress, but no comprehensive survey has yet covered the latest developments in learning ADS from demonstrations. Ravichandar (2020) [20] provides an overview in his review of machine learning methods that enable robots to learn from and imitate experts, with a primary focus on behavior cloning (BC) and reinforcement learning (RL) [21]. The review (2023) [22] discusses control Lyapunov functions. Another review (2023) [23] provides an overview of movement primitives (MP) and Experience Abstraction (EA) but does not cover methods related to ADS. Our goal is to provide a comprehensive review of the latest developments in learning stable dynamic systems from demonstrations, offering a concise introduction for both practitioners seeking to apply these tools to real-world robotic problems and scholars. We primarily focus on two types of methods for learning dynamic systems from demonstrations: those based on Lyapunov functions and those based on diffeomorphisms, while also discussing research on second-order dynamic systems. To provide a clearer overview, this review is organized as follows:

1) Section II provides the relevant background of the dynamic system, including the definition and stability.

2) Section III categorizes ADS methods according to their implementation, reviews the representative methods, and concludes the latest research progress in higher-order ADS. Furthermore, it analyzes the relationship between the quality of DS generation and the learned Lyapunov function.

3) Section IV explores various robot application areas as well as methodological tradeoffs. Section V concludes by discussing some open challenges and future directions in this area.

4) Finally, for ease of reproduction and further research, we have compiled the relevant implementation code available on GitHub[2].

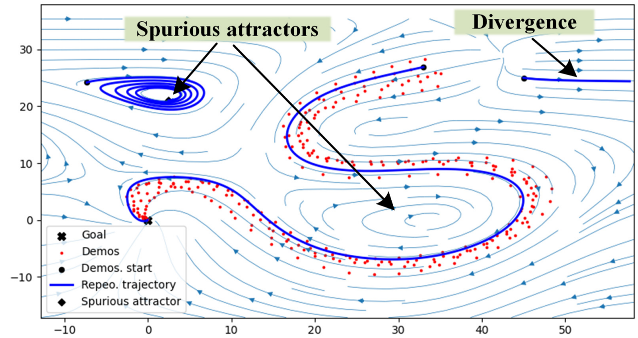

Figure 2: The original DS learned from 2-dimensional LASA dataset [24] using regression such as Gaussian processes regression (GPR) [25], and so on [26–28]. However, directly constructing the DS through regression techniques often leads to the emergence of spurious attractors and system divergence.

2. System Models and Problem Description

This section will first introduce the concepts of dynamic systems, system stability definition, second-order dynamic systems, and evaluation metrics.

2.1. Dynamic system

Consider a system with the following dynamics:

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

where $x \in X \subset \mathbb{R}^n$ is the state vector (X is the set of all allowable x), and $f(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is a continuous function. Here, $x(t)$ is for short as x in convenience. It is an autonomous system since $f(x)$ is only related to x, rather than time. The stability of system (1) indicates that x asymptotically approaches a fixed point, which is also referred to as an equilibrium point x^*.

It is worth noting that not all dynamic systems are stable. The most common scenario is that a system may be inherently unstable, as illustrated in Fig. 2, but it can be controlled to achieve stability. This situation is typically modeled as:

$$\dot{x} = g(x) = f(x) + u(x) \tag{2}$$

where $f(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $u(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is relate to x.

[2]Most of the code for the state-of-art approach has been made publicly available to us at https://github.com/fjyggg/review/tree/master/.

In the context of these dynamics, the task of control system designers is to find a control law $u(x)$, or directly learn a stable dynamic system $g(x)$, so that the learned dynamic system $\dot{x} = f_c(x)$ exhibits the desired properties (e.g., stability). In this paper, both $u(x)$ or $g(x)$ rely on the learned Lyapunov function $V(x)$, and $V(x)$ will guide the behavior of the system to achieve goals such as stability, safety, or generalization. Next, we will define these goals in turn and illustrate how they can be achieved through Lyapunov functions.

2.2. Stability of dynamic systems

In this paper, we consider global asymptotic stability (GAS) because, in real-world robotics end-to-end applications, it is often expected that the system has a unique goal direction, which implies that an equilibrium point exists and that the system can be ensured to evolve towards that goal by imposing stability constraints.

Compared to Eq. (2), GAS provides a more intuitive definition. According to Lyapunov stability theory and LaSalle's invariance principle [29], GAS [30] is defined as follows:

Theorem 1: *Consider the dynamic system $\dot{x} = f(x)$ with equilibrium point x^*. Suppose there exists a positive-definite, radially unbounded, and continuously differentiable Lyapunov function $V(x)$ defined on $\mathbb{R}^n$ that satisfies:*

$$\begin{cases} V(x) = 0 \quad \Leftrightarrow \quad x = x^*, \\ V(x) > 0 \quad \Leftrightarrow \quad \forall x \in X \setminus \{x^*\}, \\ \dfrac{dV(x)}{dt} \leqslant 0 \quad \Leftrightarrow \quad \forall x \in X. \end{cases} \tag{3}$$

*Let $S = \{x \in \mathbb{R}^n | \dot{V}(x) = 0\}$. If the only solution that remains identically in S is $x(t) \equiv x^*$ (i.e., only the equilibrium point x * is an invariant set), then the equilibrium x^* is GAS.*

The proofs can be found in most control textbooks and are omitted here, but the primary insights are: 1) if V is monotonically decreasing and bounded below, then it must eventually approach its minimum value at 0; 2) the goal of $\frac{dV(x)}{dt} \leqslant 0$ is to make the learned Lyapunov function $V(x)$ consistent with the demonstrated preference that the demonstrated trajectory evolves from the region of high-value of $V(x)$ to its low-value region.

In dynamic system approaches, the focus is generally on teaching robots goal-directed motion skills. Most robotic tasks can be formulated as goal-directed motions or combinations of different goal-directed motions. To ensure consistency with goal-directed movements, it is necessary to guarantee that ADS $\dot{x} = f(x)$ must be GAS.

Recalling from Eq. (3), any Lipschitz strategy that selects or learns control inputs from these sets will necessarily stabilize the system. Since these conditions are affine in u, a common approach is to choose quadratic programming (QP), which searches for the minimum corrective control such that u satisfy $\frac{dV(x)}{dt} \leqslant 0$, for example

$$\begin{aligned} &\min_u \; u^T u \\ &s.t. \left(\frac{\partial V(x)}{\partial x}\right)^T (f(x) + u) \leqslant -\rho(x) \end{aligned} \tag{4}$$

where $V(x)$ is the Lyapunov functions, $\rho(x)$ is any positive-definite functions. Some common choices for $\rho(x)$ can be found in [15, 31].

Another common approach is to directly estimate the stable ADS $\dot{x} = g(x)$, which can be formulated as the following optimization problem:

$$\begin{aligned} &\min_\theta J(\theta) = \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left\| g(x_{t,n}, \theta) - \dot{x}_{t,n} \right\|_2^2 \\ &s.t. \left(\frac{\partial V(x)}{\partial x}\right)^T g(x, \theta) \leqslant -\rho(x) \end{aligned} \tag{5}$$

where θ is the learnable parameter of the ADS $\dot{x} = g(x)$.

Both methods (4) or (5) introduce constraints related to the Lyapunov function to ensure convergence. Specifically, these constraints dictate that the trajectory of the learned ADS $g(x)$ must transition from a high-energy region to a low-energy region. Therefore, the choice of the Lyapunov function $V(x)$ can greatly affect the quality of the learned ADS. This leads to the crucial question of how to construct suitable Lyapunov functions. In Section 3, we will categorize the ADS methods according to the different ways of constructing Lyapunov functions.

2.3. Second-order dynamic system

In our previous discussion of dynamic system and stability, we assumed that dynamic system is first-order. However, in many applications, most motion systems are of the second order, such as closed-loop robotic position tracking systems and even the more complex human motion systems. This often necessitates modeling the second-order dynamic systems as follows:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = f_1\left(\begin{bmatrix} x \\ \dot{x} \end{bmatrix}\right) \quad or \quad \ddot{x} = f_2\left(\begin{bmatrix} x \\ \dot{x} \end{bmatrix}\right) \tag{6}$$

The left-hand side of Eq. (7) resembles the brute-force modeling method described in [12–14], and attempts to apply a first-order dynamic system approach for learning. This approach is fundamentally inadequate as it neglects the crucial interplay between the state variables and their higher derivatives. It is important to note that this modeling approach may encounter limitations in task execution, particularly in its implementation of trajectory intersection tasks, potentially leading to incomplete task fulfillment [14]. Another modeling approach has not yet been fully explored, although there is a proposal for feedback linearization [32] to implement it. We will introduce some existing work on second-order dynamic systems in Section 3.4.

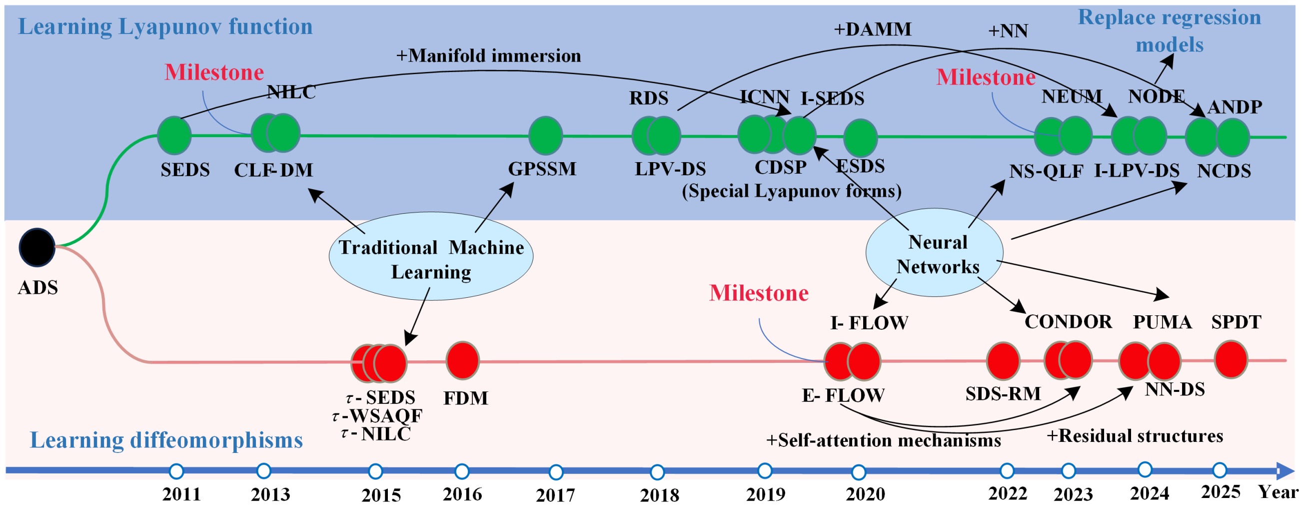

Figure 3: Historical timeline of ADS research in LfD and ADS methodology classification. The contraction metric is regarded as a special type of Lyapunov function, so methods (CDSP [8], NCDS [36]) related to contraction are classified into the first category. Method supplement not abbreviated in the paper, ESDS [37], I-SEDS [38, 39], NN-DS [40], PUMA [41].

2.4. Evaluation metrics

In practice, demonstration data are often obtained through expert demonstrations and consist of sampled trajectories. For dynamic systems requiring velocity or acceleration information, these quantities can typically be inferred or estimated from the trajectory data. One source of benchmark data is the LASA dataset [24] from Khansari-Zadeh, which includes two-dimensional trajectories describing handwriting patterns, as well as Sayantan Auddy's HELLO WORLD dataset [33] and second-order LAIR handwritten dataset [14]. These benchmark data are often used to evaluate the accuracy and effectiveness of the algorithms.

