2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Learning Dynamical System-Based Robot Motions from Demonstrations via ODE-Driven Diffeomorphic Mappings

Haoyu Zhang and Long Cheng

Abstract— **Learning from Demonstrations (LfD) has emerged as a prominent paradigm for imparting motion skills to robotic systems. Dynamical systems (DS) offer a potent mathematical framework for representing point-to-point motions, a critical requirement for numerous practical applications in robotics. While existing approaches typically construct DS models by employing diffeomorphic mappings to morph stable reference systems toward observed demonstrations, the requirement to preserve strict diffeomorphic properties introduces architectural constraints on neural network design, thereby constraining their expressiveness. To address this limitation, we present a DS-based LfD formulation that relaxes traditional diffeomorphism constraints. Our framework employs bidirectional temporal integration of ordinary differential equations (ODEs) to simultaneously satisfy stability guarantees and trajectory alignment objectives. A key innovation lies in a variational calculus framework for Jacobian estimation, enabling efficient computation of DS vector fields while maintaining numerical stability. Comprehensive evaluations demonstrate that our method achieves 33.7% improvement in trajectory reproduction accuracy compared to state-of-the-art baselines while preserving Lyapunov stability. The proposed methodology significantly expands the representational capacity of DS-based learning systems, enabling robust reproduction of complex motion patterns.**

I. Introduction

Learning from Demonstrations (LfD) has become a cornerstone methodology for robotic skill acquisition, utilizing sequences of state-action pairs captured from expert demonstrations to synthesize policies that replicate observed behaviors [1]. This paradigm proves valuable in scenarios where manual policy engineering is infeasible or where environmental complexity renders reward function specification intractable [2]. Beyond autonomous manipulators, acquiring human-like motion skills is also pivotal for developing transparent interactions in wearable exoskeletons and assistive devices [3], [4].

A critical challenge in LfD lies in encoding the inherent regularity of biological motion-properties such as Lipschitz continuity, smoothness, and energy boundedness-into learned policies. Such biological characteristics are heavily investigated in musculoskeletal systems to extract underlying motion synergies and personalized kinematics [5], [6]. In the context of robotic trajectories, a stable Dynamical System (DS) is well-suited to this task, as it naturally generates continuous and bounded outputs. Dynamical systems offer an elegant mathematical framework for this purpose, inherently guaranteeing continuous, bounded trajectories while naturally modeling the point-to-point motion primitives that constitute fundamental human movement patterns [7].

Recent advances in DS-based LfD predominantly employ diffeomorphic transformations, where invertible neural architectures (particularly normalizing flow models [8]) morph a predefined stable system to align with demonstration data [9]. While theoretically appealing, these approaches face practical limitations: The strict architectural constraints required to preserve diffeomorphism (bijective differentiability with smooth inverse) severely restrict network expressivity. Furthermore, the computational burden of maintaining exact Jacobian invertibility through automatic differentiation introduces inefficiency and scalability challenges.

This work introduces a novel approach to DS-based motion learning that transcends conventional diffeomorphism constraints. We cast the diffeomorphic mapping for DS-based LfD as a continuous-time neural flow and train it via *bidirectional* temporal integration. Rather than enforcing invertibility at the architectural (layer) level, we preserve diffeomorphism at the *trajectory* level by integrating a Lipschitz-controlled velocity field. Coupled with a variational ODE for the Jacobian, this yields stable and expressive task-space dynamics with substantially reduced Jacobian-evaluation overhead, enabling the use of arbitrary Lipschitz-constrained networks while preserving stability guarantees. Collectively, enforcing strict diffeomorphic architectures tends to bind model expressivity to invertibility-preserving blocks and to the availability of tractable Jacobians, which, in practice, complicates scaling to complex, nonlinear motion manifolds. Our key idea is to preserve the diffeomorphic nature not at the layer level but at the level of continuous-time flow integration, thus decoupling stability guarantees from architectural rigidity while keeping the mapping invertible by construction. Specifically, the main contributions are listed as follows:

- We introduce the method of formulating the diffeomorphic mapping problem through bidirectional temporal ODE integration to the LfD field, improving the performance of demonstration-based learning.
- We propose a variational approach for computing the Jacobian by solving an ODE, which enhances computational efficiency.
- We validate the proposed framework through compre-

This work was supported in part by the National Natural Science Foundation of China under Grants U25A20475 and 62333023, in part by the Beijing Municipal Natural Science Foundation under Grants F2024201068 and L243014, in part by CAS Project for Young Scientists in Basic Research under Grant YSBR-034, and in part by the Fundamental Research Funds for the Central Universities.

The authors are with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China, and are also with the State Key laboratory of Multimodel Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. Email: long.cheng@ia.ac.cn.

hensive numerical studies and real-world experiments on a 7-DOF Franka Emika arm, demonstrating accurate trajectory reproduction, Lyapunov-stable convergence, and robustness to perturbations.

II. Related Work

Over the past decade, numerous DS methods have been developed to enable point-to-point motion learning. Among these, Dynamic Movement Primitives (DMPs) emerged as a foundational approach, modeling motion trajectories through a spring-damper system framework to inherently guarantee stability and reject perturbations [11]. Building upon this foundation, DMPs have been recently extended with probabilistic frameworks to handle physical human-robot interactions, such as exoskeleton-assisted lifting [12]. Despite these advantages, standard DMPs exhibit limitations in accommodating temporal perturbations and constraining motion flexibility due to their rigid spring-damper assumptions, hindering their efficacy in complex tasks.

