Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Diffeomorphic Obstacle Avoidance for Contractive Dynamical Systems via Implicit Representations

Ken-Joel Simmoteit
Karlsruhe Institute of Technology
Karlsruhe, Germany
Ken-Joel.Simmoteit@student.kit.edu

Philipp Schillinger
Bosch Center for Artificial Intelligence
Renningen, Germany
Philipp.Schillinger@de.bosch.com

Leonel Rozo
Bosch Center for Artificial Intelligence
Renningen, Germany
Leonel.Rozo@de.bosch.com

***Abstract*—Ensuring safety and robustness of robot skills is becoming crucial as robots are required to perform increasingly complex and dynamic tasks. The former is essential when performing tasks in cluttered environments, while the latter is relevant to overcome unseen task situations. This paper addresses the challenge of ensuring both safety and robustness in dynamic robot skills learned from demonstrations. Specifically, we build on neural contractive dynamical systems to provide robust extrapolation of the learned skills, while designing a full-body obstacle avoidance strategy that preserves contraction stability via diffeomorphic transforms. This is particularly crucial in complex environments where implicit scene representations, such as Signed Distance Fields (SDFs), are necessary. To this end, our framework called Signed Distance Field Diffeomorphic Transform, leverages SDFs and flow-based diffeomorphisms to achieve contraction-preserving obstacle avoidance. We thoroughly evaluate our framework on synthetic datasets and several real-world robotic tasks in a kitchen environment. Our results show that our approach locally adapts the learned contractive vector field while staying close to the learned dynamics and without introducing highly-curved motion paths, thus outperforming several state-of-the-art methods.**

I. Introduction

Teaching robots skills has been a key challenge in robotics research for over four decades [40]. The dominant paradigm involves learning robotic skills through expert examples, commonly referred to as Learning from Demonstration (LfD) [3, 8, 53, 60]. By enabling robots to execute complex motion skills, LfD has led to numerous promising methods [7, 17, 21, 51] with applications in flexible manufacturing [16, 59], household environments [51, 69], human-robot collaboration [31, 57, 58], and robot-assisted minimally invasive surgery [64], among many others.

For safe operation in human-centric and dynamic settings, learned skills must be stable and reliable, avoiding unexpected movements under unseen situations such as different task conditions or external perturbations. This highlights the importance of *stable LfD skills* [53]. Early efforts to ensure stability in LfD focused on asymptotic stability criteria, often leveraging Lyapunov stability [12, 25, 51, 68]. While Lyapunov stability primarily ensures asymptotic point-wise stability, many robot skills require following complex task-relevant trajectories. To address this limitation, researchers have conceptualized a more general stability criterion, namely the *contraction property* of dynamical systems, which guarantees exponential convergence to a trajectory [38, 67]. This concept has been recently leveraged in LfD frameworks, where a robot skill, represented by a first-order dynamical system, is endowed with a contractive behavior [4, 9, 54, 62].

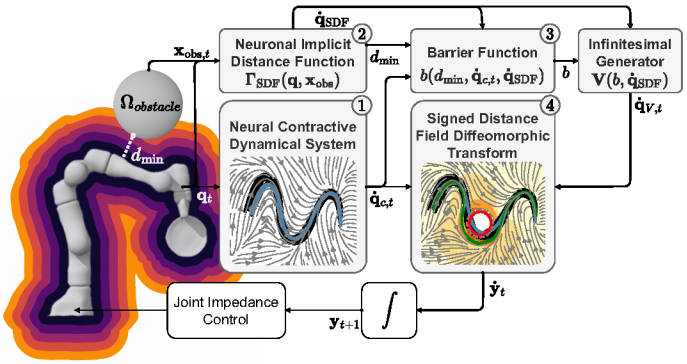


Figure 1: Overview of the proposed **Signed Distance Field Diffeomorphic Transform**: *(1)* A neural contractive dynamical system; *(2)* A learned implicit distance function; *(3)* A barrier function, and *(4)* a contraction-preserving diffeomorphic transform.

In addition to stability, ensuring safety is paramount. This often demands integrating *obstacle avoidance* into robot skills while still providing stability guarantees. In the context of contractive systems, Huber et al. [18] introduced the Modulation Matrix (MM) method to preserve contraction stability during obstacle avoidance. Subsequently, Beik-Mohammadi et al. [4] applied this approach to neural contractive dynamical systems (NCDS), enabling stable collision-free motion in the robot's task space. Despite these advances, articulated robotic arms require whole-body obstacle avoidance. While traditional methods like artificial potential fields [27] and recent approaches like Diffeomorphic Transforms (DT) [71], tackle this problem via robust joint-space obstacle avoidance solutions that account for asymptotic stability, they do not consider contraction stability guarantees. Therefore, preserving contraction during whole-body obstacle avoidance remains largely unexplored, posing a key challenge for robot motion control, which we address in this paper.

When considering cluttered and dynamic environments, such as household settings, conventional obstacle representations using simple geometric primitives are insufficient [4, 19],

as they rely on coarse approximations and may not be easily integrated with neural architectures. Instead, a differentiable *implicit distance representation* is more advantageous as it facilitates a fine-grained, real-time, and resource-efficient solution, while being seamlessly integrated with other neural network architectures. This has led to the rise of Signed Distance Fields (SDF)s [15, 22, 45, 47], which have recently been leveraged to learn articulated robot surfaces [11, 30, 36]. In these approaches, the SDF is a function of the robot's joint state, capturing the continuous robot geometry over its entire configuration space. Still, integrating implicit representations with contraction-preserving obstacle avoidance remains an open problem. To bride this gap, we propose to combine the robustness of contraction stability with the precision of implicit representations for efficient and safe whole-body obstacle avoidance.

Specifically, **this paper** introduces a novel framework, the **Signed Distance Field Diffeomorphic Transform (SDT)**, for contraction-preserving obstacle avoidance based on an implicit scene representation. The key contributions of this work are:

1) **Contraction-Preserving Implicit Obstacle Avoidance**: We introduce a new method for achieving contraction-preserving obstacle avoidance in contractive dynamical systems by leveraging DT, SDF techniques, and barrier functions. First, we define an infinitesimal generator derived from the robot's implicit representation and regulate it using barrier functions. Second, using the resulting vector field, we construct a flow that reshapes the contractive dynamics, either through a differential coordinate change, or by applying a standard coordinate change via a pullback.
2) **Obstacle Avoidance Metrics**: We introduce a set of quantitative metrics to assess obstacle avoidance performance, focusing on flow curvature and vector field misalignment of the modulated contractive motion.
3) **Experiments**: We extensively evaluated our framework on the 2D LASA dataset [25] and on two real-world tasks in a kitchen environment: *(1)* EMPTYING A DISHWASHER; and *(2)* OPENING A DISHWASHER. We also provide a comprehensive comparison among various SDF methods and obstacle avoidance approaches including Modulation Matrix (MM), DT and artificial potential functions.

II. RELATED WORK

A. Learning Contractive Dynamical Systems

The use of contractive guarantees in learned dynamical systems is a growing trend in robot motion skill learning. In general, contraction guarantees can be introduced into the skill model via three methods: *(i)* As a regularizer or constraint on the main skill learning objective [54, 55]; *(ii)* As a contraction metric designed to stabilize a non-contractive skill [65, 66]; and *(iii)* By learning a skill model that is contractive by design [1, 4, 23, 5]. The latter strategy is the one we follow in our framework as it provides stronger theoretical and practical guarantees. We review this set of works in more detail next.

Beik-Mohammadi et al. [4] introduced Neural Contractive Dynamical System (NCDS), which inherently embeds the contraction property within a Neural Network (NN), thus avoiding regularizers or a separate model for learning a contraction metric. NCDS was later extended with regularizers controlling the system's contraction rate, in addition to conditioning on task variables and providing contraction-preserving obstacle avoidance via pullback Riemannian metrics [5]. Similarly, Jaffe et al. [23] presented an inherently contractive model, which leverages diffeomorphisms between the data space and a latent space to provide global contraction guarantees. Another approach is followed by Abyaneh et al. [1], who propose learning an inherent contractive dynamical system via recurrent equilibrium networks and coupling layers. We build upon previous work to learn a contractive dynamical system via NCDS. However, our approach can be seamlessly integrated with any contractive dynamical system. Thereby, we address the contraction-preserving obstacle avoidance problem, which has been largely overlooked in previous works except in [4, 18].

B. Stability-Preserving Obstacle Avoidance

Applying reactive obstacle avoidance in LfD settings requires careful consideration to ensure safe and robust robot behavior, particularly when motion skills are encoded as stable dynamical systems. To address this, Khansari-Zadeh and Billard [26] enhanced the Harmonic Potential Function [28], which uses potential flows from fluid mechanics for obstacle avoidance, to ensure that asymptotically-stable dynamical systems maintain stability via an MM. Later, Huber et al. [18] demonstrated that this approach can also preserve contraction guarantees, even for concave obstacles [19, 13]. While Huber et al. [19] primarily relied on simple geometric primitives for obstacle avoidance, Fourie et al. [13] employed learned obstacle representations of the entire robot configuration space combined with MM. Their work, conducted in parallel to our research, preserves asymptotic stability but did not address contraction stability guarantees.

Building on prior work in learning stable dynamical systems via diffeomorphisms [44, 51, 68, 70], Zhi et al. [71] introduced the DT method, which defines an obstacle-avoidance diffeomorphism via a flow field. This diffeomorphism transforms the learned dynamics while providing asymptotically-stable obstacle avoidance. Importantly, the DT method also accounts for joint-space stable skills using the Moore-Penrose inverse and the robot's forward kinematics. This work, however, shared a limitation with some of the foregoing approaches: A focus on asymptotic stability rather than the more general contraction property. In summary, whole-body obstacle avoidance methods that preserve contraction, especially in complex scenes, presents a significant challenge that, to our knowledge, remains unresolved.

C. Implicit Robot Representations

Capturing the robot's spatial structure is essential for enabling effective obstacle avoidance, preventing self-collisions, and facilitating physical interaction. In this context, implicit representations provide a powerful way to model a robot's shape and volume. For example, SDFs provide an implicit, memory-efficient, and computationally fast surface representation. Studies by Li et al. [36] and Chen et al. [11] demonstrate that SDFs offer significant advantages over geometric primitives by providing a higher level of detail, facilitating precise interaction.