Several common metrics are used to evaluate the similarity between dynamic system trajectories. Root Mean Square Error (RMSE) is a widely employed measure of accuracy, providing a simple quantitative assessment of the difference between predicted and actual values. In certain contexts, such as two-dimensional curve comparisons, the Swept Error Area (SEA) [15] is often used. A more versatile approach is Dynamic Time Warping Distance (DTWD) [34], which is particularly useful for handling nonlinear temporal distortions. Another, less commonly applied, method for evaluating dynamic systems is the Fréchet Distance (FD) [35], which measures the similarity between two trajectories by considering the continuous matching of points along their paths.

3. Classification and Introduction of ADS

In this section, we discuss the two main classes of methods for learning ADS from demonstrations, categorized by the construction of Lyapunov functions: the direct method (learning Lyapunov functions) (Section 3.1) and the indirect method (learning diffeomorphisms) (Section 3.2), and present their previous work and recent progress. A focus is made on the landmark methods (Section 3.1.1 and 3.2.1), and some simulation examples and comparisons are given (Section 3.1.2 and 3.2.2). Additionally, we analyze the relationship between Lyapunov generation efficiency and DS quality (Section 3.3). Finally, we discuss advances in more complex scenarios, including dynamic systems with intersections and second-order dynamic systems (Section 3.4). The aim is to emphasize the development of the latest techniques in ADS. The timeline and classification of ADS development are shown in Fig. 3.

3.1. Learning Lyapunov functions methods

As previously mentioned, the first category of DS methods optimizes the original DS by introducing Lyapunov functions to achieve the desired optimization goal. The core of this approach lies in learning Lyapunov functions that are consistent with expert demonstrations and guide the learning process of the ADS.

An early method is the Stable Estimator of Dynamic Systems (SEDS) [12], which assumes that the demonstrations conform to a Lyapunov (or energy) function defined by the squared distance to the target, thus being restricted to motions that monotonically converge to the target over time. Under strict stability constraints, the precision of learning motion modeling from demonstrations is low, leading to a dilemma between precision and stability, known as a trade-off between precision and stability. The Control Lyapunov Function-based Dynamic Movements (CLF-DM) proposed by Khansari-Zadeh et al. [15] relaxes the requirement of a stability criterion by assuming the Lyapunov function in the form of WSAQF. The parameters of the Lyapunov function are learned independently of the (unstable) dynamics and subsequently used online to generate stability controls [42]. Matteo et al. introduced Reshaping Dynamic Systems (RDS) [37, 43], which can alter the trajectory of a dynamic system to follow the demonstrated trajectory while retaining its final stability properties. However, since this process is incremental, it implies a higher computational burden with each retrieval training iteration. Figueroa et al. proposed the linear variable parameter dynamic system (LPV-DS) type and its learning framework based on Gaussian mixture models (GMM) with good convergence to complex nonlinearities, but with low model accuracy and low online computational efficiency [44, 45]. To address the above challenges, sun et al. introduced the Directionality-Aware Mixture Model (DAMM) and demonstrated that LPV-DS integrated with DAMM can achieve higher reproduction accuracy, better computational efficiency, and near real-time/online learning [46]. Harish et al. proposed Contracting Dynamic System Primitives (CDSP) to ensure incremental stability through an updated approach [8]. CDSP introduces positive definite contraction metrics as a special Lyapunov function instead of directly learning Lyapunov functions. However, CDSP limits the category of contraction measures to a polynomial sum of squares [36]. Jonas et al. proposed a Gaussian Process State Space Model (GPSSM) that learns under stability constraints, enforcing convergence through a data-driven Lyapunov function [47, 48]. Despite the advances in the traditional methods described above, their limitations in global stability and expressive power remain significant.

With the rapid development of neural network technology, researchers have begun to explore new methods of learning Lyapunov functions using neural networks, aiming to address the limitations of traditional approaches in terms of global stability and expressive power. Among these efforts, Andre et al. proposed learning Lyapunov functions using Extreme Learning Machines (ELM) [16, 49, 50]. However, their use of sampling-based methods only provides approximate solutions to the problem of global asymptotic stability, with no theoretical guarantees. To overcome this limitation, in [18], an input convex neural network (ICNN) [51, 52] is utilized to parameterize data-driven Lyapunov functions for eliminating local minima. However, the drawback is that the strong convexity guarantee weakens the expressive power of the neural network model. In [31], Jin et al. proposed learning Lyapunov functions through radial basis function neural networks (NS-QLF), theoretically ensuring the stability of the learned dynamic system. However, the uniqueness of its extremal points is obtained through weak conditions. Subsequently, Jin et al. [19] proposed a neural energy function with a unique minimum (NEUM) proof of global asymptotic stability, solving the challenge of constructing a Lyapunov function consistent with non-self-intersecting demonstrations. The introduction of NEUM marked a milestone contribution to the field. Dionis Beyond these approaches, Totsila et al. [53] proposed an Autonomous Neural Dynamic Policy (ANDP), which integrates general neural network strategies into dynamic system-based policies, bridging the gap between the two. This method combines differentiability and flexible generality, enabling it to accept any observational input while ensuring asymptotic stability. In addition, some other authors have proposed to learn complex, possibly periodic, and perturbation-resistant motions by replacing regression models with Neural ordinary differential equations (ODEs) [54, 55]. At the same time, an anticipatory strategy is incorporated, enabling the robot to precisely track time-varying target trajectories. However, these methods have limited generalization capability.

This first class of ADS methods consists of three main steps: the first step is to learn the Lyapunov function $V(x)$ consistent with the provided demonstration motion; the second step is to construct the ODS. Finally, the original dynamic system is corrected using the learned Lyapunov function consistent with the demonstration, applying additional correction speeds to ensure stability and convergence of the system.

NEUM [19] is a landmark method in this class of approaches and has laid the foundation for subsequent methods. The following discussion will focus on this method, while a comparison of other ADS methods can be found in the figure at the end of this section.

3.1.1. *Neural energy function with a unique minimum*

Although the WSAQF method provides a way to construct a Lyapunov that is consistent with the demonstration, its Lyapunov function tends to appear rigid under complex demonstrations and requires an excessively large L to support it, which is clearly not the best approach. With the development of neural networks, the proposal of ICNN seems to have found a general expression for the Lyapunov function. However, its disadvantage is that it needs to guarantee strong convexity, which weakens the expressiveness of neural network models. Based on ICNN, the NEUM method proposed by Jin et al. perfectly overcomes this issue, and this framework basically lays down a data-driven universal expression for the Lyapunov function.

The NEUM method model the Lyapunov function as follows:

$$\left\{\begin{array}{c} V(x) = V_1(x) - V_1(0) + V_2(x) \\ V_2(x) = \alpha x^T x \\ V_1(x) = \omega^T f(x) \\ f(x) = \left[f_1(x), \cdots, f_k(x), \cdots, f_{d_H}(x)\right]^T \\ f_k(x) = \sigma(a_k^T z(x) + b_k) \\ z(x) = \left[||x||_2^{1+\epsilon} \; ||x||_2^{\epsilon} x^T\right]^T \end{array}\right. \tag{7}$$

where $\alpha \in \mathbb{R}_{++}$ is a positive scalar, $V_1(x)$ is represented by a neural network with the weight parameter $\omega \in \mathbb{R}^{d_H}$ and the feature $f(x) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_H}$. The activation function $\sigma(s)$ is chosen to be the well-known "tanh" function. Function $z(x) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x+1}$ is a manually designed encoder, $\epsilon \in \mathbb{R}_{++}$ is a positive scalar, $a_k \in \mathbb{R}^{d_x+1}$ and $b_k \in \mathbb{R}$ are feature parameters. Function $V_1(x)$ is the learnable part of $V(x)$, and $V_2(x)$ is used to ensure the radially unbounded property of the $V(x)$.

Lemma 1: $V(x)$ denoted by (14) is positive definite, radially unbounded, continuously differentiable, and has a unique minimum at the origin if the following parameters are satisfied:

$$\begin{cases} a_{k,1} > 0 \\ a_{k,1}^2 - \sum_{i=2}^{d_{x+1}} a_{k,i}^2 > 0, \forall k \in [1, \ldots, d_H] \\ \omega_k \succ 0 \end{cases} \tag{8}$$

For a detailed proof of the derivation, please refer to [19]. The NEUM purpose is to make the learned $V(x)$ consistent with the demonstration preferences. That is, the demonstration trajectories evolve from high-energy-value areas to low-energy-value areas. Mathematically, this purpose can be described as $\left(\frac{\partial V(x_{t,n},\theta)}{\partial x_{t,n}}\right)^T \dot{x}_{t,n} < 0, \forall \left(x_{t,n}, x_{t,n}\right) \in D$, where θ is the learnable parameter of NEUM containing the feature parameters $\{a_k, b_k\}$ and weight parameter ω.

Two approaches for learning the Lyapunov function are proposed by NEUM. In the first approach, the features $f(x)$ are manually designed to satisfy the constraints in Lemma 1. In this case, the learning problem is convex by designing an appropriate objective function. In the second approach, the feature parameters $\{a_k, b_k\}$ and the weight parameter ω are learned by a constrained learning algorithm. In this case, the learning problem is not convex, and the features $f(x)$ can be automatically fitted to the demonstration set.

1)The first learning approach: In this learning approach, the feature parameters $\{a_k, b_k\}$ are fixed and only the weight parameter ω is learnt, i.e., $\theta = \{\omega\}$, which can be obtained by solving an optimisation problem:

$$\begin{aligned} &\min_{\omega} J(\omega) = \sum_{t,n} \log\left(e^{\beta\left(\frac{\partial V(x_{t,n},\theta)}{\partial x_{t,n}}\right)^T \dot{x}_{t,n}} + 1\right) \\ &s.t. \quad \varpi_1 \leqslant \omega \leqslant \varpi_2 \end{aligned} \tag{9}$$

where ϖ_1 and ϖ_2 are any positive scalars. According to the rules of composition of convex functions [56], $J(\omega)$ is easily proved to be convex.

2)The second learning approach: In this learning approach, both feature parameters $\{a_k, b_k\}$ and weight parameter ω will be learned, which can be obtained by solving an optimization problem:

$$\begin{aligned} &\min_{\theta} J(\theta) = \sum_{t,n} \tanh\left(\frac{\left(\frac{\partial V(x_{t,n},\theta)}{\partial x_{t,n}}\right)^T \dot{x}_{t,n}}{\|\dot{x}\|_2 \left\|\frac{\partial V(x_{t,n},\theta)}{\partial x_{t,n}}\right\|_2}\right) + L_2 \|\theta\|_2^2 \\ &s.t. \quad \begin{cases} a_{k,1} > 0 \\ a_{k,1}^2 - \sum_{i=2}^{d_{x+1}} a_{k,i}^2 > 0, \forall k \in [1, \ldots, d_H] \\ \omega_k \succ 0 \end{cases} \end{aligned} \tag{10}$$

where $L_2 \|\theta\|_2^2$ is the L_2 regularization term.

Note that the objective functions $J(\cdot)$ of both methods have nested activation functions on top of (16), which has two advantages: firstly, it avoids the misinterpretation of the learning algorithm caused by using (16) directly as the evaluation function. Second, it is more helpful to weigh the accuracy and generalization of the learned $V(x)$.