To overcome these constraints, the Stable Estimator of Dynamical Systems (SEDS) [13] eschews the spring-damper paradigm, instead leveraging Gaussian Mixture Models (GMMs) to encode demonstrations while enforcing Lyapunov stability. Although SEDS enhances generalization, its reliance on quadratic Lyapunov functions imposes a restrictive accuracy-stability trade-off, thereby compromising trajectory reproduction fidelity. Building on this, the Control Lyapunov Function-based Dynamic Movements (CLF-DM) method [14] introduces an asymmetric quadratic Lyapunov function formulation. This innovation broadens the set of stable demonstrations and improves trajectory accuracy. However, CLF-DM necessitates online resolution of constrained convex optimization problems, entailing significant computational costs that hinder real-time applications.

From a control perspective, DMPs encode stability via spring–damper templates, SEDS via Lyapunov-constrained mixture fields, and CLF-DM via online convex programs that enforce a control Lyapunov decrease. These mechanisms trade off global guarantees against model flexibility in distinct ways: as the motion manifold becomes more nonlinear or multi-modal, the template or quadratic Lyapunov restrictions often dominate the achievable fidelity. As indicated in [15] that imposing overly rigid stability requirements can degrade how well the learned dynamics reproduce the demonstrated behaviors.

Recent advances explore latent space transformations to reconcile stability and accuracy. For instance, τ-SEDS [15] employs diffeomorphic mappings to project system states into a latent space for DS learning. Parallel efforts integrate advanced function approximators, such as neural networks and models based on Reproducing Kernel Hilbert Space (RKHS), as seen in [9], [16], [17], [18], [19], [20], [21], [22]. Notably, SDS-EF [9] generalizes diffeomorphic mappings via normalizing flows, yet its assumption of linear latent-space dynamics restricts its ability to capture nonlinear, real-world motion complexities. To mitigate linearity constraints, Modified Attention-Based Dynamical Systems (MABDS) [18] adopts an attention-driven architecture. However, its nonlinear structure amplifies sensitivity to input noise, diminishing robustness in noisy settings. Stochastic DS approaches [19] address this by embedding stochastic dynamics, enhancing resilience to disturbances. Meanwhile, Neural Liénard Systems (NLS) [20] target periodic motions but inherit limitations from diffeomorphic assumptions. Collectively, these methods remain constrained by the representational rigidity of diffeomorphisms, limiting their adaptability. In contrast, our approach preserves the same Lyapunov inheritance property by treating the diffeomorphism as a neural flow generated by a Lipschitz v_t, thereby avoiding explicit invertible blocks while keeping a provably nonsingular Jacobian through the variational dynamics. This reframes the 'expressivity vs. stability' trade-off as a solver/design choice rather than an architectural constraint, paving the way for deeper networks without normalization overhead.

In summary, this work proposes a novel DS learning framework that enhances diffeomorphic flexibility through ODE-based mappings. By integrating ODE solutions into the diffeomorphism design, our approach captures intricate dynamical patterns while preserving stability, thereby advancing trajectory accuracy and generalizability.

III. Preliminary

A. Robot Motion Skill Learning via DS

We address the modeling of a robot's state $x \in \mathbb{R}^d$ through an autonomous dynamical system governed by the equation:

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

where x represents the actuator position, and $\dot{x}$ is the time derivative of x. The function $f : \mathbb{R}^d \to \mathbb{R}^d$ is a continuously differentiable vector field defining the system dynamics.

Given a set of expert demonstrations $\{x_{k,n}, \dot{x}_{k,n} \mid k = 1, 2, ..., T; n = 1, 2, ..., N\}$, where T denotes the number of time steps per trajectory and N the total number of demonstrations, our objective is to learn an optimal vector field f^* that satisfies:

$$\dot{x}_{k,n} = f^*(x_{k,n}), \quad \forall k \in \{1, 2, ..., T\}, n \in \{1, 2, ..., N\}.$$

while ensuring the dynamical system $\dot{x}(t) = f^*(x(t))$ exhibits *Lyapunov Asymptotically Stable* at the point-to-point motion target x_e.

Definition 1 (Lyapunov Asymptotic Stability): Consider an autonomous system as described in (1). Suppose f has an equilibrium at x_e such that $f(x_e) = 0$. This equilibrium is said to be *Lyapunov stable* if for every $\epsilon > 0$, there exists a $\delta > 0$ such that if $\|x(0) - x_e\| < \delta$, then for every $t \geq 0$, we have $\|x(t) - x_e\| < \epsilon$. Furthermore, the equilibrium is said to be *Lyapunov asymptotically stable* if it is Lyapunov stable and there exists a $\delta > 0$ such that if $\|x(0) - x_e\| < \delta$, then $\lim_{t\to\infty} \|x(t) - x_e\| = 0$.

B. Learning Stable Dynamics via Diffeomorphisms

Directly synthesizing a dynamical system that simultaneously satisfies Lyapunov asymptotic stability and accurately fits demonstration data presents significant challenges. To

address this, contemporary neural network-based approaches often leverage diffeomorphic mappings to morph a predefined stable system onto the demonstration space.