The use of implicit representations for articulated kinematic chains is particularly challenging as the configuration of the robot's structure in the workspace varies as a function of the robot joint position. In this case, each robot link can be modeled by separate SDFs using architectures such as NNs [30] and Bernstein polynomial basis functions (BP) [36], or the entire robot body can be represented using a single model [37], all of which yielding smooth surfaces. Notably, Li et al. [35] observe that learning the Euclidean distance between the robot surface and a workspace query point is highly nonlinear in configuration space. To mitigate this, they propose computing the distance in configuration space, leading to the Configuration Space Distance Fields (CDF) method. While Li et al. [35] and Marticorena et al. [43] made obstacle avoidance for point-to-point motion possible using SDFs and a QP solver, no existing approaches, to the best of our knowledge, perform contraction-preserving obstacle avoidance with SDFs.

III. Background

A. Contraction Stability

We are interested in learning contractive robot motion skills, as contraction provides a rigorous and general stability certificate for robust generalization. Intuitively, contraction stability studies the evolution of two trajectories of a dynamical system given different initial conditions [38]. In other words, contraction implies that a dynamical system "forgets" its initial conditions rapidly, with the distance between any two trajectories decreasing exponentially over time. Formally, consider two trajectories τ_i and τ_j generated by a dynamical system of the form $\dot{\mathbf{x}}_t = f(\mathbf{x}_t)$, where $\mathbf{x}_t \in \mathbb{R}^D$ is the system state, $f\colon \mathbb{R}^D \to \mathbb{R}^D$, and $\dot{\mathbf{x}}_t = {}^{\mathrm{d}\mathbf{x}}/_{\mathrm{d}t}$. Using their virtual displacement $\delta\mathbf{x} = \tau_j - \tau_i$, the squared distance between the trajectories corresponds to $\delta\mathbf{x}^\top\delta\mathbf{x}$ [38]. Contraction theory cares about the evolution of this distance, i.e., its rate of change,

$$\frac{\mathrm{d}}{\mathrm{d}t}(\delta\mathbf{x}^\top\delta\mathbf{x}) = 2\delta\mathbf{x}^\top\delta\dot{\mathbf{x}} = 2\delta\mathbf{x}^\top\mathbf{J}_f(\mathbf{x})\delta\mathbf{x}, \tag{1}$$

where $\delta\dot{\mathbf{x}} = \mathbf{J}_f(\mathbf{x})\delta\mathbf{x}$, with $\mathbf{J}_f(\mathbf{x}) = {}^{\partial f}/_{\partial\mathbf{x}}$. Lohmiller and Slotine [38] show that a dynamical system exhibits contraction if the largest eigenvalue $\lambda_{\max}$ of the symmetric part of its Jacobian is uniformly and strictly negative. Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}t}(\delta\mathbf{x}^\top\delta\mathbf{x}) \leq 2\lambda_{\max}\delta\mathbf{x}^\top\delta\mathbf{x} \implies \|\delta\mathbf{x}\| \leq \|\delta\mathbf{x}_0\|e^{-\alpha t}, \tag{2}$$

which means that the virtual displacement decreases exponentially to zero with contraction rate α [38].

Definition 1 (Contraction stability [38])**.** A dynamical system $\dot{\mathbf{x}}_t = f(\mathbf{x}_t)$ is contractive if its Jacobian $\mathbf{J}_f = \frac{\partial f}{\partial\mathbf{x}}$ is uniformly negative definite for all $\mathbf{x} \in \mathbb{R}^n$ and at all times $t \geq 0$. This means that,

$$\exists\alpha > 0,\ \forall\mathbf{x},\ \forall t \geq 0,\ \frac{1}{2}\left(\frac{\partial f}{\partial\mathbf{x}} + \left(\frac{\partial f}{\partial\mathbf{x}}\right)^\top\right) \preceq -\alpha\mathbf{I} \prec \mathbf{0}. \tag{3}$$

Definition 1 can be generalized to account for a more general distance definition, namely $\delta\mathbf{x}^\top\mathbf{G}(\mathbf{x})\delta\mathbf{x}$, where $\mathbf{G}(\mathbf{x})$ is a Riemannian metric [32, 61]. This also allows us to analyze the contractive behavior of a dynamical system under a coordinate change via a smooth diffeomorphism $\mathbf{y} = \psi(\mathbf{x})$. The corresponding differential change in coordinates is given by $\delta_\mathbf{y} = \mathbf{J}_\psi\delta_\mathbf{x}$, where $\mathbf{J}_\psi$ is the Jacobian of the diffeomorphism ψ. Manchester and Slotine [41] demonstrated that contraction is invariant under such coordinate changes.

Theorem 1 (Invariance under coordinate change [41])**.** If Definition 1 is satisfied for a dynamical system, then it is preserved under the following transformations:

1) Affine feedback transformations $u(\mathbf{x}, \mathbf{v}) = \mathbf{a}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{v}$, with $\mathbf{B}(\mathbf{x})$ being a smooth nonsingular $n \times n$ matrix function;
2) Differential coordinate changes $\delta_\mathbf{y} = \mathbf{J}_\psi(\mathbf{x})\delta_\mathbf{x}$, where $\mathbf{J}_\psi(\mathbf{x})$ is a nonsingular matrix for all $\mathbf{x}$, and induces a new contraction metric,
$$\mathbf{G}_\mathbf{y}(\mathbf{x}) := \mathbf{J}_\psi(\mathbf{x})^\top\mathbf{G}(\mathbf{x})\mathbf{J}_\psi(\mathbf{x}); \tag{4}$$
3) Coordinate changes $\mathbf{y} = \psi(\mathbf{x})$, where ψ is a diffeomorphism, with corresponding contraction metric $\mathbf{G}_\mathbf{y}$.

Given a contraction metric as stated in Theorem 1, we can also determine whether a given system is contractive by following the results in [67].

Theorem 2 (Contraction conditions [67])**.** Assume that a uniformly positive definite matrix $\mathbf{G}(\mathbf{x}, t) = \mathbf{J}_\psi(\mathbf{x})^\top\mathbf{J}_\psi(\mathbf{x}) \succ \mathbf{0}$ exists, $\forall\mathbf{x}, t$, where $\mathbf{J}_\psi(\mathbf{x})$ defines a smooth coordinate transformation of the differential $\delta\mathbf{x}$, i.e., $\delta\mathbf{y} = \mathbf{J}_\psi(\mathbf{x})\delta\mathbf{x}$. The following equivalent condition holds for $\alpha \in \mathbb{R}_{>0}$ and $\forall\mathbf{x}, t$,

$$\dot{\mathbf{G}} + \mathbf{G}\frac{\partial f}{\partial\mathbf{x}} + \frac{\partial f}{\partial\mathbf{x}}^\top\mathbf{G} \leq -2\alpha\mathbf{G}, \tag{5}$$

then all solution trajectories of the system converge exponentially fast to a single trajectory, irrespective of their initial conditions, with an exponential convergence (contraction) rate α.

B. Flow-based Diffeomorphisms

As contraction is preserved under a change of coordinates, we leverage this to design obstacle-avoidance behaviors based

on diffeomorphic mappings, which we revise next. A diffeomorphism $\psi\colon \mathcal{Y} \to \mathcal{X}$ is a smooth, bijective map with a smooth inverse, thereby providing a coordinate transformation between two differentiable manifolds $\mathcal{Y}$ and $\mathcal{X}$. According to Lee [33, Theorem 9.12], any such diffeomorphism can be realized as a flow generated by an infinitesimal generator $\mathbf{V}$, often represented as a vector field on a smooth manifold. Specifically, let $\mathbf{V}\colon \mathbb{R}^d \to \mathbb{R}^d$ be a time-independent vector field and the flow $\gamma\colon \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ be defined by,

$$\gamma(t, \mathbf{y}) = \mathbf{y} + \int_0^t \mathbf{V}(\gamma(u, \mathbf{y}))\, du = \mathbf{x}. \tag{6}$$

This flow $\gamma(t, \mathbf{y})$ provides the position $\mathbf{x}$ at time t of a trajectory starting at $\mathbf{y}$ when $t = 0$. For each fixed time t, this flow defines a diffeomorphism $\psi\colon \mathcal{Y} \to \mathcal{X}$ by $\psi(\mathbf{y}) := \gamma(t, \mathbf{y})$. Therefore, the flow defines an invertible mapping, whose inverse can be computed by reversing the direction of time,

$$\gamma(-t, \mathbf{x}) = \mathbf{x} + \int_{-t}^0 \mathbf{V}(\gamma(u, \mathbf{x}))\, du = \mathbf{y}. \tag{7}$$

Note that this flow-based diffeomorphism $\psi(\mathbf{y}) := \gamma(t, \mathbf{y})$ maps the initial point $\mathbf{y} \in \mathcal{Y}$ to the point $\mathbf{x} = \gamma(t, \mathbf{y}) \in \mathcal{X}$. Furthermore, given a vector field $f\colon \mathcal{X} \to \mathcal{T}\mathcal{X}$, where $f(\mathbf{x})$ assigns a tangent vector in $\mathcal{T}_{\mathbf{x}}\mathcal{X}$ to each point $\mathbf{x} \in \mathcal{X}$, we can use ψ to pullback f to a vector field on $\mathcal{Y}$. Specifically, let $\mathbf{J}_\psi$ be the Jacobian of ψ, then the pullback of f via ψ is,

$$\dot{\mathbf{y}} = \mathbf{J}_\psi^{-1} f(\psi(\mathbf{y})), \tag{8}$$

thereby transforming tangent vectors on $\mathcal{X}$ to corresponding tangent vectors on $\mathcal{Y}$ [33].