After learning $V(x)$, GAS ADS $\dot{x} = g(x)$ can be obtained by introducing an additional input u into the original ADS, where the input u can be obtained by online solving the convex optimization problem with equation (4):

$$\begin{aligned} &\min_{u} \; u^T u \\ &s.t \; (f(x) + u)^T \frac{\partial V(x)}{\partial x} \leqslant -\rho(x) \end{aligned} \tag{11}$$

It is important to note that the constraint in (11) only ensures the stability requirements of the dynamic system. However, should other constraints be considered, such as the existence of certain position/velocity constraints for robot operation in the real world ([19]-Eq. (47)) or the need to avoid obstacles in the environment, etc., these constraints are necessary and can be included in the constraint framework. This setup is applicable to all ADS methods, including the second category. Some constraints are defined as follows:

$$\begin{cases} (a) \; (o(x) + u)^T x \leqslant 0, \|x\|_2 = r_{thres} \\ (b) \; \underline{v}^2(x) \leqslant (o(x) + u)^T (o(x) + u) \leqslant \bar{v}^2(x) \end{cases} \tag{12}$$

The constraint (11) is used to ensure globally asymptotically stable. The left part of constraint (12.a) is the time derivative of the $||x||_2^2$, and thus it is a position constraint for the ADS. Specifically, when the condition $||x||_2^2 = r_{thres}$ is activated, constraint (12.a) will force a contraction velocity to ensure the constraint $||x||_2 \leqslant r_{thres}$. As a result, when x falls into the region $R = ||x||_2 \leqslant r_{thres}$, it will never leave from R. The constraint (12.b) is a velocity constraint for the ADS, and $\underline{v}(x)$, $\bar{v}(x)$ is the lower and upper bound of the velocity. Just some examples are given here. In the real world with actual physical constraints, the actual constraints need to be designed according to the corresponding physical constraints.

3.1.2. Simulation and summary

Based on the analysis above, it can be concluded that the key aspect of the first class of methods is to go about learning a Lyapunov function that is highly consistent with the demonstration and possesses the global minimum property. The quality of the learned Lyapunov function will directly influence the accuracy of the learned control input $u(x)$ during the correction process.

Fig. 4 illustrates the Lyapunov function and dynamic system learning results for various ADS methods applied to the LASA handwriting dataset. These methods include QLF, WSAQF [15], NILC [16], ICNN [18], GPSSM [47], NSQLF [31], and NEUM [19]. In Tab 1, we also performed a quantitative analysis of these methods, specifically using the "S" trajectories in the LASA dataset as an example. These trajectories were sampled at 20-point intervals and contained 6 trajectories totaling 300 data points. As shown

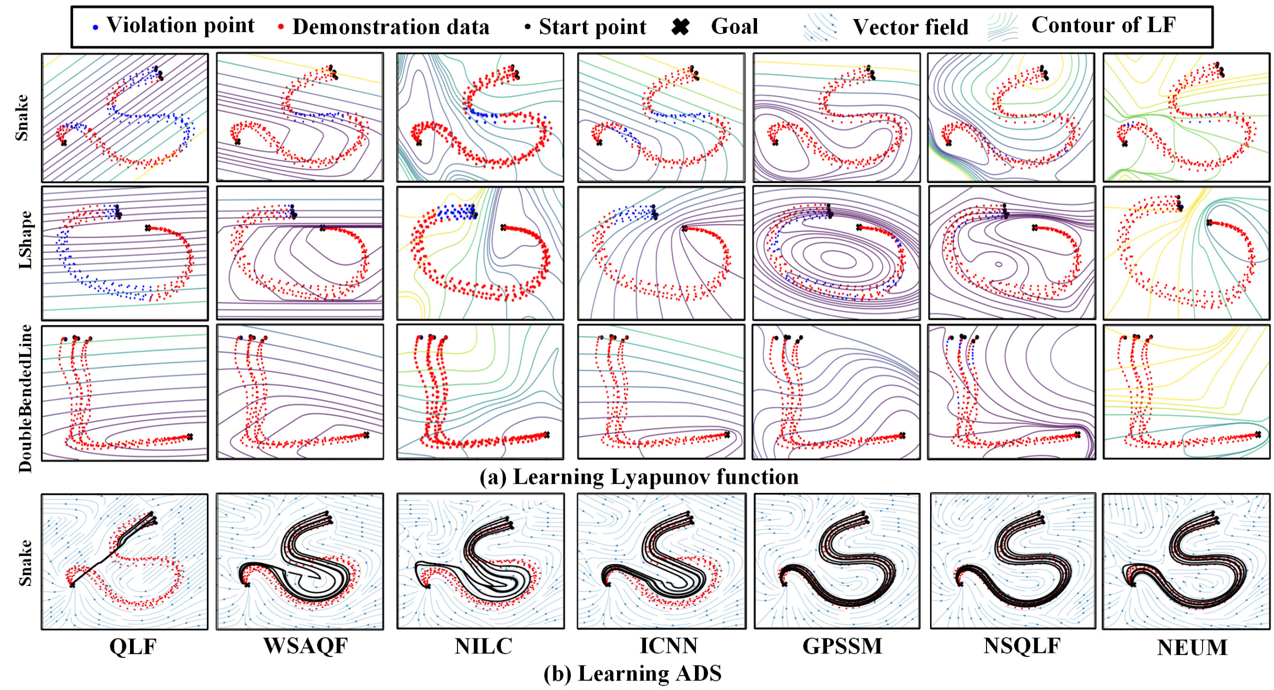

Figure 4: The Lyapunov function and dynamic system learning results for various ADS methods in the first class, including QLF, WSAQF [15], NILC [16], ICNN [18], GPSSM [47], NSQLF [31], NEUM [19] in LASA dataset, where QLF denotes the quadratic Lyapunov function of the form $x^T Px$.

Table 1
Lyapunov function for the ADS method in the first category generates speed and DS quality results(Taking the "S"-shaped trajectory in the LASA dataset as an example)

Methods	Convergence accuracy	Iterations	LF learning time/s	Violation points	Local minima	DS generation quality
QLF	1e-9	200	4.73	138	✗	low
WASQF	1e-9	200	27.15	33	✗	middle
NILC	1e-9	27	136.28	23	✗	middle
ICNN	1e-9	100	470.18	61	✗	middle
GPSSM	1e-9	230	11.14	1	✓	high
NSQLF	1e-9	29	38.9	11	✓	high
NEUM	1e-9	1000	59.27	11	✗	high

in Fig. 4(a), blue points highlight violations of the properties defined in Theorem 1, with a higher density of blue points indicating suboptimal Lyapunov function learning. Notably, some methods, such as NSQLF and GPSSM, exhibit local optima, underscoring their limitations in ensuring a globally unique minimum and highlighting their deficiencies as ADS solutions. Fig. 4(b) further confirms these observations, revealing that simpler methods, such as QLF, are effective for learning straightforward demonstration trajectories. However, for more complex trajectories, advanced approaches like NEUM demonstrate superior performance. This comparison underscores the necessity of selecting appropriately complex methods based on the trajectory complexity to achieve reliable and accurate ADS learning.

Although we believe that the NEUM provides the possibility for broad, stable dynamic system identifications, the gap between the Lyapunov function learning and dynamic system identification still exists. The above methods have learned stable estimators by controlling the Lyapunov method, i.e., introducing the correct term into the original estimator when the energy along the system is not decreasing, which, in fact, leads to the switching of the dynamic system. We propose an innovative idea to address this issue: whether it is possible to discard the hierarchical learning architecture of the original ADS method and instead use the gradient of the learned Lyapunov function to directly design a suitable ADS, which seemed to be impossible before the NEUM method was proposed because of the lack of a method to learn a Lyapunov function that is highly consistent with the demonstration. By using this strategy, the learned ADS architecture can be formulated as:

$$\dot{x} = f(x) = G(x)\dot{V}(x) \tag{13}$$

where $G(x)$ can be a machine learning or neural network approach whose main goal is to learn ADS that are consistent with real dynamic systems. Furthermore, this framework can effectively eliminate issues such as local attractors and divergence.

3.2. Learning diffeomorphism methods

As highlighted in the first category of methods, directly learning Lyapunov functions from demonstrations presents significant challenges. This has motivated researchers to investigate alternative approaches, such as learning complex Lyapunov functions indirectly by learning diffeomorphisms. This approach has received increasing attention in recent years.

However, diffeomorphisms, by their definition, must satisfy the conditions of bijectivity, smoothness and invertibility. One of the early approaches in this direction was proposed by Nicolas et al., who combined locally weighted translations to implement diffeomorphic transformations [34]. However, this method is limited to learning from a single (or average) demonstration. Neumann et al. proposed the τ-SEDS method [57], which learns diffeomorphic transformations from multiple demonstrations, mapping the proofs into a space with negligible deformations by introducing quadratic stability constraints. However, τ-SEDS relies on WSAQF to define diffeomorphic transformations, thereby imposing constraints on the hypothesis class. Urain J. et al. proposed using invertible neural networks [58, 59] to fit a diffeomorphism, mapping linear motions into complex motions [60, 61]. Conversely, the opposite approach is to map complex motions into linear motions. Euclideanizing Flows (E-FLOW) [62, 63] linearizes demonstrations using flexible function approximators such as kernel methods [64] and neural networks [65], making them behave like diffeomorphisms generated by linear DS. However, methods based on deep networks require longer training times and intensive hyperparameter search. Saveriano et al. proposed a method called Riemannian manifold into a stable dynamic system (SDS-RM) [66]. Through GMM [67–69], the differential diffeomorphism transformation of stable dynamic systems evolving on the Riemannian manifold [70] was achieved. Rodrigo et al. introduced the use of contrastive learning loss [14] in deep neural networks to train models that resemble diffeomorphism, a method they coined Convergent Dynamics from Demonstration (CONDOR). This approach enables the network to effectively learn complex motions while improving its accuracy. However, while CONDOR demonstrates the ability to handle complex trajectories, it does not provide a rigorous proof of the stability of the learned dynamic systems. Instead, it relies on soft constraints during optimization to approximate stability [41]. Furthermore, Rodrigo et al. proposed a more flexible learning framework by reformulating stability conditions and introducing triplet loss [41], which effectively addresses problems in non-Euclidean state spaces. Based on E-FLOW, by introducing residual structures into the neural network architecture [71], zhang et al. effectively solved the invertibility of diffraction and ensured the superiority of both the accuracy and stability of DS. Recent work by Zhi et al. [72] has introduced the Stable Periodic Diagrammatic Teaching (SPDT) framework, which models robotic motion using orbitally asymptotically stable (O.A.S.) dynamical systems. The framework's core innovation lies in employing diffeomorphisms - differentiable and invertible transformations such as invertible neural networks - to deform existing O.A.S. systems, thereby stabilizing motion trajectories. This approach effectively overcomes the well-documented limitation in prior literature [73] where only simple periodic skills could be learned.

The first class of methods corrects the system by directly learning a Lyapunov function that is consistent with the demonstrated preferences. In contrast, the second class of methods focuses more on going through the learning of a mapping that ensures the GAS of the dynamic system by transforming the complex problem into a simple linear problem, and then utilizing a simple quadratic Lyapunov function (which does not need to be learned). Since the learning process of the mappings has implicitly encoded stability constraints in the optimization objective, the method requires only a single-step optimization training to achieve both motion mimicry and stability guarantees. According to the different mapping objects, there are two sub-methods: one explores the mapping between simple and complex trajectories, while the other establishes the mapping between complex and simple DS. An intuitive description is provided in Fig. 5.

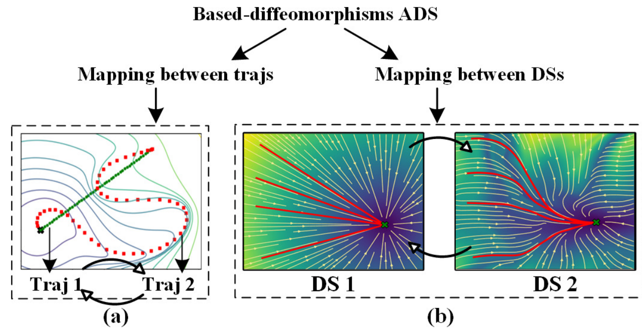

Figure 5: The based-diffeomorphism ADS methods subdivision. (a) Mapping between simple and complex trajectories, (b) Mapping between simple and complex DSs. The classic methods on the left include FDM [34], while on the right, there are E-Flow [62], I-Flow [60], and so on.