Formally, let $y \in \mathbb{R}^d$ represent a latent state governed by a Lyapunov asymptotically stable dynamical system:

$$\dot{y} = f_0(y), \tag{2}$$

where $f_0 \colon \mathbb{R}^d \to \mathbb{R}^d$ is a predefined stable vector field (e.g., $\dot{y} = -y$). Let $\phi \colon \mathbb{R}^d \to \mathbb{R}^d$ denote a diffeomorphic mapping satisfying $y = \phi(x)$. The induced dynamical system in the original state space becomes:

$$f(x) = \left(\frac{\partial \phi^{-1}(x)}{\partial x}\right)^{-1} f_0\big(\phi^{-1}(x)\big). \tag{3}$$

Theorem 1 ([9]): The dynamical systems defined in (3) and (2) share identical stability properties.

By Theorem 1, the system (3) inherits Lyapunov asymptotic stability from (2) provided f_0 is asymptotically stable. While Theorem 1 guarantees that stability can be transferred from the latent dynamics to the task space, practical learning hinges on how restrictive the mapping class is. If the diffeomorphism is implemented by strictly invertible blocks, the hypothesis class is implicitly tied to architectural choices; in contrast, our flow-based construction preserves diffeomorphic structure at the trajectory level. This decoupling allows one to use expressive, Lipschitz-controlled networks for the velocity field while retaining a provably nonsingular Jacobian along the flow. In other words, stability is inherited by integration, not by layer design, which enlarges the practically learnable set of stable vector fields without sacrificing the Lyapunov guarantee.

To learn the diffeomorphism ϕ, we parameterize it as ϕ_θ with trainable parameters θ. The supervised training objective minimizes the mean squared error over all demonstrations:

$$\theta^* = \arg\min_\theta \frac{1}{NT} \sum_{n=1}^{N} \sum_{k=1}^{T} \| f(x_{k,n}) - \dot{x}_{k,n} \|_2^2 . \tag{4}$$

IV. ODE-Driven Diffeomorphic Learning

While existing approaches implement the diffeomorphic mapping ϕ_θ in (3) via flow models [9], [16], [19], [20], we propose a framework leveraging ordinary differential equation integration for both forward and inverse mappings. This section details our methodology.

A. Diffeomorphic Mapping via Neural ODE Flows

Let $v : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ denote a Lipschitz-continuous time-dependent vector field. We construct a diffeomorphic flow map $\phi_t : \mathbb{R}^d \to \mathbb{R}^d$ by solving the initial value problem:

$$\begin{aligned} \frac{d}{dt}\phi_t(x) &= v_t(\phi_t(x)), \\ \phi_0(x) &= x, \end{aligned} \tag{5}$$

where $t \in [0,1]$ parameterizes the flow trajectory. Given a predefined asymptotically stable latent dynamics $\dot{y} = f_0(y)$, we derive the task-space dynamical system via the inverse flow map:

$$f_t(x) = \left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1} f_0(\phi_t^{-1}(x)). \tag{6}$$

During training we use the flow parameter $t \in [0,1]$ to construct an invertible mapping ϕ_t. At test time, the deployed dynamics is autonomous and defined by the terminal map,

$$f(x) \;=\; \big(\partial \phi_1^{-1}(x)/\partial x\big)^{-1} f_0\big(\phi_1^{-1}(x)\big), \tag{7}$$

where we set $t{=}1$ for execution.

B. Efficient Jacobian Computation via Variational Equations

A key challenge in implementing (6) lies in efficiently computing the inverse Jacobian term $\left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1}$. Direct evaluation via automatic differentiation proves computationally prohibitive. We instead leverage the variational equation formalism to derive an analytical solution.

Theorem 2 (Variational Jacobian Dynamics): Let $J_t(y) = \frac{\partial \phi_t(y)}{\partial y}$ denote the Jacobian of the flow map. Then:

$$\frac{d}{dt} J_t(y) = Dv_t(\phi_t(y)) \cdot J_t(y), \quad J_0(y) = I, \tag{8}$$

where Dv_t is the Jacobian of v_t. Consequently, $\left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1} = J_t(\phi_t^{-1}(x))$.

Proof: By the inverse function theorem, $\frac{\partial \phi_t^{-1}(x)}{\partial x} = \big[J_t(\phi_t^{-1}(x))\big]^{-1}$. Differentiating $J_t(y)$ with respect to time:

$$\begin{aligned} \frac{d}{dt} J_t(y) &= \frac{\partial}{\partial y}\left(\frac{d}{dt}\phi_t(y)\right) \\ &= \frac{\partial}{\partial y} v_t(\phi_t(y)) \\ &= Dv_t(\phi_t(y)) \cdot J_t(y). \end{aligned}$$

The identity $\left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1} = J_t(\phi_t^{-1}(x))$ follows directly. ■

Theorem 2 enables efficient Jacobian computation by solving (8) concurrently with (5) during ODE integration, eliminating the need for explicit matrix inversion. The total procedure of computing the state derivative in (1), which effectively serves as the pseudo-algorithm for the deployment phase mapping the current state to the desired velocity command, is outlined in Algorithm 1.