C. Signed Distance Fields (SDFs)

Representing objects as meshes or point clouds is often inefficient due to high memory requirements and redundant information [47]. Moreover, these discrete representations lack smoothness and differentiability as they approximate surfaces with sampled points or polygons. To overcome these limitations, SDFs offer a compact, smooth, and differentiable representation. An SDF defined by $\Gamma_{\text{SDF}}\colon \mathbb{R}^d \to \mathbb{R}$, encodes the objects surface through the minimum distance d_{min} from a point $\mathbf{x} \in \mathbb{R}^d$ to the surface using a smooth function [47]. Three regions can be identified: Free space $\mathcal{H}^f = \{\mathbf{x} \in \mathbb{R}^d \mid \Gamma_{\text{SDF}}(\mathbf{x}) > 0\}$, surface boundary $\mathcal{H}^s = \{\mathbf{x} \in \mathbb{R}^d \mid \Gamma_{\text{SDF}}(\mathbf{x}) = 0\}$, and interior $\mathcal{H}^i = \{\mathbf{x} \in \mathbb{R}^d \mid \Gamma_{\text{SDF}}(\mathbf{x}) < 0\}$ (see Fig. 2 for an illustration).

Note that this approach decouples spatial resolution from memory usage, enabling high-resolution reconstructions with fast inference [42]. Park et al. [47] propose using a Multilayer Perceptron (MLP) to model the SDF, offering a lightweight and efficient model. Additionally, piecewise polynomial representations enable continuous surface modeling [49]. Marić et al. [42] extend this by employing BP basis functions which enables smooth surfaces that can be incrementally refined.

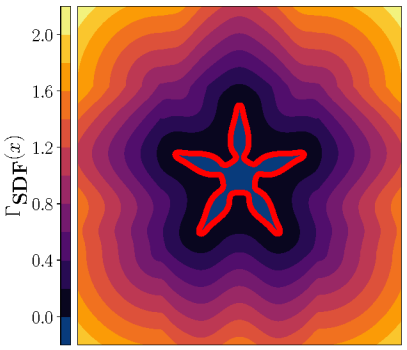


Figure 2: Illustration of a 2D SDF: Contour lines of an star-shaped SDF with the distance Γ_{SDF} to the obstacle surface, depicted as a solid red line.

IV. Contraction Preserving Obstacle Avoidance

Here we introduce our contraction-preserving obstacle avoidance method. First, we explain the contractive dynamical system used in this paper. Later, we describe the contraction-preserving obstacle avoidance method.

A. Learned Contractive Dynamical System

Consider a contractive dynamical system $\dot{\mathbf{x}} = f_{\text{NCDS}}(\mathbf{x})$, where $\dot{\mathbf{x}}$ represents the first-order time derivative of the system state $\mathbf{x}$. From Theorem 2, we know that if the symmetric part of the system dynamics Jacobian $\mathbf{J}_{f_{\text{NCDS}}}$ is negative definite, our dynamics is contractive. Beik-Mohammadi et al. [4] propose directly learning this Jacobian to intrinsically embed the contraction property into f_{NCDS}. Specifically, the Jacobian is modeled with parameters $\boldsymbol{\theta}$ as,

$$\hat{\mathbf{J}}_{f_{\text{NCDS}}}(\mathbf{x}) = -(\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x})^\top \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}) + \boldsymbol{\epsilon}), \tag{9}$$

where $\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x})\colon \mathbb{R}^D \to \mathbb{R}^D$ represents the square root of the Jacobian, thus ensuring symmetry. The small positive vector $\boldsymbol{\epsilon} > \mathbf{0}$ guarantees that the eigenvalues are bounded by $-\boldsymbol{\epsilon}$ [4]. Given the negative-definite Jacobian (9), the contractive dynamics is obtained via the calculus for line integrals [39],

$$\begin{aligned} \dot{\mathbf{x}} &= f_{\text{NCDS}}(\mathbf{x}) \\ &= \dot{\mathbf{x}}_0 + \int_0^1 \hat{\mathbf{J}}_{f_{\text{NCDS}}}\big(c(\mathbf{x}, t, \mathbf{x}_0)\big)\dot{c}(\mathbf{x}, t, \mathbf{x}_0)\mathrm{d}t, \end{aligned} \tag{10}$$

where,

$$c(\mathbf{x}, t, \mathbf{x}_0) = (1 - t)\mathbf{x}_0 + t\mathbf{x}, \quad \dot{c}(\mathbf{x}, t, \mathbf{x}_0) = \mathbf{x} - \mathbf{x}_0, \tag{11}$$

and $\mathbf{x}_0$ is the initial system state [4]. The velocity $\dot{\mathbf{x}}_t$ is computed using a numerical integral solver given $\mathbf{x}_t$ [4, 10].

When learning robot skills via NCDS, we have two options: Learning the NCDS in joint space $\mathcal{C}$ or instead, encoding the skill in task space $\mathcal{X}$. In the latter case, NCDS uses Lie groups to represent the end-effector orientation in $\mathrm{SO}(3)$ [4], and employs its Lie algebra $\mathfrak{so}(3)$ to obtain skew-symmetric matrices $\hat{\mathbf{w}} \in \mathfrak{so}(3)$ [20]. The mapping between $\mathrm{SO}(3)$ and $\mathfrak{so}(3)$ is defined via the exponential and the logarithmic map. To train a NCDS model, the loss function minimizes the average reconstruction error between the true next state $\mathbf{x}_{t+1}$ and the predicted next state $\hat{\mathbf{x}}_{t+1} = \mathbf{x}_t + \hat{\dot{\mathbf{x}}}_{t+1}$ [4],

$$\mathcal{L}_{\text{Jac}} = \frac{1}{T_n} \sum_{t=0}^{T_n - 1} \|\mathbf{x}_{t+1} - (\mathbf{x}_t + \hat{\dot{\mathbf{x}}}_{t+1})\|^2. \tag{12}$$

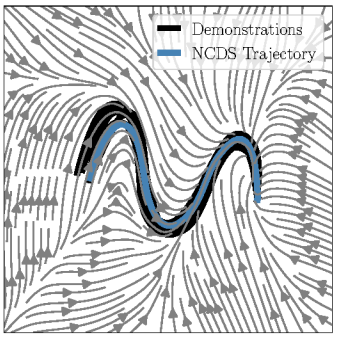


Figure 3: Exemplary NCDS model of a skill learned in 2D space, with demonstrated trajectories from the LASA dataset in black and the integrated motion over the NCDS model in blue. The gray arrows represent the NCDS vector field.

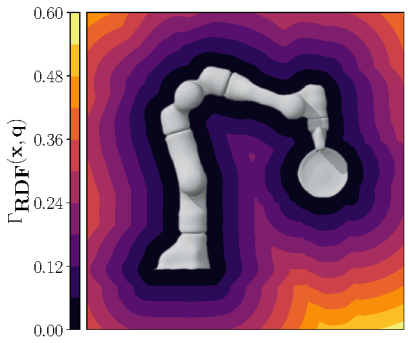


Figure 4: Illustration of a learned RDF showing the contour lines of the robot's SDF for a Franka-Emika Panda robot grasping a plate. The contours indicate the distance Γ_{RDF} to the robot's surface. The inner gray mesh shows the learned robot surface $\mathcal{R}$.

Algorithm 1: NCDS Training in Joint Space

Input : Demonstrations: $\tau = \{\mathbf{q}_{n,t}\}_{n=1,t=0}^{N,T_n-1}$, where $n \in [1, N]$, $t \in [1, T_n]$, and initial parameters $\boldsymbol{\theta}$, initial regularizer $\boldsymbol{\epsilon}$

Output: Learned Jacobian network parameters $\boldsymbol{\theta}$, learned regularizer $\boldsymbol{\epsilon}$

```
while not converged do
    for each demonstration n ∈ N do
        for each time step t ∈ T_n do
            Calculate Ĵ_{f_NCDS} of the NCDS with Eq. (9)
            Define the contractive dynamics f_NCDS with Eq. (10)
    Train the Jacobian network by solving:
    θ*, ε* = argmin_{θ,ε} (L_Jac + β_ε L_ε + β_noise L_noise)
```

In addition, a state-independent regularizer is added to optimize the eigenvalue bound $\boldsymbol{\epsilon} = [\epsilon_1, \cdots, \epsilon_D]$ as follows [5],

$$\mathcal{L}_\epsilon = -\sum_{n=2}^{D} |\epsilon_1 - \epsilon_n|^2. \tag{13}$$

To enhance the local contraction properties of NCDS, we propose an additional noise-injected reconstruction loss,

$$\mathcal{L}_{\text{noise}} = -\frac{1}{T_n} \sum_{t=0}^{T_n-1} \left\| \mathbf{x}_{t+1} - \left(\tilde{\mathbf{x}}_t + \hat{\dot{\mathbf{x}}}_t \right) \right\|^2, \tag{14}$$

which acts orthogonally to the direction of motion $\dot{\mathbf{x}}$, with $\tilde{\mathbf{x}}_t \sim \mathcal{N}(\mathbf{x}_t, \boldsymbol{\Sigma}_{\perp \dot{\mathbf{x}}_t})$, representing noise injected in the directions perpendicular to the movement. Our regularizer encourages NCDS to be robust against orthogonal perturbations, thereby implicitly improving its local contractive behavior. The total loss is $\mathcal{L} = \mathcal{L}_{\text{Jac}} + \beta_\epsilon \mathcal{L}_\epsilon + \beta_{\text{noise}} \mathcal{L}_{\text{noise}}$, with β_ϵ and β_{noise} being weights balancing the losses influence. NCDS training is summarized in Algorithm 1. Additionally, Fig. 3 illustrates a vector field learned by NCDS.

B. SDF-based Diffeomorphic Transform

Our obstacle avoidance method consists of two components: An intrinsic representation of the robot surface and a contraction-preserving transformation of a contractive vector field f_c. The latter can be: *(1)* A learned joint-space NCDS $f_{\text{NCDS}}(\mathbf{q})$; *(2)* A learned task-space NCDS $f_{\text{NCDS}}(\mathbf{x})$ mapped to joint space; and *(3)* Any differentiable contractive dynamical system in joint space. Next, we describe the implicit representation of the robot surface, which provides the basis for deriving the infinitesimal generator for obstacle avoidance.