E-FLOW [62] is a representative method in this class of approaches and has laid the foundation for such methods to some extent, while a comparison of other diffeomorphism-based ADS approaches is provided in the figure at the end of this section.

3.2.1. Euclideanizing Flows

Inspired by work on density estimation [74], E-FLOW utilizes a class of parametric differential homography methods for learning a variety of actions that can be adapted to different tasks with minimal parameter tuning. By encoding a complex human action as a dynamic system, it is connected to a simple gradient descent dynamic system in potential space using a learnable differential homography method. This connection allows the stability performance of the manually specified dynamic system to be directly carried over to the learned dynamic system.

Before introducing the algorithm, it is first necessary to understand a concept: the flow-based diffeomorphisms method. By definition, a differential pass must be both bijective and continuously differentiable. To achieve this, E-FLOW adds structure to the learning problem, i.e., diffeomorphism by a combination of K differential deformations $\varphi = \varphi_1 \circ \varphi_2 \circ \cdots \circ \varphi_K$, each of which is given by coupling layer $\varphi_k : \mathbb{R}^n \to \mathbb{R}^n$ [74]. Fig. 6 illustrates the architecture of the coupling layers. Each coupling layer φ_k splits the input z_{k-1} into two parts and applies a scaling function s_k and a translation function t_k to one of them to obtain the output z_k. This approach ensures that the inverse operation of the transformation is easy to compute, thus guaranteeing the reversibility and stability of the model. It is specified as follows:

$$z_k = \begin{bmatrix} z_k^a \\ z_k^b \end{bmatrix} = \begin{bmatrix} z_{k-1}^a \\ z_{k-1}^b \odot \exp\left(s_k(z_{k-1}^a)\right) + t_k(z_{k-1}^a) \end{bmatrix} = \varphi(z_{k-1}) \quad (14)$$

where the scaling and translation functions are given by a single-layer neural network whose layers resemble an approximate kernel machine [75]. There are many similar coupling layer architectures [76–78] and modifications, such as adding attention mechanisms [13], etc.

After understanding the concept of the flow-based diffeomorphisms, consider a simple gradient descent dynamic system $\dot{y} = f(y) = -\nabla_y \Phi(y)$ with a potential function $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}$. The choice of the potential function can be some potential function, such as a quadratic potential function $x^T x$ that provides a guarantee of ideal stability. This potential function generates unit-velocity straightline motions to the globally asymptotically stable equilibrium point y^*. The diffeomorphism acts to deform these straight lines to arbitrarily curved motions converging to x^*, where the demonstrations converge. The change of coordinates defined by the diffeomorphism φ can then describe the same dynamics in the x-coordinate:

$$\dot{x} = -G_\varphi(x)^{-1} \nabla_x \Phi(\varphi(x)) = f_\varphi(x) \quad (15)$$

where the induced Riemannian metric in the domain is given by $G_\varphi(x) = J_\varphi(x)^T J_\varphi(x) \in \mathbb{R}^{n \times n}_{++}$. The aforementioned dynamics is known as natural gradient descent, which is the steepest descent on a Riemannian manifold [79, 80], with respect to the potential function $\Phi \circ \varphi$.

With a parameterized diffeomorphism φ_θ, the learning problem reduces to solving:

$$\hat{\theta} = \arg\min_\theta \frac{1}{\sum_{i=1}^N T_i} \sum_{i=1}^{N} \sum_{t=1}^{T_i} \left\| \dot{x}_{i,t} - f_{\varphi_\theta}(x_{i,t}) \right\|_2^2 \quad (16)$$

where N and T_i are the maximum number of demonstration trajectories and the corresponding trajectory length, respectively.

3.2.2. Simulation and summary

From the analysis of the methods described above, it can be concluded that the primary focus of the second class of approaches is learning mappings. This includes point-to-point mappings between complex and simple trajectories, as well as mappings between complex and simplified dynamic systems. The quality of these learned mappings plays a crucial role in determining the overall performance and accuracy of the learned dynamic system.

Fig. 7 presents the dynamic system learning results for various ADS methods in the second class, providing a more intuitive understanding of their performance. The methods compared include FDM [34], τ-SEDS [57], E-FLOW [62], and CONDOR [14], evaluated on the LASA handwriting dataset. The results shown in Fig. 7(a) support the conclusions drawn. In Fig. 7(b), the blue dots represent data points that violate the property outlined in Lemma 1, with a higher concentration of blue dots indicating poorer performance of the Lyapunov function derived from the diffeomorphisms. From these results, it is evident that the accuracy of WSAQF significantly improves after incorporating diffeomorphism processing, as demonstrated in the second panel of Fig. 7(a). On the other hand, FDM is limited to single inter-trajectory mappings. The CONDOR method, while effective, suffers from the issue of local minima due to its weak stability interpretation, as highlighted in Fig. 7(c), where multiple local minima are observed in the velocity vector diagram. It is crucial to note that diffeomorphism-based methods are particularly well-suited for handling complex trajectory distributions. In contrast, for simpler cases, methods like QLF are often sufficient.

Finally, To provide a clear overview of the development of DS methods in LFD, as well as performance metrics, we present a detailed representation in Tab.2.

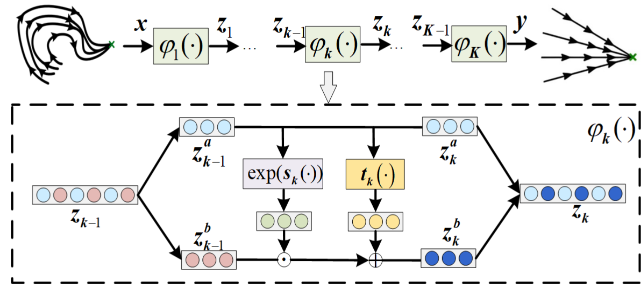

Figure 6: The architecture of the coupling layers [62].

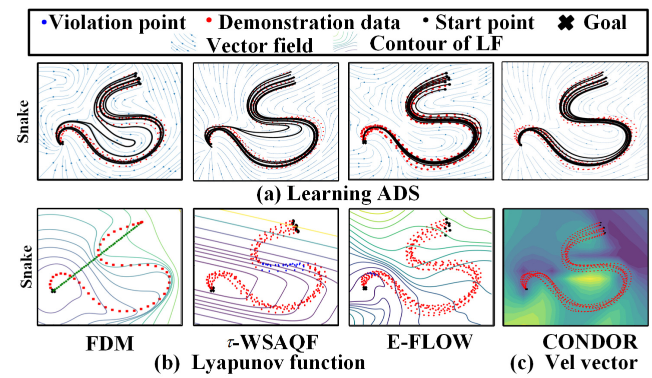

Figure 7: The dynamic system and Lyapunov function learning results for various ADS methods in the second class, including FDM [34], τ-SEDS [57], E-FLOW [62], CONDOR [14] in LASA handwriting data set. There are also simulations of some of the ADS methods that are not shown in the figure, but they can be found in the GitHub repository we uploaded.

Table 2
Summary of the main dynamic system methods.

Methods	Lyapunov form	DS quality	Global minimum	Training time	Generalization	Advantages	Disadvantages
SEDS [12]	SOS	middle	✓	short	bad	Easy to handle simple trajectories	Difficult to handle complex trajectories
CLF-DM [15]	WSAQF	middle	✓	short	bad	Good applicability to complex trajectories	Accuracy is determined by the number of asymmetric terms
NILC [16]	ELM	middle	✓	long	middle	High accuracy at high sampling rate	GAS has no theoretical guarantee
GPSSM [47]	SOS	high	✗	short	bad	Enhancing the convergence of the model's motion	Global uniqueness is not guaranteed
ICNN [18]	ICNN	middle	✓	long	middle	Strong nonlinear properties of Lyapunov function	Weak representation of ICNN models
CDSP [8]	Contraction metric	middle	✓	short	middle	Accurately model a wider class of motions	Restrictive and difficult to model highly complex shapes
NS-QLF [31]	Neural-shaped quadratic form	high	✗	short	bad	Relatively good accuracy and generalization	Global uniqueness is not guaranteed
NEUM [19]	Neural network form	high	✓	middle	good	Strong nonlinear properties of Lyapunov function	Require substantial computation time
LPV-DS [44]	P-QLF	middle	✓	short	middle	Good convergence for complex nonlinearities	Low model accuracy and low computational efficiency
N-ODE [54]	WSAQF	high	✗	short	middle	Easily capture invariant features of target trajectories	Global uniqueness cannot be guaranteed
RDS [43]	-	middle	✓	long	bad	Higher replication accuracy	Higher computational burden
τ-SEDS [57]	-	middle	✓	short	middle	modular implementation without much coding effor	Diffeomorphic candidate transformation simple
FDM [34]	Diffeomorphic matching*	high	✓	long	middle	Higher precision	Learning single-trajectory motion only
I-FLOW [60]	INN*	high	✓	long	middle	High accuracy, not limited to point-to-point tasks	Require substantial computation time
E-FLOW [62]	INN*	high	✓	long	good	Highly accurate and generalisable	Require substantial computation time
SDS-RM [66]	GMM*	middle	✓	short	middle	Fast calculation speed	Inability to converge precisely to the target point
CONDOR [14, 41]	DNN*	high	✗	long	middle	Good non-linear properties good generalisation	Weak stability high computational burden

1. The upper part of the table represents the set of methods for the first class, while the lower part represents the set of methods for the second class.
2. - represents that the form of the Lyapunov function is determined by the selected DS method.
3. * represents the diffeomorphisms of the Lyapunov function SOS.

3.3. DS with intersections and Second-order DS

As discussed in Section 2.3, there is currently no universal framework to address the learning of DS with intersections and second-order DS, as shown in Fig. 8, but some works have begun to tackle this issue.

In [38, 39], Jin et al. learned a cross-dynamic system using a dimension enhancement method based on manifold immersion. Although the method is clever, an inherent dilemma of this approach is that models learned under stability constraints may produce inaccurate reproductions when given demonstrations that violate Lyapunov functions. With the same idea of dimensional enhancement, Zhang et al. proposed a Neural Liénard System (Neural LS) [81], which utilizes Liénard-type differential equations to construct a dynamic system with stability and unique characteristics. By describing the additional dimension as a function of the Liénard system's state, this method extends the applicability of Neural LS to higher dimensions while endowing it with the capability to represent trajectories with intersections. In [13], Zhang et al. constructed a dynamic system using a neural network-based approach, considering both the position, velocity and acceleration of demonstrations, enabling the proposed DS to learn complex intersecting motions still. However, this method of forcibly defining inputs and outputs lacks concrete mathematical guarantees and practical significance. For this problem, Ratliff et al. [82] introduced the Riemannian Motion Strategy (RMP), which generates second-order DS whose behavior is inherently related to the Riemannian metric. In addition, Bernardo et al. [83]

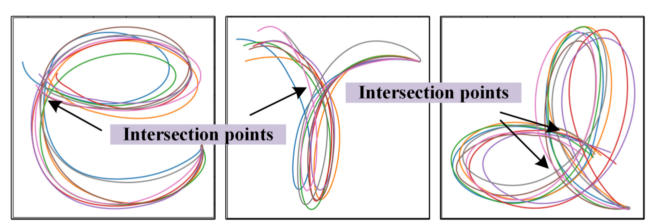

Figure 8: Self-intersection demonstration in the Hello World dataset [33]. Trajectory shapes such as "e", "r", and "d" have one to multiple intersections.

proposed a nonlinear DS learning method based on a pure geometric framework, where the curvature of the Riemannian manifold captures the inherent non-linearity of second-order dissipative DS. However, the limitation of this method is that not all d-dimensional manifolds can be isometrically embedded into a $d+1$-dimensional Euclidean space, which restricts the nonlinearity complexity it can effectively learn.