C. Flow Invertibility and Diffeomorphism Conditions

We recall the flow $\phi_t : \mathbb{R}^d \to \mathbb{R}^d$ induced by the time-varying vector field v_t

$$\frac{d}{dt}\,\phi_t(x) = v_t(\phi_t(x)), \qquad \phi_0(x) = x, \tag{9}$$

and its Jacobian $J_t(y) = \partial \phi_t(y)/\partial y$ governed by the variational ODE

$$\frac{d}{dt} J_t(y) = Dv_t(\phi_t(y))\, J_t(y), \qquad J_0(y) = I. \tag{10}$$

(a). Existence & Uniqueness. If $v_t(\cdot)$ is (uniformly in $t \in [0,1]$) locally Lipschitz in x and continuous in t, then

Algorithm 1 Steps to Compute $f_t(x)$

Input: x

Output: $f_t(x) = \left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1} f_0(\phi_t^{-1}(x))$

Step 1: Compute the inverse flow $\phi_t^{-1}(x)$

Solve the backward-time ODE:

$$\frac{d}{ds}\psi_s(x) = -v_{t-s}(\psi_s(x)), \psi_0(x) = x, \text{for } s \in [0,t],$$

where $\phi_t^{-1}(x) = \psi_t(x)$. This gives $y = \phi_t^{-1}(x)$.

Step 2: Solve the variational equation for $J_t(y)$

Along the trajectory $y \to \phi_t(y) = x$, compute the Jacobian $J_t(y)$ by solving:

$$\frac{d}{dt}J_t(y) = Dv_t(\phi_t(y)) \cdot J_t(y), \quad J_0(y) = I,$$

where Dv_t is the Jacobian of v_t. Integrate this equation from $t = 0$ to t with initial condition $y = \phi_t^{-1}(x)$.

Step 3: Evaluate $J_t(y)$ **at** $y = \phi_t^{-1}(x)$

The result $J_t(\phi_t^{-1}(x)) = \frac{\partial \phi_t(y)}{\partial y}\Big|_{y=\phi_t^{-1}(x)}$ is the desired term:

$$\left(\frac{\partial \phi_t^{-1}(x)}{\partial x}\right)^{-1} = J_t(\phi_t^{-1}(x)).$$

Step 4: Compute $f_t(x)$

The final dynamical system is:

$$f_t(x) = J_t(\phi_t^{-1}(x)) f_0(\phi_t^{-1}(x))$$

for any x_0 there exists a unique maximal solution $\phi_t(x_0)$ to (9) on $t \in [0,1]$ by Picard–Lindelöf. Hence ϕ_t defines a homeomorphism family.

(b). Non-singularity of J_t (Liouville's formula). Taking determinants on (10) and using Jacobi's formula,

$$\frac{d}{dt}\det J_t(y) = \operatorname{tr}\big(Dv_t(\phi_t(y))\big)\ \det J_t(y). \tag{11}$$

With $\det J_0(y) = 1$, we get

$$\det J_t(y) = \exp\Big(\int_0^t \operatorname{tr}\big(Dv_s(\phi_s(y))\big)\, ds\Big) \neq 0, \quad \forall t \in [0,1]. \tag{12}$$

Therefore $J_t(y)$ is invertible for all t in the integration horizon.

(c). Local Diffeomorphism and Global Invertibility. Since J_t is nonsingular, ϕ_t is a local diffeomorphism by the inverse function theorem. Global injectivity follows from uniqueness of solutions: if $\phi_t(x_1) = \phi_t(x_2)$ for some t, then integrating (9) backward yields $x_1 = x_2$. Surjectivity holds because ϕ_t is a continuous bijection with continuous inverse given by the backward flow. Hence, for each $t \in [0,1]$, ϕ_t is a diffeomorphism.

V. Empirical Evaluation

We evaluate our method's accuracy in modeling point-to-point motions across three key aspects: numerical integration stability, architectural choices, and trajectory reproduction fidelity. All demonstrations are normalized to $[-0.5, 0.5]$ to ensure scale-invariant hyperparameter tuning. Optimization uses Adam [23] with default parameters.

A. Ablation Study: Diffeomorphic Mapping Properties

We first validate the diffeomorphic properties of our transformation function by analyzing the impact of ODE solvers and neural architectures on reconstruction accuracy.

ODE Solver Analysis: We test ten solvers (five adaptive-step, five fixed-step) using a three-layer fully-connected network (512 hidden units) for $v_t(\cdot)$. Adaptive methods use a tolerance of 10^{-6}, fixed methods 10^{-15}. As shown in Fig. 1, all solvers except Euler achieve reconstruction errors below 10^{-3} in normalized units (data scaled to $[-0.5, 0.5]$). Fig. 2 reveals fixed-step solvers (e.g., RK4, Midpoint) reduce computation time versus adaptive methods (e.g, DOPRI). In practice we adopt a fixed-step RK4 (3/8 rule) for both the state flow and the variational Jacobian. The shared step sequence ensures synchronized evaluations of v_t and Dv_t, which stabilizes the coupled integration numerically while delivering fourth-order local accuracy without the overhead of error-controlled adaptivity

Architecture Comparison: We analyze three architectures for $v_t(\cdot)$:

- **FC-n**: n-layer fully-connected network (default n=3, 512 units/layer)
- **ResNet-n**: n-block residual network (default n=8, 512 hidden dim)
- **ResNet-LN**: ResNet with layer normalization (default n=8, 512 hidden dim)

Fig. 3 shows FC-3 achieves lowest reconstruction error (8.7×10^{-7}) but with lower learning ability. ResNet-8 balances expressivity (3.6×10^{-5} error) and temporal consistency, while LN increases error (3.4×10^{-4}). We adopt ResNet as it enables stable gradient flow through deep networks without normalization overhead.