1) Implicit Robot Representations: To achieve obstacle avoidance, one must know the minimal distance $d_{\min}$ between the scene surface $\mathcal{S}$ and the robot surface $\mathcal{R}$. Therefore we start by learning an implicit representation of $d_{\min}$. There are two ways to achieve this: *(1)* A learned SDF of the robot and; *(2)* A learned SDF of the scene. To learn a SDF for the scene, we refer the reader to the established approaches in [15, 45, 47]. On the other hand, the distance w.r.t the robot surface can be defined by $d_{\min} = \Gamma_{\text{RDF}}(\mathbf{x}, \mathbf{q})$, given a query point $\mathbf{x}$ and the robot state $\mathbf{q} \in \mathcal{C}$ [11, 30, 36]. This implicit robot representation is called Robot Signed Distance Fields (RDF). We follow the approach from Li et al. [36] and define the RDF as,

$$\Gamma_{\text{RDF}} = \min(\Gamma_{\Omega_1^r}, \Gamma_{\Omega_2^r}, ..., \Gamma_{\Omega_K^r}), \tag{15}$$

with $\Gamma_{\Omega_k^r}, k = 1, ..., K$, and Γ_{RDF} defining the SDF of K links in the robot base frame r. Thereby, each link is transformed via the robot's kinematic chain ${}^r\mathbf{T}_k(\mathbf{q}) \in \mathbb{SE}(3)$ from the frame of the k-th link to the base frame, resulting in $\Gamma_{\Omega_k^r}(\mathbf{x}, \mathbf{q}) = \Gamma_{\Omega_k^r}\left({}^r\mathbf{T}_k(\mathbf{q})\mathbf{x}\right)$ [36]. Given the SDF Γ_{Ω_o} (e.g., a grasped object) and its corresponding transformation ${}^r\mathbf{T}_o(\mathbf{q}) \in \mathbb{SE}(3)$ to the robot base frame, we can easily extend the RDF by adding the transformed $\Gamma_{\Omega_o^r}$ to the $\min$ operation in Equation (15). We choose BP to learn the SDF $\Gamma_{\Omega_k^r}$ of each link k as it provides a particularly smooth surface. However, any other SDF method can be used. An exemplary learned RDF with a grasped plate is shown in Fig. 4.

Alternatively, Li et al. [35] propose learning joint-level distances rather than Euclidean point-to-surface distances, leading to the CDF with corresponding implicit distance function Γ_{CDF}. The CDF measures the distance in radians, corresponding to the movement of joint angles towards the zero-level configuration set $\mathbf{q}^*$, resulting in [35],

$$\Gamma_{\text{CDF}}(\mathbf{x}, \mathbf{q}) = \min_{\mathbf{q}^*}(\mathbf{q} - \mathbf{q}^*). \tag{16}$$

CDF solves the inverse kinematics problem in a single step by updating the current robot joint configuration along the CDF

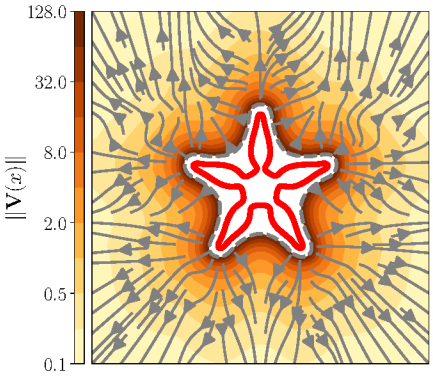


Figure 5: Vector field of a learned star-shaped SDF function with inverse barrier function $b_{\text{inv}}(\mathbf{x}, \mathbf{q})$. The contour lines represent the barrier's magnitude. The solid red line shows the surface of the object and the dashed grey line the safety threshold t_{save}.

gradient toward the zero-level configuration set $\mathbf{q}^*$ via,

$$\mathbf{q}^* = \mathbf{q} - \Gamma_{\text{CDF}}(\mathbf{x}, \mathbf{q}) \nabla_{\mathbf{q}} \Gamma_{\text{CDF}}(\mathbf{x}, \mathbf{q}). \tag{17}$$

However, a limitation of this approach is that extending the CDF with an object's SDF, such as that of a grasped object, is not as straightforward as in the RDF framework. Accomplishing this would require training a new model or incorporating additional conditioning into the existing model. In this work, both RDF and CDF are employed and compared. In the following, we use the term $\Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q})$ to denote a general implicit distance function between the scene and the robot. When referring specifically to the robot's implicit distance function, we denote it by $\Gamma_{\mathcal{R}}$. Analogously, $\Gamma_{\mathcal{S}}$ is used for the implicit distance function of the scene.

2) Implicit Infinitesimal Generator: For obstacle avoidance, the gradient $\nabla_{\mathbf{q}} \Gamma_{\text{SDF}}$ determines how the robot joints should move away from obstacles. Thus, this gradient acts as an *infinitesimal generator* $\mathbf{V}$. To ensure strict collision avoidance, we use an inverse barrier function $b_{\text{inv}}(\mathbf{x}, \mathbf{q})$, similarly to [29]. This barrier is defined as follows,

$$b_{\text{inv}}(\mathbf{x}, \mathbf{q}) = \frac{1}{\Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q})}, \tag{18}$$

which is combined with the gradient, leading to the infinitesimal generator $\mathbf{V} = b_{\text{inv}} \nabla_{\mathbf{q}} \Gamma_{\text{SDF}}$. As $\Gamma_{\text{SDF}} \to 0$, then $b_{\text{inv}} \to \infty$, ensuring collision avoidance. When $\Gamma_{\text{SDF}} \to \infty$, then $b_{\text{inv}} \to 0$, canceling the effect of the obstacle avoidance gradient $\nabla_{\mathbf{q}} \Gamma_{\text{SDF}}$. The introduction of this barrier function is motivated by the fact that SDFs are often learned via the Eikonal loss, enforcing $\|\nabla_{\mathbf{x}} \Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q})\| = 1$ [47]. Thus, gradients remain non-negligible even far away from the obstacles. Figure 5 illustrates the resulting vector field for a geometric primitive following the aforementioned approach.

We can make the barrier function more general by adding tunable parameters like a safety threshold t_{save} and a scaling factor s_{grad} as follows,

$$b_{\text{inv}}(\mathbf{x}, \mathbf{q}) = \frac{s_{\text{grad}}}{\Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q}) - t_{\text{save}}}. \tag{19}$$

Moreover, we also propose to leverage *swept features*, aimed at weighting movements towards and tangential to the obstacle while ignoring those away from it, using the dot product $\dot{\mathbf{q}}_{\text{NCDS}} \cdot \nabla_{\mathbf{q}} \Gamma_{\text{SDF}}$. To do so, we need to distinguish whether the SDF of the scene $\Gamma_{\mathcal{S}}$ or of the robot $\Gamma_{\mathcal{R}}$ is used, since the corresponding velocity fields and gradient direction may differ. Specifically, if $\Gamma_{\mathcal{R}}$ is employed, then the gradient of $\Gamma_{\mathcal{R}}$ and f_c point in the same direction when moving towards the obstacle and point in opposite directions when moving away from it. If $\Gamma_{\mathcal{S}}$ is used, this phenomenon is the opposite. For the implicit distance function of the robot and the scene, swept-augmented barriers are defined as,

$$b_{\text{swept},\mathcal{R}}(\mathbf{x}, \mathbf{q}) = \frac{1}{2}\left(1 + \frac{\dot{\mathbf{q}}_{\text{NCDS}} \cdot \nabla_{\mathbf{q}} \Gamma_{\mathcal{R}}(\mathbf{x}, \mathbf{q})}{|\dot{\mathbf{q}}_{\text{NCDS}}| \; |\nabla_{\mathbf{q}} \Gamma_{\mathcal{R}}(\mathbf{x}, \mathbf{q})|}\right), \tag{20}$$

$$b_{\text{swept},\mathcal{S}}(\mathbf{x}, \mathbf{q}) = \frac{1}{2}\left(1 - \frac{\dot{\mathbf{q}}_{\text{NCDS}} \cdot \nabla_{\mathbf{q}} \Gamma_{\mathcal{S}}(\mathbf{x}, \mathbf{q})}{|\dot{\mathbf{q}}_{\text{NCDS}}| \; |\nabla_{\mathbf{q}} \Gamma_{\mathcal{S}}(\mathbf{x}, \mathbf{q})|}\right). \tag{21}$$

Although the foregoing barrier functions can be used independently, it is possible to leverage the properties of both barrier functions (18) and (20) or (21) by simply defining $b(\mathbf{x}, \mathbf{q}) = b_{\text{inv}}(\mathbf{x}, \mathbf{q}) b_{\text{swept}}(\mathbf{x}, \mathbf{q})$. Given the barrier function $b(\mathbf{x}, \mathbf{q})$, the infinitesimal generator results in,

$$\mathbf{V}(\mathbf{x}, \mathbf{q}) = -b(\mathbf{x}, \mathbf{q}) \nabla_{\mathbf{q}} \Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q}). \tag{22}$$

3) Contraction-Preserving Differential Coordinate Change: Given the infinitesimal generator $\mathbf{V}$ (22) obtained from the SDF introduced in Section IV-B2, we know the joint directions that locally maximize obstacle avoidance. Directly forcing the robot away from the obstacle via,

$$\dot{\mathbf{q}} = \hat{f}_c(\mathbf{q}) = f_c(\mathbf{q}) - b(\mathbf{x}, \mathbf{q}) \nabla_{\mathbf{q}} \Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q}), \tag{23}$$

as proposed by Ravichandar and Dani [52], could violate the contraction conditions of Theorem 2. This is because the $b(\mathbf{x}, \mathbf{q}) \nabla_{\mathbf{q}} \Gamma_{\text{SDF}}(\mathbf{x}, \mathbf{q})$ term is not guaranteed to be negative definite, potentially leading to unstable behaviors.

To achieve obstacle avoidance without breaking contraction stability, we propose reshaping the contractive flow f_c around the obstacle by defining a mapping to a new obstacle-free manifold $\mathcal{Y}$ that preserves contraction. To do this, we leverage Theorem 1, which tells us that contraction is preserved under coordinate transformations or diffeomorphisms. Thus, we cast the contraction-preserving obstacle avoidance problem as finding a diffeomorphism that reshapes our contractive dynamical system f_c around the obstacle as a function of the gradient of its implicit representation $\nabla_{\mathbf{q}} \Gamma_{\text{SDF}}$. As explained in Section III-B, a diffeomorphism ψ can be designed from a flow. If such a flow is differentiable, it is thus a valid diffeomorphism.