Although foundational work in this area is still limited, the current methods provide valuable insights for learning higher-order DS.

3.4. Lyapunov generation efficiency and DS quality analysis

During the training generation of Lyapunov functions, we compare the performance of different methods in terms of generation speed and quality. As shown in Tab 1 and Tab 2, the length of the training time in Tab 2 is compared to the time in Tab 1. The first method learns the Lyapunov function through direct optimization, which is faster. However, when dealing with complex tasks, this method requires the use of complex neural network models in order to learn Lyapunov functions consistent with the trajectory, which increases the computational time. The second method indirectly generates the Lyapunov function by learning the diffeomorphism. Since the diffeomorphism usually relies on a neural network for learning and requires more training epochs to ensure the quality of the generated function, this method is slower but can effectively handle more complex dynamic systems.

Current research is mainly based on a small amount of demonstration data (usually a few demonstration trajectories), so the generation speed has not become a major bottleneck. As can be seen in Tab 1, training with a small data set takes only a few tens of seconds to a few minutes. However, as the amount of data increases and the model size expands, training efficiency will gradually become a challenge that cannot be ignored. Future research can explore methods such as parallel computing, approximation algorithms, and incremental learning to ensure the quality of generated dynamic systems while improving training efficiency, thus adapting to the needs of larger and more complex tasks.

4. Applications of ADS

In the preceding sections, we have provided an overview of the theoretical advancements in ADS. We will focus on the applications of these methods to real-world robotics tasks, in particular, how to encode human motor skills through learning and transfer these skills to robot operations, illustrated by relevant legends, as shown in Fig. 9. In addition, we add a critical comparative analysis of the tradeoffs between the various approaches for applied environments. These additions highlight the applicability of ADS beyond the benchmark dataset.

4.1. Manufacturing

In manufacturing applications, DS-based learning methods, in particular the use of Lyapunov functions to construct globally stable control models, have been widely used to encode human motor skills and migrate them to robot operations. This approach offers significant advantages in terms of improved production adaptability and portability, and is particularly suitable for robots that are able to learn from a small number of examples. These robots are able to collect task motion trajectory data through kinesthetic teaching or teleoperation and, in turn, learn how to perform complex tasks in real-world environments. In particular, global stability control based on Lyapunov functions not only ensures the safety and stability of the robot's task execution, but also provides strong generalization capabilities, reduces the need for reprogramming, and effectively reduces production downtime [19]. For example, in operations such as pick-and-place [31, 84], peg insertion [85, 86], polishing [73, 87], and assembly [19] that use the ADS method, robots such as Franka Emika Panda are able to achieve high accuracy and robustness by learning task motion trajectories (or impedance parameters) and stability control of dynamic systems, and excelled in complex tasks such as assembly and polishing [88]. In contrast, traditional pre-programming methods, time-varying trajectory deviations of MP methods [23], and internal force effects and rigidity shocks faced by tasks such as assembly may cause failure of high-precision assembly tasks and damage to the robot body.

However, despite the excellent performance of the Lyapunov function in these tasks, its applications in high-dimensional tasks still faces some challenges, especially in three-dimensional space. Model complexity, external disturbances, and environmental changes in high-dimensional environments may affect the robustness and stability of the model. For example, in the polishing task, the robot needs to precisely control the force and angle, and the uncertainty of the environment may lead to a degradation of the control system's performance. Therefore, although the Lyapunov function provides theoretical support in ensuring system stability, its effectiveness in real robotics applications still depends on model tuning and environment adaptation.

4.2. Assisting and healthcare robotics

In recent years, learning methods based on dynamic systems have been increasingly applied to assistive and medical robots, especially in tasks such as autonomous ultrasound scanning, feeding, and robotic surgery [19, 87]. Similar to manufacturing applications, medical robots are often learned using kinesthetic teaching or teleoperation, where motion trajectories are extracted from a small number of examples to perform complex operations. However, unlike robotic tasks in manufacturing, assistive and medical robots often require higher accessibility and interaction with humans, which also brings higher safety requirements. In a medical environment, robots not only need to perform highly precise tasks, but also ensure the safety of patients and medical staff. Therefore, providing convergence and stability guarantees for learning strategies, such as using Lyapunov functions to ensure the global stability and robustness of the system, is the key to ensuring that robots can operate safely and efficiently. For example, during robotic surgery, Lyapunov functions can

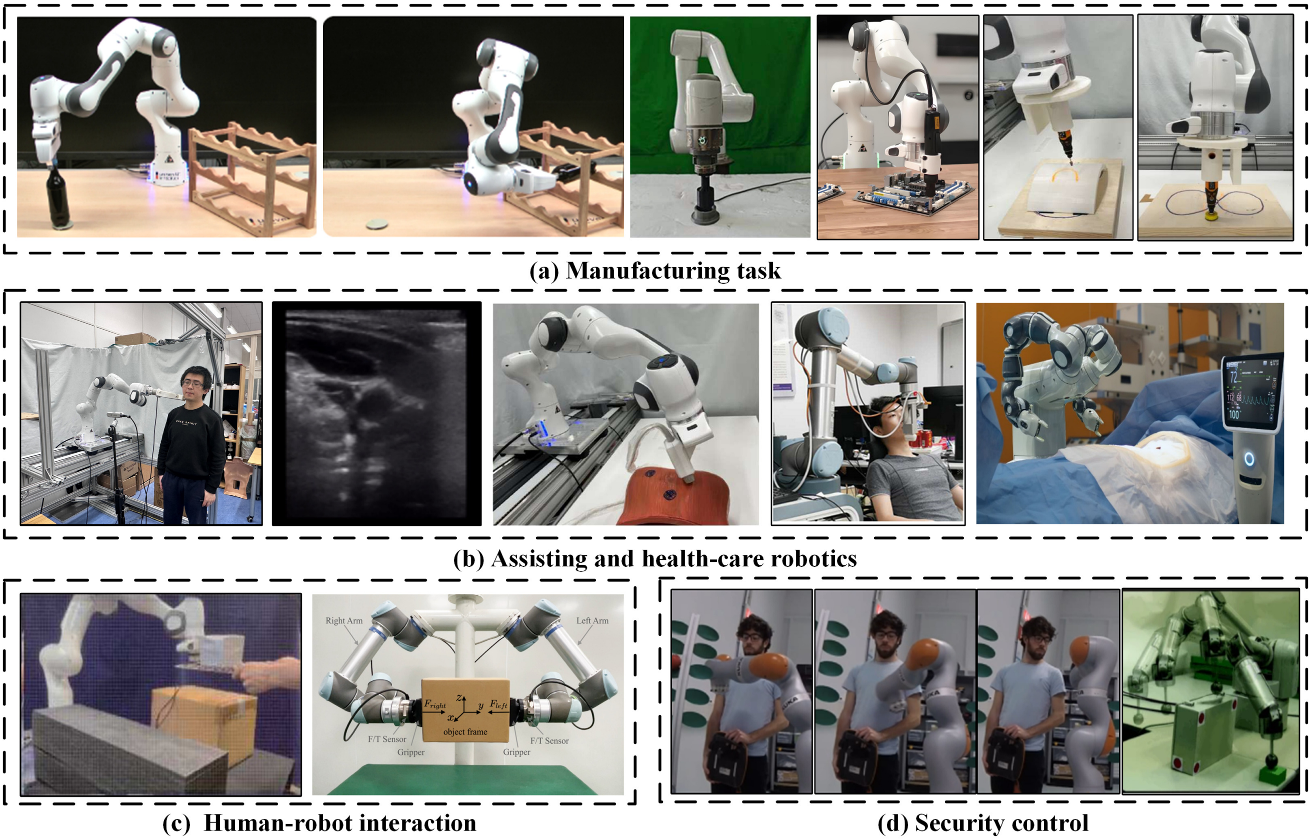

Figure 9: The various robot applications of ADS methods. (a) manufacturing task: pick and place [66], peg insertion [31, 86] assembly operations, polishing [73, 87, 88]. (b) assisting and healthcare robotics: Ultrasound scanning [19, 89], assisted surgery. (c) Human-robot interaction: collaborative transportation [90]. (d) Security control: obstacle avoidance [12, 14].

be used to optimize the stability of the robot during tasks such as high-precision cutting and suturing, reducing the risk caused by unstable operation. Although Lyapunov functions can provide theoretical stability guarantees in high-dimensional spaces, especially in the real-time decision-making process of complex surgical operations, the amount of computation and the complexity of adjusting strategies may become limiting factors.

4.3. Human-robot interaction and collaboration

In addition to allowing robots to autonomously complete tasks in different fields, the dynamic system approach is also widely used to enable robots to work closely with humans. In this process, the focus of dynamic system learning is no longer just the relationship between position and speed, but also other dynamic factors such as force [90]. Effective collaboration requires the robot to generate ideal movements that complement human actions, which not only improves the fluency of the collaboration, but also ensures safety. In human-robot collaboration scenarios such as collaborative material handling and collaborative assembly [91, 92], the robot's movements need to be highly coordinated with human movements, especially in situations where the environment changes frequently, or the task requirements are complex. Such tasks often require the robot to quickly adapt and adjust in a dynamic environment, and this process can be modeled and optimized using a dynamic system approach. The active dynamic system learning method enables the robot to automatically compensate for disturbances during manipulation, thereby achieving more accurate and stable motion control. In addition, human-robot interaction applications also increase the demand for compliant robot control, especially in tasks that require delicate adjustments [93]. By learning appropriate joint torque, stiffness and damping parameters [40, 88, 94, 95] through a dynamic system approach, robots can flexibly adapt to the needs of human collaboration. In particular, by coupling robot dynamics and impedance control models through the state-driven mechanism of ADS, combined with the forward-looking optimization capability of Model Predictive Control (MPC) [96, 97], robots can be synergistically optimized for force-control accuracy and dynamic response. Such force control operations are essential for tasks such as robot trajectory tracking and surface lettering.

4.4. Security control

Learning and generalizing obstacle avoidance trajectories is crucial for safe robot control. In robotic obstacle avoidance tasks, the Control Barrier Function (CBF) [98], and modulation matrices [99] have been shown to be effective methods for autonomous dynamic system obstacle avoidance in such approaches [100–102]. Specifically, CBF ensures the safety of the generated trajectory by enforcing forward invariance of the safety set. To prevent additional control inputs from compromising the system's stability, a CLF-CBF framework combining the Control Lyapunov Function (CLF) and CBF has been proposed. In this framework, the CLF is used to design the stability control strategy for the robotic arm, while the CBF ensures that the system remains stable while satisfying safety constraints simultaneously. The modulation matrix method, on the other hand, achieves effective obstacle avoidance by precisely modulating the dynamic system[86, 103, 104].

However, despite the theoretical stability guarantees provided by these methods, there are still some shortcomings in practical applications. In real-time obstacle avoidance, the computational complexity of the CLF-CBF framework is relatively high, which may lead to real-time problems. At the same time, the obstacle avoidance interval introduced by CBF may cause the dynamic system to fall into a local optimal solution, which may lead to task failure. The modulation matrix method, although it can effectively cope with complex environments, relies on accurate system modeling and may require frequent adjustments when facing unknown or rapidly changing obstacles, which places higher demands on computing power and real-time control.

4.5. Comparative analysis of ADS methods

In this subsection, we will analyze the key trade-offs between different ADS methods across various application domains, building on the comparative insights introduced in earlier sections. This will help readers select the appropriate method based on their specific requirements and constraints.