B. Jacobian Computation Efficiency

We conduct a comprehensive comparison between our variational Jacobian method and conventional automatic differentiation (autograd) to evaluate numerical precision and computational efficiency.

As quantified in Fig. 4, our method exhibits a mean relative eigenvalue error of 1.7×10^{-2} and a mean relative eigenvector angular deviation of 7.8×10^{-2} radians compared to autograd benchmarks. This marginal discrepancy occurs alongside a 20.8% reduction in computation time, significantly enhancing feasibility for real-time control scenarios.

The eigenvector visualization (Fig. 4, bottom row) further validates our method's numerical fidelity. Sampled eigenvectors derived from both approaches demonstrate close alignment, with angular deviations consistently below 2°. This precision is particularly noteworthy given the inherent trade-off between computational speed and numerical accuracy in Jacobian approximation methods.

Complexity Note: Variational Jacobian vs. Autograd Let d be the state dimension. A naive autograd computation of $\left(\partial \phi_t^{-1}/\partial x\right)^{-1}$ typically involves d reverse-mode sweeps

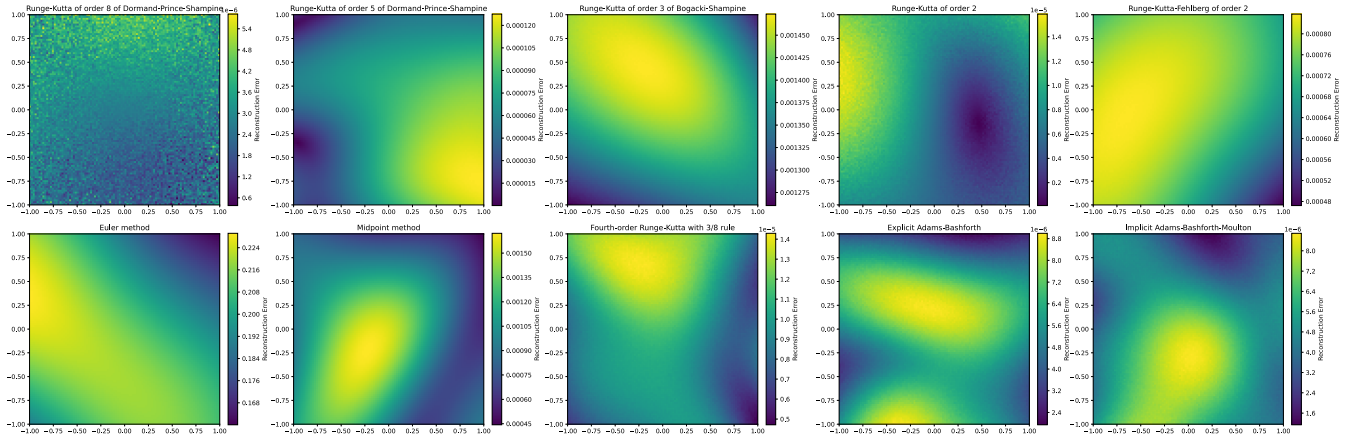

Fig. 1. Reconstruction errors of diffeomorphic mappings using adaptive-step (top) and fixed-step (bottom) ODE solvers. Lower errors (blue) indicate better preservation of diffeomorphic properties.

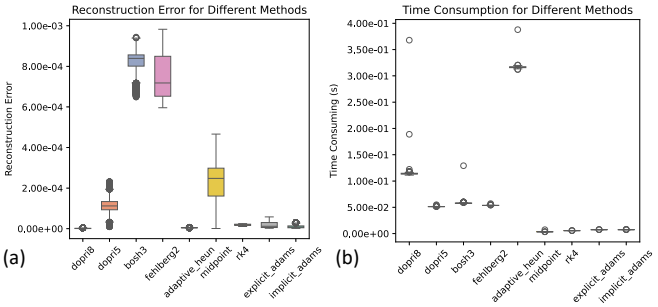

Fig. 2. Reconstruction error (a) and computational time (b) across ODE solvers. Fixed-step methods achieve faster computation while maintaining sub-millimeter accuracy.

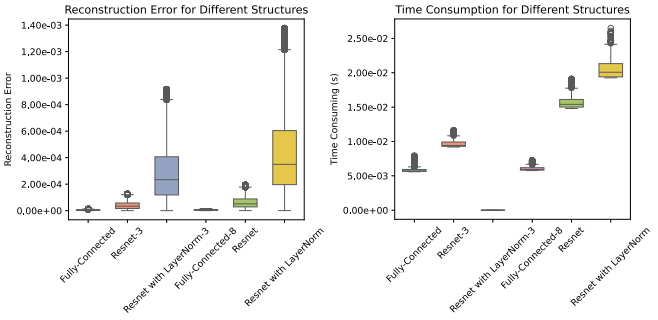

Fig. 3. Reconstruction error (left) and inference time (right) across network architectures. Residual networks (ResNet-n) balance representational capacity and efficiency.