We propose to leverage the flow generated by the infinitesimal generator (22). Specifically, using the flow $\psi \colon \mathcal{Y} \to \mathcal{C}$ as defined by Equation (6), we transform the obstacle-free manifold $\mathbf{y} \in \mathcal{Y}$ into the configuration space $\mathbf{q} \in \mathcal{C}$ as follows,

$$\begin{aligned} \psi(\mathbf{x}, \mathbf{y}) &= \gamma(\mathbf{x}, \mathbf{y}, t) = \mathbf{q}, \\ &= \mathbf{y} - \int_0^t b\big(\mathbf{x}, \gamma(\mathbf{x}, \mathbf{y}, u)\big) \nabla_{\mathbf{y}} \Gamma_{\text{SDF}}\big(\mathbf{x}, \gamma(\mathbf{x}, \mathbf{y}, u)\big) du. \end{aligned} \tag{24}$$

Similarly, the corresponding inverse flow is,

$$\psi(\mathbf{x},\mathbf{q})^{-1} = \gamma(\mathbf{x},\mathbf{q},-t)^{-1} = \mathbf{y},$$
$$= \mathbf{q} - \int_{-t}^{0} b\big(\mathbf{x},\gamma(\mathbf{x},\mathbf{q},u)^{-1}\big)\nabla_{\mathbf{q}}\Gamma_{\text{SDF}}\big(\mathbf{x},\gamma(\mathbf{x},\mathbf{q},u)^{-1}\big)du. \tag{25}$$

Given the diffeomorphism (24), we can carry out a differential coordinate change,

$$\delta\mathbf{y} = \mathbf{J}_{\psi}{}^{-1}\delta\mathbf{q}, \tag{26}$$

using the Jacobian of the flow $\mathbf{J}_{\psi}$. We refer to this transformation as *Signed Distance Field Differential Coordinate Change (SDDC)*. Based on Theorem 1, which states that contraction is preserved under differential coordinate transformations, we conclude that the transformed system f_{SDDC} is contractive. This therefore ensures that the robot can successfully avoid obstacles while preserving the stability of the underlying skill.

4) Contraction-Preserving Diffeomorphic Transform: In Equation (26), we consider only the differential coordinate change, which captures how velocity and direction transform under the diffeomorphism ψ. However, to fully account for geometric properties (e.g., lengths, distances, and angles), we must consider the Riemannian metric of the obstacle-avoiding manifold $\mathcal{Y}$ induced by ψ [32]. By applying the diffeomorphism ψ, the original dynamics f_c, defined on $\mathcal{C}$, can equivalently be described on the obstacle-avoiding manifold $\mathcal{Y}$ via,

$$\dot{\mathbf{y}} = f_{\text{SDC}}(\mathbf{x},\mathbf{y}) = \mathbf{J}_{\psi}(\mathbf{y})^{-1} f_c(\psi(\mathbf{x},\mathbf{y})). \tag{27}$$

where $\mathbf{J}_{\psi}$ denotes the Jacobian of ψ. We refer to this transformed dynamical system as *Signed Distance Field Coordinate Change (SDC)*. This coordinate change also induces the Riemannian metric $\mathbf{G}_{\psi} = \mathbf{J}_{\psi}^{\top}\mathbf{J}_{\psi}$, which characterizes the local geometry on $\mathcal{Y}$ and corresponds to the contraction metric. A proof of this equivalence is provided in Appendix B4. To show that contraction is invariant under a change of coordinates we can use Theorem 1. Consequently, the proposed SDC preserves the contraction property. We illustrate our contraction-preserving obstacle avoidance methods SDDC and SDC using a toy example in Fig. 6a. This example shows that, given an obstacle, the original vector field f_{c} from Fig. 3 is reshaped around the obstacle by the SDDC and SDC, thus avoiding it successfully.

5) Enhancing Obstacle Avoidance with a Friction Term: In the following, we introduce an extension to the proposed obstacle avoidance methods. To avoid notational clutter, we represent the vector field modulated by both the SDDC (26) and SDC (27) collectively as f_{m}. As observed in Fig. 6a, the modulated velocity profile of f_{m} may deviate significantly from the original velocity profile learned by the NCDS skill. To address this, we introduce a friction term aimed at preserving the velocity profile of the underlying contractive vector field f_{c}. Interestingly, this friction term can also be employed to reduce the velocity near obstacles and to damp fluctuations caused by discontinuities in the learned SDF. Inspired by [19],

Algorithm 2: Modulated Robot Skill Execution

Input : Learned NCDS $f_{\text{NCDS},\boldsymbol{\theta}}$, learned SDF $\Gamma_{\mathcal{R},\hat{\theta}}$
Initialize: $N \leftarrow$ maximum number of steps, $t \leftarrow 0$
while $t \leq N$ **do**
  Sample point cloud $\{\mathbf{s}_i\}_{i=1}^{M} \sim \mathcal{S}$ from the scene
  Read robot joint and task state $\mathbf{q}_t \in \mathcal{C}, \mathbf{x}_t \in \mathcal{X}$
  Find nearest point $\mathbf{s}^*$ to robot surface such that $\Gamma_{\mathcal{R}}(\mathbf{s}^*,\mathbf{q}) = \min_{\{\mathbf{s}_i\}_{i=1}^{M}} \Gamma_{\mathcal{R}}(\mathbf{s}_i,\mathbf{q})$
  Solve the ODE from Equation (24) $\rightarrow \psi(\mathbf{x},\mathbf{q}_t)$
  Solve ODE from Equation (10) $\rightarrow \dot{\mathbf{y}}_{\text{NCDS},t}$
  Modulate $\dot{\mathbf{y}}_{\text{NCDS},t}$ via Equation (26) or (27) $\rightarrow \dot{\mathbf{y}}_t$
  Integrate modulated joint velocity $\dot{\mathbf{y}}_t \rightarrow \mathbf{y}_{t+1}$
  Send $\mathbf{y}_{t+1}$ to the robot's joint impedance controller
  $t \leftarrow t+1$;

our friction term is defined as,

$$\dot{\mathbf{y}}_f = \eta_f \frac{\|f_{\text{c}}(\mathbf{y})\|}{\|f_{\text{m}}(\mathbf{x},\mathbf{y})\|} f_{\text{m}}(\mathbf{x},\mathbf{y}). \tag{28}$$

When $\eta_f = 1$, the resulting dynamics adapts the velocity magnitude to the underlying learned vector field f_{c}, but keeps the modulated direction from f_{m}. Note that $\eta_f = 1$ must not be chosen for SDDC, because it just guarantees strict collision avoidance at velocity level. If we enforce the SDDC to keep the underlying velocity, it could potentially force the system to penetrate the obstacle.

To slow down near obstacles, we design n_f as follows,

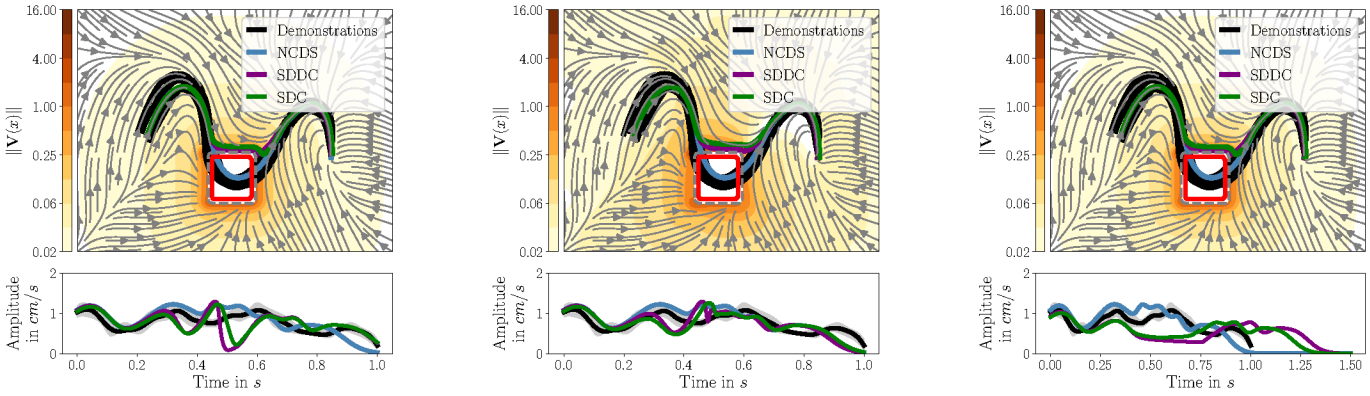
$$\eta_f = 1 - \frac{1}{(1+\beta_f \Gamma_{\text{SDF}}(\mathbf{x},\mathbf{y}))^2}, \tag{29}$$

with hyperparameter β_f. The resulting vector field under this friction term is depicted in Fig. 6c. Since $\frac{\|f_{\text{c}}(\mathbf{y})\|}{\|f_{\text{SDC}}(\mathbf{x},\mathbf{y})\|}$ is strictly positive and $\eta_f > 0$, they induce no directional changes. According to Theorem 1, contraction is preserved under affine transformations, consequently the friction (28) does not compromise the contraction guarantees.

6) Modulated Skill Execution: Finally, trajectories can be integrated along f_{m}, defined by the Equations (26) or (27). First, f_{c} is computed and then mapped into the obstacle-avoiding space via SDC (27) or SDDC (26). Note that integrating the flow (24) requires solving an Ordinary Differential Equation (ODE), which we address using off-the-shelf solvers (more information is provided in the Appendix in Table V). After computing the flow, applying Equation (26) or (27) reshapes the vector fields to avoid obstacles while preserving global contraction guarantees. Algorithm 2 summarizes how a robot is controlled using our approach.