The applications of ADS in robotics involve several fundamental trade-offs spanning all domains: 1) Stability vs. Flexibility: Methods based on Lyapunov functions and control barrier functions provide strong stability guarantees but may limit the system's adaptability to new situations. Conversely, more flexible learning-based approaches may offer better adaptability but with reduced stability assurances; 2) Computational Complexity vs. Real-Time Performance: High-dimensional tasks and complex environments require sophisticated models, which often demand significant computational resources—potentially impacting real-time performance, a critical requirement in most robotics applications; 3) Generalization vs. Task-Specific Performance: Methods that generalize well across multiple tasks may underperform compared to specialized solutions designed for specific applications.

Different domains have distinct requirements. For example, In manufacturing environments, trade-offs manifest particularly in precision vs. speed: High-precision tasks, such as peg insertion, polishing, or assembly, require slower, more controlled motions, whereas throughput demands often push for faster operations. Additionally, structured vs. unstructured environments play a role: Well-structured settings allow simpler models to perform better, while increasingly unstructured environments necessitate more complex and adaptive approaches, such as NEUM methods or second-class diffeomorphism-based techniques. In medical robotics, greater emphasis is placed on safety margins vs. performance: Conservative control parameters ensure patient safety but may limit surgical effectiveness or duration. Another consideration is human-in-the-loop vs. full autonomy, where the required level of human supervision reflects both technical and ethical considerations. In human-robot interaction and collaborative scenarios, predictability vs. adaptability is key: Predictable robot behavior enhances human trust but may reduce the system's ability to adapt to human variability. Additionally, physical compliance vs. task efficiency must be weighed: Softer, safer interactions often come at the cost of slower or less precise task execution. For safety-critical applications, ADS methods must incorporate safety control techniques to enable obstacle avoidance and safety assurance. However, formally verifiable methods are often simpler but may struggle to handle complex real-world scenarios effectively. There is also a trade-off between safety guarantees and computational tractability.

This discussion underscores that selecting an ADS method for a given application requires careful consideration of domain-specific requirements and constraints. There is no universally optimal approach; instead, the best solution carefully balances competing needs for each application, potentially combining different methods to fulfill corresponding tasks.

5. Challenges and future directions

The evaluation of various methods primarily hinges on their ability to deliver practical value in real-world applications. The Movement Primitive (MP) methods [105–110] have been extensively validated over many years, proving their stability and practicality across a variety of real-world applications [111–114]. These methods can effectively operate in complex environments, offering reliable solutions, and have been widely applied in fields such as industrial automation, robotic control [115], and medical assistance [116]. In contrast, while ADS methods are innovative and theoretically advanced, they are still in the early stages of practical application. Many of their core technologies and algorithms remain immature. Although ADS methods may demonstrate strong performance in controlled laboratory settings, they often encounter significant challenges when applied in real-world environments, such as environmental complexity, data uncertainty, and real-time processing requirements. These factors undermine the stability and reliability of ADS methods outside the laboratory setting.

This section will outline the challenges and future directions for the advancement of the ADS method.

5.1. The dilemma of generalization

Generalization in robotic learning can be classified into two types: intra-task generalization and inter-task generalization. Intra-task generalization refers to an algorithm's ability to adapt to new conditions within a given task, such as varying initial and target locations, different object placements, or changing environmental factors. In contrast, inter-task generalization involves transferring learned skills to new, but related tasks. Dynamic systems-based approaches often rely on expert demonstrations to collect training data. However, because these demonstrations typically do not cover the entire task space, robots may encounter situations where the input distribution deviates from the demonstration distribution. This mismatch can result in poor intra-task generalization and failure in inter-task generalization. To address this challenge, it is essential to develop learning methods capable of extrapolating acquired knowledge to novel scenarios. Moreover, these methods must include mechanisms for evaluating the applicability of learned policies to new environments. In practical terms, this means that robots must be able to recognise when they can operate autonomously and when user intervention is required, striking a balance between generalization with system reliability.

5.2. Hyperparameter selection

Hyperparameter selection remains a significant challenge in machine learning-based approaches for learning the mapping between representations and actions. While this issue is common across many machine learning methods, it is particularly critical in LfD, where automatic hyperparameter tuning is needed to enhance usability, especially for non-expert users. One of the main motivations for applying LfD is to enable non-experts to program robotic systems, but the necessity for manual hyperparameter adjustment limits the accessibility of this approach.

In dynamic system-based approaches, hyper-parameters can be found in many representations of Lyapunov function construction and diffeomorphism architectures. From Fig.2, it can be seen that since 2019, the representations of Lyapunov functions are often modeled by neural networks. In these cases, the hyper-parameters include the number of hidden layers and neurons per layer. Modeling complex, nonlinear demonstration data may require more hidden units, whereas simpler demonstrations may need fewer. Similarly, in diffeomorphism architectures, it is often necessary to set the hyper-parameters along with the parameters for stability guarantees to learn the mapping relationship between two distributions via a neural network. But before 2019, automatic hyper-parameter selection can be achieved using machine learning methods with learning rules that trade-off between model fitting and complexity (e.g., Gaussian processes). However, their availability as a strategy function is limited by their computational complexity and the challenge of guaranteeing system stability.

5.3. High-order ADS

In first-order systems, we are mainly concerned with the position and velocity of the system, and these variables can intuitively describe the state of motion of the object. However, in second-order systems, the introduction of acceleration makes the motion state of the system more complex. Acceleration is a quantity that describes the change of velocity, and its introduction not only increases the dimensionality of the system but also makes the dynamic properties of the system richer. Of the above-mentioned methods, all are for first-order systems (position, velocity). However, when these methods are applied to more complex second-order systems (position, velocity, acceleration) (e.g., cross-motion), most of the methods become less effective or applicable. Some literature [13, 38, 39] models the second-order dynamic system in the form of the left-hand side of the Eq. (7). However, as mentioned earlier, these methods fail to provide a clear physical interpretation of the generated equation. Similarly, the DNN architecture used in CONDOR [14] for second-order systems suffers from weak interpretability, further limiting its practical application and understanding.

Therefore, future research should focus on developing more interpretable, scalable methods capable of addressing the unique challenges of second and even higher-order dynamics, ultimately enabling more reliable and adaptive robotic control in real-world applications.

5.4. Robust multitasking applications

In multitasking applications, LfD enables robots to perform multistep tasks by partitioning demonstrations into subtasks, goals, phases, keyframes, or skills/prototypes [117–122], [123, 124], [125, 126], [127–129]. Most of these abstractions assume that sequentially achieving subgoals will lead to the desired outcome. However, the successful imitation of many manipulation tasks with spatial and temporal constraints cannot be reduced solely to motor-level imitation, unless the learned motor strategies also satisfy these constraints. This is particularly relevant when robots are expected not only to imitate, but also to generalize, adapt, and maintain robustness against human-imposed perturbations that occur during task learning and execution. For instance, while the method in [130] guarantees the learning of stable motion strategies with convergence, this guarantee is limited to the motion level. DS-based methods, such as trajectory segmentation, behavioral decision trees [131], and dynamic regression models [84], have been applied in multi-skill learning. However, these methods face challenges in reliably abstracting subtasks. Even when abstractions are provided, DS methods that lack invariant or reachable properties struggle to ensure robust task execution and replay. Task robustness refers to a system's ability to handle perturbations during execution without causing failure or significant deviation. However, the methods mentioned above are limited to motion-level adaptation. While they guide the robot to complete tasks, they often do not fully account for broader task-level perturbations, which could lead to failures or inefficiencies in real-world applications.

5.5. Embodied intelligence

Within the field of embodied intelligence, the application of LfD is particularly evident in several key areas: behavior imitation, task planning and environmental adaptation [132]. As LfD methods, particularly those based on ADS, continue to evolve, they offer the potential to significantly enhance the autonomy and decision-making abilities of robots. This can drive advancements in both the theoretical foundations and practical applications of embodied intelligence. For instance, the integration of ADS facilitates the development of more robust, flexible, and adaptive robotic systems capable of performing complex tasks in dynamic, real-world environments, especially humanoid robots [12]. By improving the stability, generalization, and efficiency of learning strategies, ADS-based techniques can help robots autonomously adapt to new scenarios and perform tasks with high precision and reliability.

However, in the real world, robotic systems still face significant challenges in goal specification due to perceptual and generalization difficulties coupled with severe data scarcity. Future research should prioritize exploring deeper integration between ADS and multimodal architectures, which could drive paradigm-shifting advances in embodied intelligence: At the perception level, visual foundation models (e.g., ViT [133], CLIP [134]) can empower ADS with scene understanding and feature encoding capabilities, establishing vision-action closed-loop systems. At the decision-making level, large language models (LLMs) [135] can parse natural language instructions into ADS dynamic parameters through semantic reasoning, enabling language-guided behavior generation. At the control level, diffusion models [136] can collaborate with ADS through synergistic optimization—where the former generates candidate trajectory distributions while the latter ensures motion stability—thereby significantly improving imitation learning's sample efficiency and generalization performance. Though this integrated framework will encounter challenges like modality alignment and real-time computation, its demonstrated "perception-cognition-control" triad offers a verifiable technical pathway for humanoid robots performing complex tasks in open-world environments. Future studies should further investigate hierarchical ADS architectures and physics-informed joint training paradigms. These advancements will propel embodied intelligence systems from mere functional implementation to genuine cognitive emergence in unstructured settings.

6. Conclusion

This paper provides a comprehensive review of recent advancements in autonomous dynamic systems for demonstration learning, proposing a classification framework that distinguishes between two primary categories: Lyapunov function-based methods and diffeomorphism-based methods. For each category, we summarize the key features, advantages, and limitations, placing particular emphasis on foundational methods that have notably propelled progress in the field, and provided intuitive simulation comparison demonstrations. Additionally, the paper explores the research landscape of dynamic systems with intersection properties (DS) and second-order dynamic systems. To facilitate further exploration and experimentation, we also present simulation results and provide open-source code.

Furthermore, the paper explores emerging industrial applications of ADS methods, with a particular focus on sectors such as manufacturing, medical robotics, human-robot interaction, and security control. We discussed the potential benefits provided by these methods, the trade-off analysis between methods, and the challenges they face in practical deployment. The paper identifies several key challenges that remain, such as improving the generalization of skill learning, optimizing hyperparameter selection, enabling higher-order skill learning, and advancing multi-task learning for complex robotic tasks. These areas of development are crucial for enhancing the stability and safety of robotic skill learning, driving further innovation in embodied intelligence, and supporting advancements in industrial automation and related fields.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Jiayun Fu: Conceptualization, Methodology, Software, Writing – original draft. **Haotian Huang:** Investigation, Writing – original draft. **Zhehao Jin:** Supervision, Writing – review & editing. **Andong Liu:** Supervision, Review & editing, Funding acquisition. **Wen-An Zhang:** Review & editing. **Li Yu:** Writing- Review & editing. **Weiyong Si:** Writing- Review & editing. **Chenguang Yang:** Writing-Review & editing.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F030002.

References

[1] H. C. Ravichandar, A. Dani, Learning position and orientation dynamics from demonstrations via contraction analysis, Autonomous Robots 43 (4) (2019) 897–912.

[2] D. Hood, S. Lemaignan, P. Dillenbourg, When children teach a robot to write: An autonomous teachable humanoid which uses simulated handwriting, in: Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction, 2015, pp. 83–90.

[3] Q. Luo, J. Wu, M. Gombolay, A generalized robotic handwriting learning system based on dynamic movement primitives (dmps), arXiv preprint arXiv:2012.03898 (2020).

[4] J. Fong, M. Tavakoli, Kinesthetic teaching of a therapist's behavior to a rehabilitation robot, in: 2018 international symposium on medical robotics (ISMR), IEEE, 2018, pp. 1–6.

[5] A. Billard, S. Calinon, R. Dillmann, S. Schaal, Survey: Robot programming by demonstration, Springer handbook of robotics (2008) 1371–1394.

[6] S. Schaal, A. Ijspeert, A. Billard, Computational approaches to motor learning by imitation, Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences 358 (1431) (2003) 537–547.