(one per column) or explicit matrix inversions, with effective cost $\mathcal{O}(d)$ evaluations of the flow plus an $\mathcal{O}(d^3)$ inversion. In contrast, integrating the matrix ODE (10) advances the full $d \times d$ Jacobian in lockstep with the state ODE, incurring per-step cost $\mathcal{O}(d^2)$ for the matrix–matrix product $Dv_t\, J_t$. Over N_{step} fixed steps, the asymptotic costs are

$$\begin{aligned} \text{Autograd:} \quad & \mathcal{O}\big(N_{\text{step}} \cdot d \cdot C_{\text{flow}}\big) + \mathcal{O}(d^3), \\ \text{Variational:} \quad & \mathcal{O}\big(N_{\text{step}} \cdot (d^2 + C_{\text{flow}})\big), \end{aligned} \tag{13}$$

where C_{flow} denotes the cost of one v_t evaluation. For

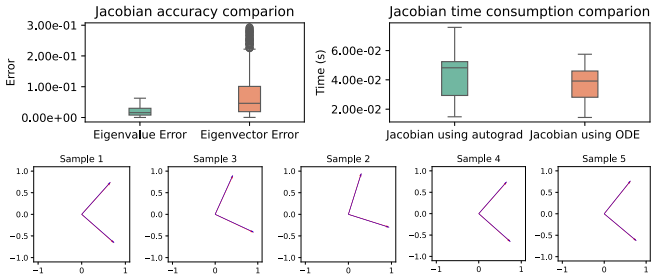

Fig. 4. Comparative analysis of numerical precision and computational efficiency. **(Top left):** Relative eigenvalue discrepancies between variational and autograd methods. **(Top right):** Computation time comparison. **(Bottom row):** Sampled eigenvectors (blue: proposed method; red: autograd), demonstrating close alignment. Our approach achieves sub-10^{-2} error levels while reducing latency by 20.8%, critical for real-time robotic applications.

moderate d and suitably optimized kernels, the latter avoids repeated reverse sweeps and the terminal matrix inversion, explaining the latency reduction observed empirically.

C. Trajectory Reproduction on LASA Dataset

We benchmark our method against SDS-EF [9] and DMP [11] using the LASA dataset [24]. For SDS-EF, we retain the default configuration, while for DMP, we set the number of basis functions to 15. The time-dependent vector field $v_t(\cdot)$ in (5) is implemented as a 10-block ResNet with 512 hidden units per layer [25], with time-state concatenation as input. We evaluate two metrics:

(i.) **Mean Square Error (MSE)**:

$$\text{MSE} = \frac{1}{T}\sum_{t=1}^{T} \|x_r^t - \Phi(x_0, t)\|_2^2, \tag{14}$$

where $\Phi(x_0, t)$ is the evolution function, x_r^t denotes the demonstrations at the time step t, T is the final time step.

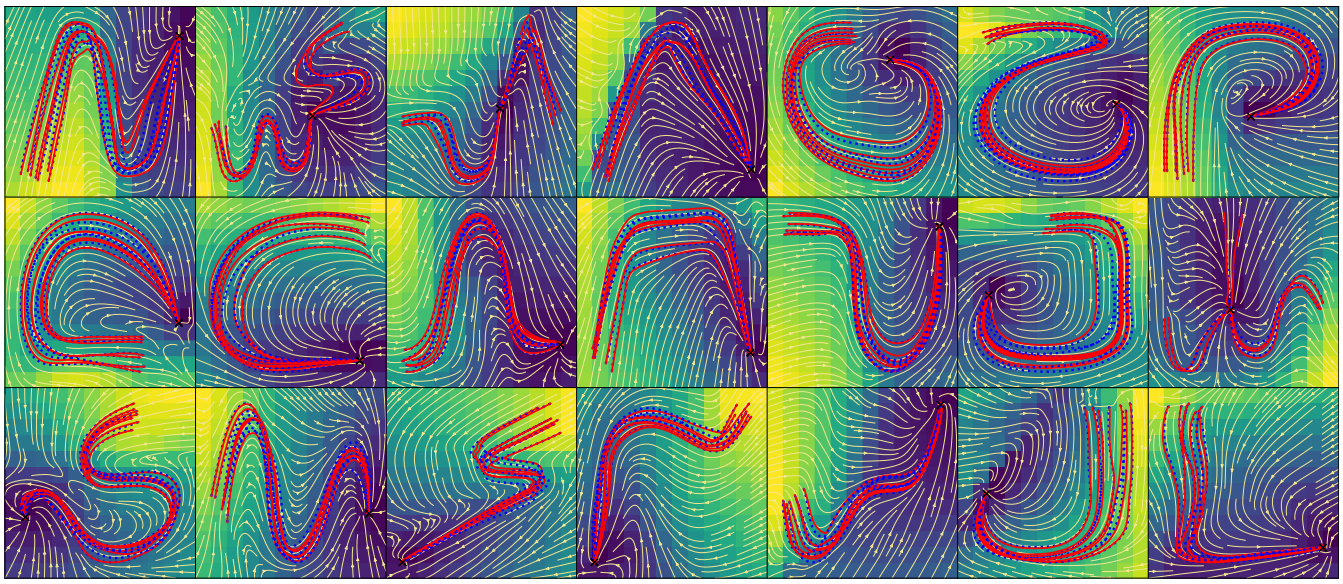

Fig. 5. Learned vector fields (background color gradient) and trajectory comparisons for 21 LASA shapes. Demonstrations (blue dashed) versus reproductions (red solid) show precise tracking while maintaining stability. Velocity magnitudes range from low (blue) to high (yellow).