V. Obstacle Avoidance Metrics

We here introduce two novel metrics: Relative Flow Curvature (RFC) and Vector Field Misalignment (VM), to quantitatively assess the obstacle avoidance performance of our approach and state-of-the-art methods. In contrast to standard metrics such as the Dynamic Time Warping Distance



(a) Without friction (b) With swept features (Eq. (21)) (c) With friction and η_f defined by Eq. (29)

Figure 6: Comparison of contraction-preserving obstacle avoidance using SDDC and SDC on an example of the LASA dataset. The vector field modulated by the SDDC is represented by a gray flow stream while the box-shaped obstacle is depicted by the red solid line. The bottom plots show the magnitude of the velocity profile along the trajectory.

(DTWD), that measure reconstruction errors, the proposed metrics focus instead on changes of the learned vector field introduced by the obstacle avoidance modulation.

A. Relative Flow Curvature

To ensure that the obstacle avoidance maneuvers do not introduce jerky or sharp trajectories, we analyze the curvature of the modulated vector field. This metric, termed RFC, should ideally remain similar to the original vector field's curvature, thus indicating smooth and gradual adjustments around obstacles. Specifically, RFC compares the maximal curvature κ along the modulated trajectory τ_m against the maximal curvature of the base trajectory τ_{base}. The curvature κ measures how sharply a trajectory τ bends at a given point relative to the tangential direction [48], and it is defined as,

$$\kappa(t) = \frac{\|\ddot{\tau}(t) \times \dot{\tau}(t)\|}{\|\dot{\tau}(t)\|^3}. \tag{30}$$

with $\times$ denoting the cross product, $\dot{\tau}(t)$ and $\ddot{\tau}(t)$ being the velocity and acceleration along the trajectory. Given the curvature (30), the RFC metric is defined as follows,

$$\text{RFC} = \left| \max_{\mathbf{x} \in \tau_{\text{base}}} \|\kappa(\mathbf{x})\| - \max_{\mathbf{x} \in \tau_m} \|\kappa(\mathbf{x})\| \right|. \tag{31}$$

B. Vector Field Misalignment

When considering contractive stable vector fields, it is particularly important to preserve the learned vector field patterns as much as possible in order to maintain the essence of the skill. Therefore, we propose to compute the cosine similarity between the learned vector field f_c and the modulated vector field f_m along the rolled-out obstacle-avoiding trajectory. Since we aim to quantify the degree of misalignment between the two vector fields, we define this metric as,

$$\text{VM} = \frac{1}{|\tau_m|} \sum_{\mathbf{x} \in \tau_m} \arccos\left(\frac{f_c(\mathbf{x}) \cdot f_m(\mathbf{x})}{|f_c(\mathbf{x})|\ |f_m(\mathbf{x})|} \right) \tag{32}$$

A high VM value indicates that the base vector field and the modulated vector field significantly deviate from each other. Therefore a low VM is desirable as this may indicate that the learned vector field patterns are just slightly modified.

VI. Experiments

In the following experiments, we systematically analyze the performance, real-time feasibility, and potential edge-cases of the proposed obstacle avoidance methods. Thereby, we evaluate the proposed methods SDDC and SDC on the 2D LASA dataset [34] and on two real-world tasks: Emptying a Dishwasher and Opening a Dishwasher. These experiments are carried out in a kitchen environment with unknown obstacles. For the real-world experiments, the robot surface is learned as an SDF function while the scene is represented by an incrementally-sampled point cloud. For the 2D LASA dataset experiments, only implicit models of the obstacles are learned while the robot is assumed to be a virtual point. Here, simple geometries such as circles, rectangles, triangles or arcs are used as obstacles, as shown in Fig. 7.

Our methods are compared against MM [19] as this is the only contraction-preserving obstacle avoidance method known to us, which was also employed in combination with NCDS [4]. Moreover, we compare our methods against the DT [71], which employs diffeomorphic transformations but relies on a natural gradient formulation and thus does not

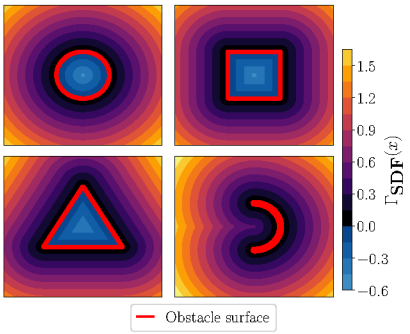


Figure 7: Illustration of the obstacles SDF used in the 2D LASA dataset experiments. The contour lines represent the distance Γ_{SDF} to the obstacle surface, depicted as a solid red line. Four simple obstacles are considered: Circle, Box, Triangle, Arc.

preserve contraction. Additionally, we consider classical Artificial Repulsive Potential Field (ARPF) [27]. We ablate our methods using different SDF functions based on MLP, BP, and simple geometric primitives. These different versions are compared against Hilbert maps [50] as this method was used in combination with DT for obstacle avoidance [71].

A. Evaluation Metrics

The following metrics are used to quantitatively evaluate the obstacle avoidance performance. Firstly, we evaluate our method's ability to preserve the underlying learned skill during obstacle avoidance. To do so, we compare the skill's execution with and without obstacles using the DTWD [6]. A low DTWD indicates that the reproduced trajectory was slightly modified by the obstacle avoidance term. Secondly, we compute the minimum distance $\mathrm{D}_{\min} = \min_{\mathbf{x}\in\tau_m} \Gamma_{\mathrm{SDF}}(\mathbf{x})$ between the robot surface and the obstacle over the course of the skill execution. Our goal should be to keep this distance sufficiently large to guarantee a collision-free skill execution. Thirdly, to prevent obstacle avoidance from generating abrupt or discontinuous trajectories, we use our proposed RFC metric (31) to measure the curvature of the modulated vector field. As fourth metric we analyze the trajectory jerkiness, which measures abrupt acceleration changes possibly introduced by the obstacle avoidance term. To assess this, we compare the maximal trajectory jerkiness in both obstacle-free $\dddot{\tau}_{\mathrm{base}}(\mathbf{x})$ and obstacle-avoidance settings $\dddot{\tau}_m(\mathbf{x})$, resulting in the *Modulation Jerk (MJ)* metric. The jerk difference should be as small as possible. Finally, we evaluate the proposed VM metric (32), which measures the deviation between the base vector field and the modulated vector field. Details about the DTWD, MJ and $\mathrm{D}_{\min}$ metrics are given in Appendix C1.

B. Implementation Details

All experiments are run on a NVIDIA RTX A2000 12GB GPU and a 12th Gen Intel(R) Core(TM) i5-12600K CPU. Next, we provide the architectures and the hyperparameters used for learning the skill models and the obstacle implicit representations.

1) Neural Contractive Dynamical System: Our NCDS implementation fundamentally follows that of Beik-Mohammadi et al. [4]. The NCDS Jacobian network is modeled using an MLP with 2 hidden layers, each consisting of 100 units and employing TANH activation functions. We define the total loss function as a combination of Equations (12), (13) and (14), with weighting factors $\beta_\epsilon = 1.0 \times 10^{-3}$ and $\beta_{\text{noise}} = 1.0$ All models are trained for 1000 epochs with a learning rate of 1×10^{-3} using the Adam optimizer. A decay factor of 0.1 is applied every 250 epochs to improve training stability. Furthermore, the demonstration data of each skill are linearly interpolated such that each trajectory consists of exactly 1000 points. At inference time, a fourth-order Runge-Kutta solver with the 3/8 rule is employed to solve the ODE in Equation (10) using two solver steps.

2) Signed Distance Fields: For the LASA dataset experiments, we first model the SDF based on the mathematical description of the simple geometric shapes, as displayed in Fig. 7. Using this representation, we sample $N = 160K$ points $\{\mathbf{x}_i\}_{i=0}^N$ from a grid around the obstacle surface and we then assign each point its corresponding SDF $d_i = \Gamma_{\mathrm{SDF}}(\mathbf{x}_i)$ and gradient values $\mathbf{g}_i = \nabla_{\mathbf{x}}\Gamma_{\mathrm{SDF}}(\mathbf{x}_i)$. Given this training set $\mathcal{D} = \{(\mathbf{x}_i, d_i, \mathbf{g}_i)\}_{i=0}^N$, a two-dimensional SDF is learned via an MLP with an hourglass-shaped architecture of size (128, 64, 32, 32) and RELU activation functions, inspired by [15, 35]. The loss function is a combination of a reconstruction loss $\mathcal{L}_{\mathrm{SDF}}$ [47], Eikonal loss [15], gradient-based loss [45], and a tension loss [24], leading to,

$$\mathcal{L}(\boldsymbol{\theta}) = w_{\mathrm{SDF}}\mathcal{L}_{\mathrm{SDF}} + w_{\mathrm{grad}}\mathcal{L}_{\mathrm{grad}} + w_{\mathrm{eik}}\mathcal{L}_{\mathrm{eik}} + w_{\mathrm{ten}}\mathcal{L}_{\mathrm{ten}}, \quad (33)$$

where w_{SDF}, w_{grad}, w_{eik} and w_{tension} are weighting factors with values $w_{\mathrm{SDF}} = 10.0$, $w_{\mathrm{eik}} = 0.01$, $w_{\mathrm{grad}} = 0.1$ and $w_{\mathrm{ten}} = 0.01$. The models are trained for 5000 epochs using an Adam optimizer with a learning rate of 1×10^{-3}. In addition, we learn a BP-based SDF similarly to Li et al. [36]. We choose a model size of 8 basis functions, which are trained for 200 epochs using the training set $\mathcal{D}$.

For the robot experiments, we learn an implicit representation of the 7-DoF Franka-Emika Panda robot. On the one hand, we use the BP architecture proposed by Li et al. [36]. We choose 14 basis functions and train the model for 200 epochs. On the other hand, we learn a CDF of the robot based on a MLP model combined with positional encoding and adopt the architecture of Li et al. [35]. We choose the same loss as described for the 2D MLP model but with an architecture of size (1024, 512, 256, 128, 128) and learn the model for 50000 epochs.