[7] G. Mamakoukas, I. Abraham, T. D. Murphey, Learning stable models for prediction and control, IEEE Transactions on Robotics 39 (3) (2023) 2255–2275.

[8] H. C. Ravichandar, A. Dani, Learning position and orientation dynamics from demonstrations via contraction analysis, Autonomous Robots 43 (4) (2019) 897–912.

[9] P. Giesl, S. Hafstein, Review on computational methods for lyapunov functions, Discrete and Continuous Dynamical Systems-B 20 (8) (2015) 2291–2331.

[10] S. Singh, S. M. Richards, V. Sindhwani, J.-J. E. Slotine, M. Pavone, Learning stabilizable nonlinear dynamics with contraction-based regularization, The International Journal of Robotics Research 40 (10-11) (2021) 1123–1150.

[11] H. Tsukamoto, S.-J. Chung, J.-J. E. Slotine, Contraction theory for nonlinear stability analysis and learning-based control: A tutorial overview, Annual Reviews in Control 52 (2021) 135–169.

[12] S. M. Khansari-Zadeh, A. Billard, Learning stable nonlinear dynamical systems with gaussian mixture models, IEEE Transactions on Robotics 27 (5) (2011) 943–957.

[13] Y. Zhang, L. Cheng, H. Li, R. Cao, Learning accurate and stable point-to-point motions: A dynamic system approach, IEEE Robotics and Automation Letters 7 (2) (2022) 1510–1517.

[14] R. Pérez-Dattari, J. Kober, Stable motion primitives via imitation and contrastive learning, IEEE Transactions on Robotics 39 (5) (2023) 3909–3928.

[15] Khansari-Zadeh, S Mohammad and Billard, Aude, Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions, Robotics and Autonomous Systems 62 (6) (2014) 752–765.

[16] A. Lemme, K. Neumann, R. F. Reinhart, J. J. Steil, Neurally imprinted stable vector fields., in: ESANN, Citeseer, 2013.

[17] Y.-C. Chang, N. Roohi, S. Gao, Neural lyapunov control, Advances in neural information processing systems 32 (2019).

[18] J. Z. Kolter, G. Manek, Learning stable deep dynamics models, Advances in neural information processing systems 32 (2019).

[19] Z. Jin, W. Si, A. Liu, W.-A. Zhang, L. Yu, C. Yang, Learning a flexible neural energy function with a unique minimum for globally stable and accurate demonstration learning, IEEE Transactions on Robotics 39 (6) (2023) 4520–4538.

[20] H. Ravichandar, A. S. Polydoros, S. Chernova, A. Billard, Recent advances in robot learning from demonstration, Annual review of control, robotics, and autonomous systems 3 (1) (2020) 297–330.

[21] B. Zheng, S. Verma, J. Zhou, I. W. Tsang, F. Chen, Imitation learning: Progress, taxonomies and challenges, IEEE Transactions on Neural Networks and Learning Systems (2022).

[22] C. Dawson, S. Gao, C. Fan, Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control, IEEE Transactions on Robotics 39 (3) (2023) 1749–1767.

[23] M. Tavassoli, S. Katyara, M. Pozzi, N. Deshpande, D. G. Caldwell, D. Prattichizzo, Learning skills from demonstrations: A trend from motion primitives to experience abstraction, IEEE Transactions on Cognitive and Developmental Systems (2023).

[24] S. Khansari-Zadeh, A. Billard, Handwriting human motion dataset, lasa data, https://github.com/justagist/pyLasaDataset (2015).

[25] M. Seeger, Gaussian processes for machine learning, International journal of neural systems 14 (02) (2004) 69–106.

[26] C. M. Bishop, Pattern recognition and machine learning, Springer google schola 2 (2006) 645–678.

[27] S. Vijayakumar, A. D'souza, S. Schaal, Incremental online learning in high dimensions, Neural computation 17 (12) (2005) 2602–2634.

[28] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (2004) 199–222.

[29] H. K. Khalil, Nonlinear control, Pearson Press (2015).

[30] H. K. Khalil, Lyapunov stability, Control systems, robotics and automation 12 (2009) 115.

[31] Z. Jin, D. Qin, A. Liu, W.-A. Zhang, L. Yu, Learning neural-shaped quadratic lyapunov function for stable, accurate and generalizable human–robot skills transfer, Robotics and Computer-Integrated Manufacturing 82 (2023) 102526.

[32] A. Sultangazin, L. Pannocchi, L. Fraile, P. Tabuada, Learning to control known feedback linearizable systems from demonstrations, IEEE Transactions on Automatic Control 69 (1) (2023) 189–201.

[33] A. Sayantan, H. Jakob, S. Matteo, Handwriting human motion dataset, heelo world data, https://github.com/sayantanauddy/clfd/-tree/main/datasets (2023).

[34] N. Perrin, P. Schlehuber-Caissier, Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems, Systems & Control Letters 96 (2016) 51–59.

[35] T. Eiter, H. Mannila, Computing discrete fréchet distance (1994).

[36] H. Beik-Mohammadi, S. Hauberg, G. Arvanitidis, N. Figueroa, G. Neumann, L. Rozo, Neural contractive dynamical systems, arXiv preprint arXiv:2401.09352 (2024).

[37] M. Saveriano, An energy-based approach to ensure the stability of learned dynamical systems, in: 2020 IEEE International conference on robotics and automation (ICRA), 2020, pp. 4407–4413.

[38] S. Jin, Z. Wang, Y. Ou, W. Feng, Learning accurate and stable dynamical system under manifold immersion and submersion, IEEE Transactions on Neural Networks and Learning Systems 30 (12) (2019) 3598–3610.

[39] S. Jin, Z. Wang, Y. Ou, Y. Zhou, Improved learning accuracy for learning stable control from human demonstrations, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 2679–2685.

[40] Y. Zhang, L. Cheng, R. Cao, H. Li, C. Yang, A neural network based framework for variable impedance skills learning from demonstrations, Robotics and Autonomous Systems 160 (2023) 104312.

[41] R. Pérez-Dattari, C. Della Santina, J. Kober, Puma: deep metric imitation learning for stable motion primitives, Advanced Intelligent Systems 6 (11) (2024) 2400144.

[42] J. Lai, Z. Wu, Z. Ren, C. Chen, Q. Tan, S. Xie, A lyapunov-based framework for trajectory planning of wheeled vehicle using imitation learning, IEEE Transactions on Automation Science and Engineering 22 (2025) 12118–12133.

[43] M. Saveriano, D. Lee, Incremental skill learning of stable dynamical systems, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 6574–6581.

[44] N. B. Figueroa Fernandez, A. Billard, A physically-consistent bayesian non-parametric mixture model for dynamical system learning, CoRL (2018) 927–946.

[45] A. Billard, S. Mirrazavi, N. Figueroa, Learning for adaptive and reactive robot control: a dynamical systems approach, Mit Press, 2022.

[46] S. Sun, H. Gao, T. Li, N. Figueroa, Directionality-aware mixture model parallel sampling for efficient linear parameter varying dynamical system learning, IEEE Robotics and Automation Letters 9 (7) (2024) 6248–6255.

[47] J. Umlauft, A. Lederer, S. Hirche, Learning stable gaussian process state space models, in: 2017 American Control Conference (ACC), IEEE, 2017, pp. 1499–1504.

[48] J. Umlauft, S. Hirche, Learning stochastically stable gaussian process state–space models, IFAC Journal of Systems and Control 12 (2020) 100079.

[49] J. Duan, Y. Ou, J. Hu, Z. Wang, S. Jin, C. Xu, Fast and stable learning of dynamical systems based on extreme learning machine, IEEE Transactions on Systems, Man, and Cybernetics: Systems 49 (6) (2017) 1175–1185.

[50] J. Hu, Z. Yang, Z. Wang, X. Wu, Y. Ou, Neural learning of stable dynamical systems based on extreme learning machine, in: 2015 IEEE International Conference on Information and Automation, 2015, pp. 306–311.

[51] B. Amos, L. Xu, J. Z. Kolter, Input convex neural networks, in: International conference on machine learning, PMLR, 2017, pp. 146–155.
[52] A. Abyaneh, M. S. Guzmán, H.-C. Lin, Globally stable neural imitation policies, arXiv preprint arXiv:2403.04118 (2024).
[53] D. Totsila, K. Chatzilygeroudis, V. Modugno, D. Hadjivelichkov, D. Kanoulas, Sensorimotor learning with stability guarantees via autonomous neural dynamic policies, IEEE Robotics and Automation Letters (2025).
[54] F. Nawaz, T. Li, N. Matni, N. Figueroa, Learning complex motion plans using neural odes with safety and stability guarantees, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 17216–17222.
[55] R. T. Chen, Y. Rubanova, J. Bettencourt, D. K. Duvenaud, Neural ordinary differential equations, Advances in neural information processing systems 31 (2018).
[56] S. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.
[57] K. Neumann, J. J. Steil, Learning robot motions with stable dynamical systems under diffeomorphic transformations, Robotics and Autonomous Systems 70 (2015) 1–15.
[58] W. Zhi, K. Liu, T. Zhang, M. Johnson-Roberson, Learning orbitally stable systems for diagrammatic teaching, in: CoRL 2023 Workshop on Learning Effective Abstractions for Planning (LEAP), 2023.
[59] L. Ardizzone, J. Kruse, S. Wirkert, D. Rahner, E. W. Pellegrini, R. S. Klessen, L. Maier-Hein, C. Rother, U. Köthe, Analyzing inverse problems with invertible neural networks, arXiv preprint arXiv:1808.04730 (2018).
[60] J. Urain, M. Ginesi, D. Tateo, J. Peters, Imitationflow: Learning deep stable stochastic dynamic systems by normalizing flows, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2020, pp. 5231–5237.
[61] J. Urain, D. Tateo, J. Peters, Learning stable vector fields on lie groups, IEEE Robotics and Automation Letters 7 (4) (2022) 12569–12576.
[62] M. A. Rana, A. Li, D. Fox, B. Boots, F. Ramos, N. Ratliff, Euclideanizing flows: Diffeomorphic reduction for learning stable dynamical systems, in: Learning for Dynamics and Control, PMLR, 2020, pp. 630–639.
[63] H. Zhang, L. Cheng, Y. Zhang, Learning robust point-to-point motions adversarially: A stochastic differential equation approach, IEEE Robotics and Automation Letters 8 (4) (2023) 2357–2364.
[64] B. Schölkopf, C. J. Burges, A. J. Smola, Advances in kernel methods: support vector learning, MIT press, 1999.
[65] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.
[66] M. Saveriano, F. J. Abu-Dakka, V. Kyrki, Learning stable robotic skills on riemannian manifolds, Robotics and Autonomous Systems 169 (2023) 104510.
[67] F. Khadivar, I. Lauzana, A. Billard, Learning dynamical systems with bifurcations, Robotics and Autonomous Systems 136 (2021) 103700.
[68] D. A. Cohn, Z. Ghahramani, M. I. Jordan, Active learning with statistical models, Journal of artificial intelligence research 4 (1996) 129–145.
[69] F. Khadivar, I. Lauzana, A. Billard, Learning dynamical systems with bifurcations, Robotics and Autonomous Systems 136 (2021) 103700.
[70] J. Zhang, H. B. Mohammadi, L. Rozo, Learning riemannian stable dynamical systems via diffeomorphisms, in: 6th Annual Conference on Robot Learning, 2022.
[71] Y. Zhang, Y. Zou, H. Li, H. Zhang, L. Cheng, Learning a stable dynamic system with a lyapunov energy function for demonstratives using neural networks, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 17223–17228.
[72] W. Zhi, H. Tang, T. Zhang, M. Johnson-Roberson, Teaching periodic stable robot motion generation via sketch, IEEE Robotics and Automation Letters 10 (2) (2025) 1154–1161.
[73] Z. Jin, A. Liu, W.-A. Zhang, L. Yu, C. Yang, Learning an autonomous dynamic system to encode periodic human motion skills, IEEE Transactions on Neural Networks and Learning Systems (2024) 1–7.
[74] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real nvp, arXiv preprint arXiv:1605.08803 (2016).
[75] A. Rahimi, B. Recht, Random features for large-scale kernel machines, Advances in neural information processing systems 20 (2007).
[76] J. Ho, X. Chen, A. Srinivas, Y. Duan, P. Abbeel, Flow++: Improving flow-based generative models with variational dequantization and architecture design, in: International conference on machine learning, PMLR, 2019, pp. 2722–2730.
[77] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, J.-H. Jacobsen, Invertible residual networks, in: International conference on machine learning, PMLR, 2019, pp. 573–582.
[78] C.-W. Huang, D. Krueger, A. Lacoste, A. Courville, Neural autoregressive flows, in: International Conference on Machine Learning, PMLR, 2018, pp. 2078–2087.
[79] S.-I. Amari, Natural gradient works efficiently in learning, Neural computation 10 (2) (1998) 251–276.
[80] J. M. Lee, Riemannian manifolds: an introduction to curvature, Vol. 176, Springer Science & Business Media, 2006.
[81] H. Zhang, L. Cheng, Y. Zhang, Y. Wang, Neural liénard system: learning periodic manipulation skills through dynamical systems, Science China Information Sciences 67 (12) (2024) 1–16.
[82] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, D. Fox, Riemannian motion policies, arXiv preprint arXiv:1801.02854 (2018).
[83] B. Fichera, A. Billard, Learning dynamical systems encoding nonlinearity within space curvature, arXiv preprint arXiv:2403.11948 (2024).
[84] S. Dutta, L. Behera, S. Nahavandi, Skill learning from human demonstrations using dynamical regressive models for multitask applications, IEEE Transactions on Systems, Man, and Cybernetics: Systems 51 (1) (2018) 659–672.
[85] W. Wang, M. Saveriano, F. J. Abu-Dakka, Learning deep robotic skills on riemannian manifolds, IEEE Access 10 (2022) 114143–114152.
[86] J. Zhang, Z. Jin, Z. Zhao, C. Yang, A novel robotic skill learning approach for assembly task with dynamical system and broad learning, IEEE Transactions on Industrial Electronics (2025) 1–10.
[87] W. Si, Z. Jin, Z. Lu, N. Wang, C. Yang, A stable guidance method for teleoperation-based robot learning from demonstration, in: 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE), 2024, pp. 2376–2381.
[88] Z. Jin, A. Liu, W.-a. Zhang, L. Yu, C. Yang, Learning stable state-dependent variable impedance control for compliant manipulation, IEEE/ASME Transactions on Mechatronics (2024) 1–12.
[89] Y. Huang, W. Xiao, C. Wang, H. Liu, R. Huang, Z. Sun, Towards fully autonomous ultrasound scanning robot with imitation learning based on clinical protocols, IEEE Robotics and Automation Letters 6 (2) (2021) 3671–3678.
[90] D. Ge, H. Zhao, Y. Wang, D. Li, X. Li, H. Ding, Learning compliant dynamical system from human demonstrations for stable force control in unknown environments, Robotics and Computer-Integrated Manufacturing 86 (2024) 102669.
[91] G. Franzese, R. Prakash, J. Kober, Generalization of task parameterized dynamical systems using gaussian process transportation, arXiv preprint arXiv:2404.13458 (2024).
[92] G. Franzese, L. d. S. Rosa, T. Verburg, L. Peternel, J. Kober, Interactive imitation learning of bimanual movement primitives, IEEE/ASME Transactions on Mechatronics 29 (5) (2024) 4006–4018.
[93] S. M. Khansari-Zadeh, O. Khatib, Learning potential functions from human demonstrations with encapsulated dynamic and compliant behaviors, Autonomous Robots 41 (2017) 45–69.