(ii.) **Swept Error Area (SEA)**:

$$\text{SEA} = \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{S}(\Phi(x_0, t), \Phi(x_0, t+1), x_r^t, x_r^{t+1}), \tag{15}$$

where $\mathbb{S}(p_1, p_2, p_3, p_4) : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$ refers to the area of the tetragon made up of the four points p_1, p_2, p_3 and p_4.

Fig. 5 visualizes learned vector fields and trajectories for 21 LASA shapes, along with the reproduced trajectories generated by our proposed method. The results demonstrate close alignment between reproductions (red) and demonstrations (blue), with smooth velocity profiles and guaranteed convergence to target points.

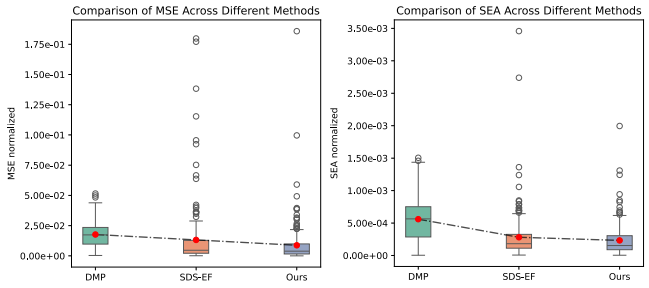

Fig. 6. Quantitative comparison on LASA dataset. Boxplots show distributions of MSE and SEA across 50 trials. Red markers indicate mean values.

Since DMP are inherently limited to modeling individual demonstrations, we trained separate DMP instances for each demonstration trajectory within the same shape category. Fig. 6 presents quantitative results from 50 independent trials. Our method achieves:

- 50.5% lower MSE and 58.5% lower SEA than DMP, highlighting its ability to generalize across demonstrations
- 33.7% MSE and 17.2% SEA improvements over SDS-EF, despite identical stability guarantees

We hypothesize that these gains largely stem from our ODE-driven diffeomorphic mapping, which relaxes SDS-EF's restrictive architectural constraints while preserving stability. The results validate that neural ODE formulations enhance representational capacity compared to traditional flow-based approaches.

D. Robot Reaching Task

To evaluate the real-world applicability of our method, we conducted validation experiments with a Franka Emika robotic arm performing a reaching task. Specifically, the model was trained using a single kinesthetic demonstration and evaluated across 10 independent trials to verify trajectory reproduction and stability. The validation protocol comprised two critical phases: (1) learning from physical demonstrations and (2) real-time execution with stability analysis under perturbations.

We first recorded a reference trajectory by physically guiding the robot's end-effector to the target position. The collected demonstration data was used to train our proposed dynamical system model. Subsequently, we implemented the learned model on the robot's control system to generate real-time end-effector velocity commands.

Figure 7 illustrates three key experimental scenarios:

- **Demonstration Reproduction**: The first sequence confirms replication of the taught trajectory.
- **Generalized Initialization**: The second sequence demonstrates global convergence from arbitrary start positions.
- **Perturbation Recovery**: The third sequence showcases disturbance rejection during execution, where manual interventions were applied to deviate the trajectory.

Fig. 7. Experimental validation of learned reaching dynamics on a Franka Emika robotic arm. Row 1: Reproduction of the demonstrated trajectory. Row 2: Generalized convergence from a new initial position. Row 3: Perturbation recovery through hand interaction. The experiment demonstrates the system's inherent stability and disturbance rejection capabilities.

In advanced practical deployments, detecting and responding to such physical interventions could be further enriched by integrating ultra-sensitive tactile and proximity sensors [26], [27].

The qualitative results presented in Fig. 7 suggest that our learned dynamical system maintains global asymptotic stability while preserving the demonstrated motion characteristics. Notably, the robot consistently converged to the target position within 0.5 cm accuracy across all trials, even when subjected to significant external disturbances. This robustness stems from the theoretically guaranteed stability properties embedded in our learning framework.

VI. CONCLUSIONS AND FUTURE WORK

This paper presents an ODE-driven framework for learning stable dynamical systems from demonstrations. Unlike existing diffeomorphic approaches that rely on restrictive architectural constraints, our method leverages bidirectional temporal neural ODE integration and variational Jacobian computation to achieve both high expressiveness and stability.

Empirical evaluations demonstrate significant improvements, with 33.7% and 17.2% gains in trajectory reproduction accuracy (MSE and SEA, respectively) over state-of-the-art methods on the LASA benchmark. Real-world experiments on a 7-DOF Franka Emika arm validate sub-centimeter tracking accuracy and robust disturbance rejection under external perturbations. Additionally, our variational Jacobian approach reduces computational latency by 20.8%, enabling real-time control. By reformulating diffeomorphic mappings through continuous ODE flows, our work bridges neural ODEs and robotic motion learning, offering a principled framework for stability-guaranteed motion generation.

This work reframes diffeomorphic DS learning through continuous-time flows, preserving Lyapunov stability while lifting architectural invertibility constraints and reducing Jacobian-computation latency via a variational ODE.

Future Work

While the proposed approach shows improved flexibility and performance, we acknowledge several limitations and potential failure modes, which point to avenues for future work:

Computational Overhead: Integrating a neural ODE (with accompanying Jacobian ODE) is more computationally intensive than a single forward pass of a feed-forward network. Although we mitigated this with efficient solvers and our variational Jacobian method, real-time high-frequency control applications or deployments on limited hardware might still pose a challenge. In future work, one could explore further optimizations such as tailor-made ODE solvers

for this task or model-reduction techniques to simplify v_t without losing accuracy.