C. Baselines

To compare our approach with state-of-the-art methods, we must adapt them to work with SDFs. Firstly, Huber et al. [19] used MM with a custom distance function Γ_{M} designed for simple geometric primitives, which is not directly comparable to our framework. Thus, we define a distance function such that $\mathcal{H}^s = \{\mathbf{x} \in \mathbb{R}^d \mid \Gamma_{\mathrm{M_{SDF}}}(\mathbf{x}) = 1\}$, resulting in,

$$\Gamma_{\mathrm{M_{SDF}}}(\mathbf{x}) = \left(\Gamma_{\mathrm{SDF}}(\mathbf{x}) + \mathbf{1}\right)^{2p}. \quad (34)$$

Given this distance function $\Gamma_{\mathrm{M_{SDF}}}$, we compute the modulation matrix $\mathbf{M}$ according to Huber et al. [19]. So, the modulated vector field is,

$$\dot{\mathbf{x}} = \mathbf{M}(\mathbf{x}) f_{\mathrm{c}}(\mathbf{x}). \quad (35)$$

Secondly, to compare against the DT approach, we replace the diffeomorphism based on Hilbert maps [71] by our diffeomorphism ψ from Eq. (24). This results in the modified natural gradient system,

$$\dot{\mathbf{y}} = f_{\mathrm{DT}}(\mathbf{x}, \mathbf{y}) = \mathbf{G}_\psi(\mathbf{y})^{-1} f_c(\psi(\mathbf{x}, \mathbf{y})) \quad (36)$$

with the Riemannian Metric $\mathbf{G}_\psi = \mathbf{J}_\psi^\top \mathbf{J}_\psi$. Finally, we adapt the ARPF method [27] by defining the repulsive potential

V_{ARPF} via the SDF function, as follows,

$$V_{\text{ARPF}}(\mathbf{x}) = \frac{1}{2}\eta\left(\frac{1}{\Gamma_{\text{SDF}}(\mathbf{x}) - t_{\text{safe}}} - 1\right)^2, \tag{37}$$

with safety threshold $t_{\text{safe}} = 0.1$ and a positive constant $\eta = 10^{-4}$. Given this repulsive potential, the resulting dynamics of the adapted contractive system f_c is,

$$\dot{\mathbf{x}} = f_c - \nabla_{\mathbf{x}} V_{\text{ARPF}}(\mathbf{x}). \tag{38}$$

D. Validation on the LASA Dataset

The aim of these experiments is to determine the quality and efficiency of obstacle avoidance as well as the limitation of the proposed methods on the basis of simplified scenarios.

1) Experimental Setup: The LASA dataset [34] is used to validate the proposed pipeline using a 2D dataset, consisting of 30 2D handwritten movements. Each movement consists of seven demonstrations. As an example, we use the *Sine* movement from the LASA dataset for the following experiments (see Fig. 3). In this case, we assume a virtual point-shaped robot following the learned contractive NCDS models trained on the *Sine* dataset. The task for the robot is to avoid one of the obstacles shown in Fig. 7. In addition, in the Appendix (see Fig. 14) multi-modal robot skills are also investigated.

The obstacle is represented by an implicit distance function. Here, we use three different SDF methods for this purpose. First, we define distance functions w.r.t geometric primitives (see Fig. 7). These are also used to generate the training data, which serve as the ground truth. Additionally, we learn an SDF represented by a MLP and another represented via BP. We analyze the performance of the methods for 40 different trajectories per obstacle, each with different obstacle positions.

2) Obstacle Avoidance Results: As an exemplary scene, we place the box-shaped obstacle in the trajectory path and apply our proposed method and compute both the trajectory with and without obstacle avoidance. As shown in Fig. 6c the learned contractive vector field is deflected around the obstacle and thus we can successfully avoid an convex obstacle using our proposed method. If the barrier function is additionally extended with the *swept* feature, Fig. 6b shows that the vector field away from the obstacle is less influenced and thus the movement in this area follows more closely the dynamics learned by NCDS.

However, the obstacles considered so far have convex shapes. In the following, we investigate how the methods behave with concave surfaces. As shown in Fig. 8, when using the SDT framework, the flow gets stuck in the concave region. Note that the learned SDF does not provide any global information regarding the shape of the obstacle. The curvature can therefore only be calculated locally. Consequently, the vector field is deflected at the edges of the obstacle profile, regardless of the obstacle shape. Therefore, Huber et al. [19] suggest defining a reference point $\mathbf{x}^r$ that specifies a clear direction for the flow. However, this does not serve any purpose for a SDF function, as the global information is only implicitly represented. In addition, the reference point would

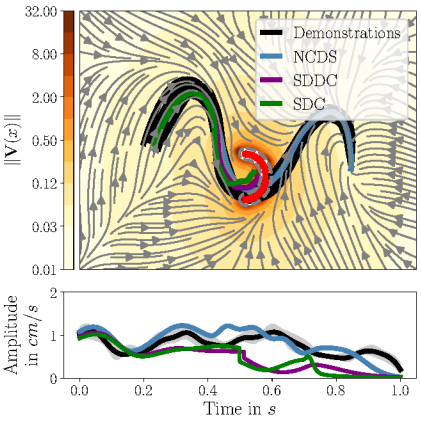


Figure 8: Obstacle avoidance with SDDC and SDC using a inverse barrier on a NCDS model trained on the LASA dataset. The arc-shaped obstacle is depicted by the red line. The velocity profile shows the absolute velocity value along the trajectory. The vector field modulated by the SDDC is represented by gray arrows.

have to be adjusted continuously for dynamic objects. This confirms our expectations that avoidance of concave obstacles may prevent a complete execution of the trajectory, thus we consider our method primarily relevant for convex obstacles.

3) Barrier Functions Hyperparameter: We here analyze the influence of the hyperparameter s_{grad} in Eq. (19) on the obstacle avoidance behavior, using the metrics proposed in Section VI-A. Table I reports the metrics for three different values of s_{grad}. The experiments show that higher values of s_{grad} increase the minimum distance to the obstacle. This in turn has a negative influence on the DTWD and the VM metrics, as this also increases the area of influence of the obstacle avoidance and thus modulates a larger region of the learned vector field. In addition, the results show that the curvature decreases for larger values of s_{grad}, except for $\text{SDDC}_{\text{inv,swept}}$, and thus the obstacle can be avoided more smoothly. We therefore see a potential trade-off between a smooth modulation with higher distance to the obstacle (i.e., high s_{grad} values), and a more accurate alignment to the skill vector field, but with less smooth modulation (i.e., low s_{grad} values). Moreover, the swept features have a significant positive impact on the majority of the metrics, particularly noticeable in the RFC and VM metric. This can be explained by the fact that these swept features minimize the area of influence of the obstacle avoidance term.

4) Comparison to State-Of-The-Art: To compare our proposed SDC and SDDC methods with state-of-the-art approaches, we first adapt the MM [19], ARPF [27] and DT [71] methods, as detailed in Section VI-C, to ensure a fair comparison. We then evaluate all approaches using three distinct SDFs architectures. The results are summarized in Table II. Overall, both SDC and SDDC outperform the baselines in key metrics such as RFC, VM and DTWD. It is noteworthy that ARPF exhibits particularly strong performance with respect to the MJ metric, indicating jerk-free obstacle avoidance. However, ARPF does not guarantee contraction, as proved in Appendix B3.

Another noticeable finding is that SDDC produces relatively large VM values, indicating weaker alignment with the original

Table I: Quantitative analysis of the influence of s_{grad} on the obstacle avoidance behavior according to Eq. 19. Only obstacles based on geometric primitives (GP) are considered. Each metric is computed as the average over 40 trajectories.

	MJ	RFC	VM	DTWD	D_{min}
$\mathbf{s_{grad} = 0.05}$					
SDDC_{inv}	$0.74e-4$	10.19	0.49	123.8	0.28
$\text{SDDC}_{\text{inv,swept}}$	$0.69e-4$	3.80	0.42	**84.9**	0.29
SDC_{inv}	$5.08e-4$	23.58	0.20	104.1	0.35
$\text{SDC}_{\text{inv,swept}}$	$3.93e-4$	21.26	**0.15**	99.9	0.34
$\mathbf{s_{grad} = 0.10}$					
SDDC_{inv}	$0.62e-4$	8.80	0.57	184.2	0.31
$\text{SDDC}_{\text{inv,swept}}$	$0.62e-4$	**3.75**	0.54	127.8	0.32
SDC_{inv}	$4.81e-4$	15.50	0.26	163.4	0.42
$\text{SDC}_{\text{inv,swept}}$	$3.25e-4$	12.59	0.20	149.9	0.41
$\mathbf{s_{grad} = 0.20}$					
SDDC_{inv}	$\mathbf{0.56e-4}$	8.14	0.69	288.5	0.34
$\text{SDDC}_{\text{inv,swept}}$	$0.62e-4$	3.99	0.70	177.2	0.35
SDC_{inv}	$3.81e-4$	8.58	0.36	269.6	**0.53**
$\text{SDC}_{\text{inv,swept}}$	$3.33e-4$	5.99	0.29	238.6	0.52

contractive vector field during obstacle avoidance. In addition, its low D_{min} and high DTWD values imply that SDDC follows a suboptimal path. This observation is qualitatively reinforced by Fig. 9, where SDDC takes a longer path around the obstacle. Nevertheless, this path is characterized by a particularly smooth trajectory, as indicated by its low RFC value.

The differences among the three SDF functions are noticeably, but do not change the ranking of the methods regarding the metrics. However, the MLP-based SDF tends to exhibit higher MJ and RFC values, indicating increased irregularities. By contrast, the BP-based SDF appears more smooth, which aligns with the features provided by polynomial-based functions. **In conclusion**, the foregoing results demonstrate that our SDC and SDDC methods provide smooth and stable obstacle avoidance behaviors without compromising contraction guarantees and reconstruction accuracy. If closer alignment to the underlying contractive vector field is preferred, SDC is recommended, whereas SDDC offers a more uniform obstacle avoidance strategy.