[94] Z. Jin, A. Liu, W.-a. Zhang, L. Yu, An optimal variable impedance control with consideration of the stability, IEEE Robotics and Automation Letters 7 (2) (2022) 1737–1744.

[95] X. Chen, Y. Michel, D. Lee, Closed-loop variable stiffness control of dynamical systems, in: 2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids), 2021, pp. 163–169.

[96] Z. Jin, D. Qin, A. Liu, W.-A. Zhang, L. Yu, Constrained variable impedance control using quadratic programming, in: 2022 International Conference on Robotics and Automation (ICRA), 2022, pp. 8319–8324.

[97] Z. Jin, D. Qin, A. Liu, W.-a. Zhang, L. Yu, Model predictive variable impedance control of manipulators for adaptive precision-compliance tradeoff, IEEE/ASME Transactions on Mechatronics 28 (2) (2022) 1174–1186.

[98] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, P. Tabuada, Control barrier functions: Theory and applications, in: 2019 18th European control conference (ECC), 2019, pp. 3420–3431.

[99] M. Saveriano, D. Lee, Distance based dynamical system modulation for reactive avoidance of moving obstacles, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 5618–5623.

[100] S. M. Khansari-Zadeh, A. Billard, A dynamical system approach to realtime obstacle avoidance, Autonomous Robots 32 (2012) 433–454.

[101] L. Huber, A. Billard, J.-J. Slotine, Avoidance of convex and concave obstacles with convergence ensured through contraction, IEEE Robotics and Automation Letters 4 (2) (2019) 1462–1469.

[102] Z. Shen, M. Saveriano, F. J. Abu-Dakka, S. Haddadin, Safe execution of learned orientation skills with conic control barrier functions, in: 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024, pp. 13376–13382.

[103] B. Li, S. Wen, Z. Yan, G. Wen, T. Huang, A survey on the control lyapunov function and control barrier function for nonlinear-affine control systems, IEEE/CAA Journal of Automatica Sinica 10 (3) (2023) 584–602.

[104] A. Coulombe, H.-C. Lin, Generating stable and collision-free policies through lyapunov function learning, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 3037–3043.

[105] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, S. Schaal, Dynamical movement primitives: learning attractor models for motor behaviors, Neural computation 25 (2) (2013) 328–373.

[106] A. Paraschos, C. Daniel, J. R. Peters, G. Neumann, Probabilistic movement primitives, Advances in neural information processing systems 26 (2013).

[107] Z. Jin, A. Liu, W.-a. Zhang, L. Yu, C. Yang, Gaussian process movement primitive, Automatica 155 (2023) 111120.

[108] Y. Huang, L. Rozo, J. Silvério, D. G. Caldwell, Kernelized movement primitives, The International Journal of Robotics Research 38 (7) (2019) 833–852.

[109] J. Fu, Z. Jin, A. Liu, W.-A. Zhang, L. Yu, Non-parametric gaussian process movement primitive with via-point constraint for effective and safe robot skill learning, Neurocomputing 589 (2024) 127711.

[110] D. Sun, Q. Liao, A. Loutfi, Type-2 fuzzy model-based movement primitives for imitation learning, IEEE Transactions on Robotics 38 (4) (2022) 2462–2480.

[111] C. Yang, C. Chen, W. He, R. Cui, Z. Li, Robot learning system based on adaptive neural control and dynamic movement primitives, IEEE transactions on neural networks and learning systems 30 (3) (2018) 777–787.

[112] C. Yang, C. Zeng, C. Fang, W. He, Z. Li, A dmps-based framework for robot learning and generalization of humanlike variable impedance skills, IEEE/ASME Transactions on Mechatronics 23 (3) (2018) 1193–1203.

[113] A. Liu, S. Zhan, Z. Jin, W.-A. Zhang, A variable impedance skill learning algorithm based on kernelized movement primitives, IEEE Transactions on Industrial Electronics 71 (1) (2024) 870–879.

[114] A. Liu, J. Fu, S. Zhan, Z. Jin, W.-a. Zhang, A policy searched-based optimization algorithm for obstacle avoidance in robot manipulators, IEEE Transactions on Industrial Electronics 71 (9) (2024) 11262–11271.

[115] P. M. Scheikl, N. Schreiber, C. Haas, N. Freymuth, G. Neumann, R. Lioutikov, F. Mathis-Ullrich, Movement primitive diffusion: Learning gentle robotic manipulation of deformable objects, IEEE Robotics and Automation Letters 9 (6) (2024) 5338–5345.

[116] J. Fong, M. Tavakoli, Kinesthetic teaching of a therapist's behavior to a rehabilitation robot, in: 2018 international symposium on medical robotics (ISMR), IEEE, 2018, pp. 1–6.

[117] S. Ekvall, D. Kragic, Robot learning from demonstration: A task-level planning approach, International Journal of Advanced Robotic Systems 5 (3) (2008) 33.

[118] D. H. Grollman, O. C. Jenkins, Incremental learning of subtasks from unsegmented demonstration, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 261–266.

[119] J. R. Medina, A. Billard, Learning stable task sequences from demonstration with linear parameter varying systems and hidden markov models, in: Proceedings of the 1st Annual Conference on Robot Learning, Vol. 78, 2017, pp. 175–184.

[120] A. Gupta, V. Kumar, C. Lynch, S. Levine, K. Hausman, Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning, arXiv preprint arXiv:1910.11956 (2019).

[121] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, L. Fei-Fei, Learning to generalize across long-horizon tasks from human demonstrations, arXiv preprint arXiv:2003.06085 (2020).

[122] S. Pirk, K. Hausman, A. Toshev, M. Khansari, Modeling long-horizon tasks as sequential interaction landscapes, arXiv preprint arXiv:2006.04843 (2020).

[123] P. Pastor, M. Kalakrishnan, L. Righetti, S. Schaal, Towards associative skill memories, in: 2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012), 2012, pp. 309–315.

[124] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, J. Peters, Towards learning hierarchical skills for multi-phase manipulation tasks, in: 2015 IEEE international conference on robotics and automation (ICRA), 2015, pp. 1503–1510.

[125] B. Akgun, M. Cakmak, J. W. Yoo, A. L. Thomaz, Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective, in: Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction, 2012, pp. 391–398.

[126] C. Pérez-D'Arpino, J. A. Shah, C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 4058–4065.

[127] G. Konidaris, S. Kuindersma, R. Grupen, A. Barto, Robot learning from demonstration by constructing skill trees, The International Journal of Robotics Research 31 (3) (2012) 360–375.

[128] S. Niekum, S. Chitta, A. G. Barto, B. Marthi, S. Osentoski, Incremental semantically grounded learning from demonstration., in: Robotics: Science and Systems, Vol. 9, Berlin, Germany, 2013, pp. 10–15607.

[129] N. B. Figueroa Fernandez, From high-level to low-level robot learning of complex tasks: Leveraging priors, metrics and dynamical systems, Tech. rep., EPFL (2019).

[130] A. Billard, S. Mirrazavi, N. Figueroa, Learning for adaptive and reactive robot control: a dynamical systems approach, Mit Press, 2022.

[131] W. Wang, C. Zeng, H. Zhan, C. Yang, A novel robust imitation learning framework for complex skills with limited demonstrations, IEEE Transactions on Automation Science and Engineering (2024) 1–13.

[132] R. Ramrakhya, E. Undersander, D. Batra, A. Das, Habitat-web: Learning embodied object-search strategies from human demonstrations at scale, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 5163–5173.

[133] W. Kim, B. Son, I. Kim, Vilt: Vision-and-language transformer without convolution or region supervision, in: International conference on machine learning, 2021, pp. 5583–5594.

[134] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International conference on machine learning, 2021, pp. 8748–8763.

[135] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, A. Mian, A comprehensive overview of large language models, arXiv preprint arXiv:2307.06435 (2023).

[136] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, S. Song, Diffusion policy: Visuomotor policy learning via action diffusion, The International Journal of Robotics Research (2023) 02783649241273668.