Handling Uncertainty: Our current formulation is deterministic, producing a fixed trajectory for a given initial state. In real-world settings, human demonstrations often exhibit multi-modality and variability in execution. Stochastic extensions of dynamical systems (as explored in [13]) could be incorporated to model such uncertainties, enabling the generation of slightly varied trajectories or providing probabilistic bounds on the robot's motions. Investigating an ODE-driven stochastic DS, or augmenting the training objective to account for variability and noise, is a promising area for future work to improve the robustness of learned skills under uncertainty.

REFERENCES

[1] B. D. Argall, S. Chernova, and M. Veloso, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] H. Ravichandar, A. S. Polydoros, and S. Chernova, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 297–330, 2020.

[3] X. Xia, L. Cheng, M. Ma, H. Zhang, L. Han, and H. Li, "Neurologically inspired transparent interaction for wearable exoskeletons," *IEEE/ASME Transactions on Mechatronics*, vol. 30, no. 6, pp. 4814–4824, 2025.

[4] H. Li, L. Cheng, S. Qin, and L. Han, "Voluntary control of the hand assistive exoskeleton based on the sEMG-driven musculoskeletal model," *IEEE Robotics and Automation Letters*, vol. 10, no. 7, pp. 7651–7658, 2025.

[5] L. Han, L. Cheng, M. Ma, F. Yang, and H. Li, "Muscle synergy-guided reinforcement learning for embodied musculoskeletal motion skill learning," *IEEE Transactions on Biomedical Engineering*, in press, 2026, doi: 10.1109/TBME.2026.3651379.

[6] L. Han, L. Cheng, H. Li, Y. Zou, S. Qin, and M. Zhou, "Hierarchical optimization for personalized hand and wrist musculoskeletal modeling and motion estimation," *IEEE Transactions on Biomedical Engineering*, vol. 72, no. 1, pp. 454–465, 2025.

[7] D. Kulić, W. Takano, and T. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains," *The International Journal of Robotics Research*, vol. 27, no. 7, pp. 761–784, 2008.

[8] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "FFJORD: Free-form continuous dynamics for scalable reversible generative models," arXiv preprint arXiv:1810.01367, 2018.

[9] M. A. Rana, A. Li, and D. Fox, "Euclideanizing flows: Diffeomorphic reduction for learning stable dynamical systems," in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, 2020, pp. 630–639.

[10] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 6571–6583.

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

[12] S. Peng, H. Zhang, Z. Liu, W. He, and L. Cheng, "iProDMP: An enhanced probabilistic dynamic movement primitives framework for hip exoskeleton-assisted lifting," *IEEE Transactions on Automation Science and Engineering*, vol. 23, pp. 4892–4903, 2026.

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

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

[15] K. Neumann and J. J. Steil, "Learning robot motions with stable dynamical systems under diffeomorphic transformations," *Robotics and Autonomous Systems*, vol. 70, pp. 1–15, 2015.

[16] D. X. Huang, X. H. Zhou, X. L. Xie, S. Q. Liu, Z. Q. Feng, Z. G. Hou, N. Ma, and L. Yan, "Real-time 2D/3D registration via CNN regression and centroid alignment," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 85–98, 2024.

[17] D. X. Huang, X. H. Zhou, M. J. Gui, X. L. Xie, S. Q. Liu, S. Y. Wang, T. Y. Xiang, R. Z. Ma, N. F. Xiao, and Z. G. Hou, "VasoMIM: Vascular anatomy-aware masked image modeling for vessel segmentation," in *Proceedings of the Annual AAAI Conference on Artificial Intelligence (AAAI)*, 2026.

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

[19] H. Y. Zhang, L. Cheng, and Y. Zhang, "Learning robust point-to-point motions adversarially: A stochastic differential equation approach," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2357–2364, 2023.

[20] H. Y. Zhang, L. Cheng, Y. Zhang, and Y. Wang, "Neural Liénard system: Learning periodic manipulation skills through dynamical systems," *Science China Information Sciences*, vol. 67, no. 12, pp. 1–16, 2024.

[21] H. Y. Zhang, L. Cheng, Z. Y. Liu, and Y. Zhang, "Learning orbitally stable dynamics via transverse contraction criteria for modeling periodic tasks," *The International Journal of Robotics Research*, 2025. doi: 10.1177/02783649251369026.

[22] H. Y. Zhang, Y. Zhang, Y. X. Zou, H. C. Li, and L. Cheng, "Neural-Lyapunov Fusion: Stable dynamical system learning for robotic motion generation," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025, pp. 11826–11831.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[24] [Online]. Available: https://bitbucket.org/khansari/lasahandwritingdataset/

[25] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[26] X. Meng, L. Cheng, Z. Li, "A tactile-proximity dual-mode photoelectric sensor: Implementation and applications," *IEEE Transactions on Robotics*, vol. 41, pp. 6607–6624, 2025.

[27] Z. Li, L. Cheng, Z. Liu, J. Wei, and Y. Wang, "FOCERS: An ultra-sensitive and robust soft optical 3D tactile sensor," *Soft Robotics*, vol. 12, no. 4, pp. 445–454, 2025.