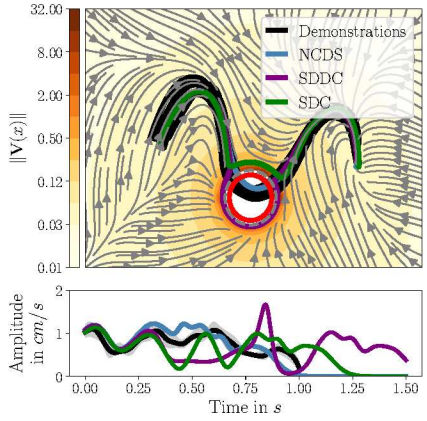


Figure 9: Obstacle avoidance with SDC and SDDC using a inverse barrier on a NCDS trained on the Sine trajectories of the LASA dataset. The circle-shaped obstacle is depicted by the red line. The velocity profile shows the absolute velocity value along the trajectory. The vector field modulated by the SDDC is represented by gray arrows.

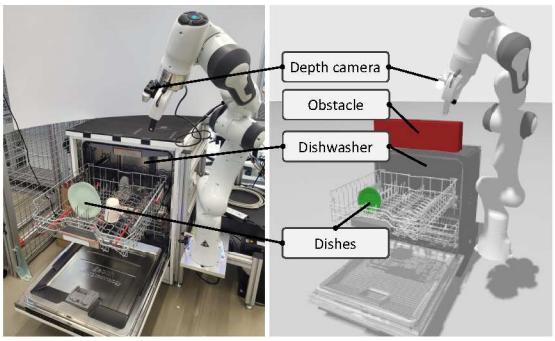


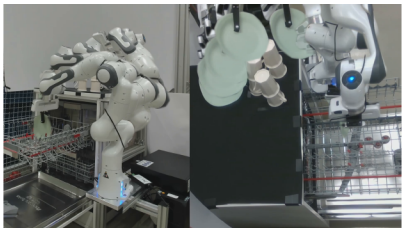
Figure 10: Real-world robot setup in a kitchen environment. It consists of a Franka-Emika Panda robot manipulator with a depth camera attached to its end-effector and a dishwasher filled with a few plastic dishes. The left image shows the real setup, while the right image displays the setup in a Gazebo simulation.

5) Inference Time: Our goal here is to analyze the computational cost of the inference process, which is relevant when implementing these methods in real-world settings. As a reference, consider that NCDS without obstacle avoidance has an inference time of 5.1ms. As shown in Table III, MM and ARPF are the fastest obstacle avoidance methods, with inference times $0.5\text{ms} - 1.0\text{ms}$ faster than the proposed SDDC and SDC. However, the influence of the chosen SDF method becomes evident. While a geometric primitive SDF is computationally cheap, it is only applicable to objects that are known in advance and can be modeled mathematically. In contrast, both MLP and BP representations can handle more complex shapes, where the former provides a faster inference time, making it a preferred choice. Compared to Hilbert Map (HM), the MLP architecture is slower. However, HM consists of radial basis functions and learns an occupancy model [50]. Consequently, HM cannot be employed as a distance function and can only modulate the vector field locally, as used in the diffeomorphic transform approach [71]. This limitation renders HM unsuitable for complex scenes, and it is not possible to maintain a safe distance from obstacles. Additionally, the gradient of HM does not provide a strict safety barrier, since any vector field stronger than the HM gradient can potentially lead to collisions.

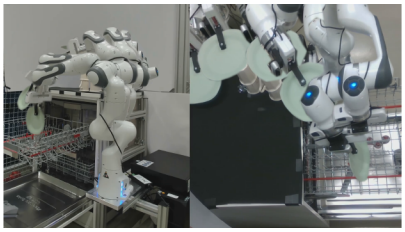
We also report the computational cost when evaluating the diffeomorphic transform for obstacle avoidance as in Eq. (27). We distinguish between the computational time of the flow t_{flow} from Eq. (24) and the computational cost of the Jacobian of Eq. (26). Table IV reports the impact of the ODE solver on the overall inference time. While convex solvers are the fastest, they are restricted to convex surfaces. In contrast, the Runge–Kutta solver is the slowest, with inference times approximately five times higher than those of the convex solver. Furthermore, Table IV shows that the largest portion of the total computation time stems from evaluating the flow in Equation (24). This observation underlines the importance of selecting an efficient implicit distance function to reduce the computation time overhead. Notably, the calculation of the Jacobian has only a minor impact on the total inference time.

Table II: Comparison of the proposed SDC and SDDC methods against state-of-the-art baselines. Three different implicit distance functions are considered for all methods: SDF of geometric primitive (GP) as ground truth, MLP as standard architecture for SDF [15, 47], and BP as polynomial architecture [36]. The methods SDC and DT use a barrier with $s_{\text{grad}} = 0.1$, while SDDC uses a barrier with $s_{\text{grad}} = 0.05$. An additional safety margin of $t_{\text{save}} = 0.1$ applies to all methods. Each metric was averaged over 40 trajectories.

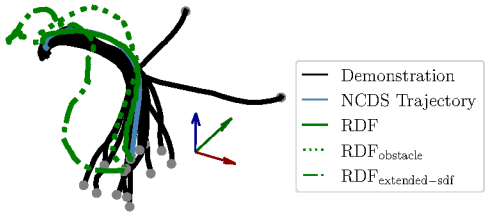
	GP					MLP					BP				
Methods	**MJ**	**RFC**	**VM**	**DTWD**	**D_{min}**	**MJ**	**RFC**	**VM**	**DTWD**	**D_{min}**	**MJ**	**RFC**	**VM**	**DTWD**	**D_{min}**
SDDC_{inv}	$0.62e-4$	8.80	0.57	184.2	0.31	$4.02e-4$	7.78	0.61	174.2	0.34	$1.51e-4$	4.38	0.74	222.4	0.31
$\text{SDDC}_{\text{inv,swept}}$	$0.62e-4$	**3.75**	0.54	127.8	0.32	$3.37e-4$	**6.96**	0.47	129.9	0.33	$1.16e-4$	**4.14**	0.49	134.8	0.30
SDC_{inv}	$5.08e-4$	23.58	0.20	104.1	0.35	$7.44e-4$	28.47	0.20	98.5	0.37	$3.16e-4$	8.42	0.10	98.8	0.34
$\text{SDC}_{\text{inv,swept}}$	$3.93e-4$	21.26	**0.15**	**99.9**	0.34	$6.33e-4$	27.79	**0.15**	**97.7**	0.35	$2.12e-4$	7.71	**0.09**	**95.4**	0.33
DT_{inv}	$0.42e-4$	9.74	0.42	151.3	**0.40**	$3.27e-4$	12.31	0.34	132.9	**0.41**	$1.58e-4$	5.32	0.42	156.8	**0.37**
$\text{DT}_{\text{inv,swept}}$	$0.52e-4$	8.34	0.40	143.4	0.38	$2.67e-4$	8.95	0.34	132.9	0.40	$1.10e-4$	5.67	0.28	147.9	0.34
MM	$2.66e-4$	9.66	0.45	199.4	0.34	$9.13e-4$	18, 63	0.42	177.1	0.35	$0.34e-4$	5, 13	0.62	208.5	0.33
ARPF	**$0.15e-4$**	8.08	0.61	146.6	0.32	**$2.00e-4$**	22, 13	0.57	135.1	0.33	**$0.02e-4$**	6.72	0.84	198.3	0.31



(a) Motion sequence without extended SDF



(b) Motion sequence with extended SDF



(c) Comparison of task space trajectories

Figure 11: EMPTYING THE DISHWASHER task: A series of overlapped snapshots shows the PLACE PLATE skill performed by the Franka-Emika Panda robot using an RDF with grasped plate (left and middle panel). The right plot shows a comparison of the resulting trajectories using different methods w.r.t the provided demonstrations starting from the gray points.

Table III: Comparison of the inference time t_{step} in milliseconds for a single inference step for obstacle avoidance methods based on implicit obstacle representations. The obstacle representations are: SDF of geometric primitive (GP) as ground truth, MLP as standard architecture for SDF [15, 47], BP as polynomial SDF architecture [36], and a Hilbert Map (HM) [50].

Methods	**GP**	**MLP**	**BP**	**HM**
SDC/SDDC/DT	2.2	4.0	9.0	2.6
MM	1.7	3.0	5.7	-
ARPF	1.2	2.5	5.2	-

Table IV: A comparison of the inference time t_{flow} in milliseconds required for a single step in the diffeomorphism ψ (see Equation (24)), where t_{Jacobian} represents the inference time needed to compute the Jacobian $\mathbf{J}_{\psi}$ for the SDC, across different solver types (see Table V in the Appendix). The following obstacle representations method are used: SDF of geometric primitive (GP) as ground truth, MLP as standard architecture for SDF [15] [47], BP as polynomial SDF architecture [36] and a Hilbert Map (HM) [50].

	$\mathbf{t_{flow}}$				$\mathbf{t_{jacobian}}$			
Solvers SDC/SDDC	**GP**	**MLP**	**BP**	**HM**	**GP**	**MLP**	**BP**	**HM**
Convex	1.8	3.4	7.3	1.9	0.4	0.5	1.5	0.6
Euler	3.7	6.7	14.8	3.2	0.5	0.8	3.0	0.9
Runge-Kutta	10.2	18.6	40.5	10.0	1.0	1.4	5.7	2.0

E. Real Robot Experiments

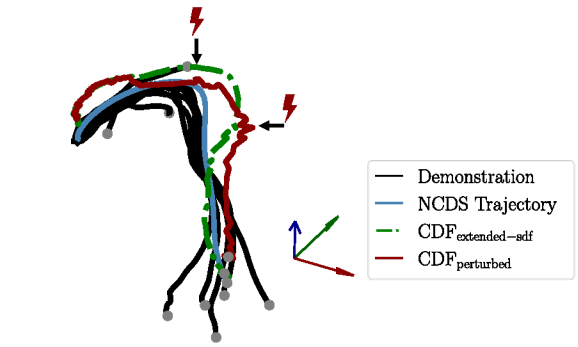
Here we aim at demonstrating the applicability of the proposed obstacle avoidance methods on real-world settings.

1) Experimental Setup: We use a 7 DoF Franka-Emika Panda robot endowed with a depth camera. The robot workspace is a part of a kitchen environment featuring a dishwasher, as shown in Fig. 10. The robot is tasked to learn and execute two real-world tasks: EMPTYING A DISHWASHER and OPENING A DISHWASHER. In all two tasks, the robot should safely and autonomously interact with a dishwasher, maintaining contraction stability against external disturbances and ensuring collision-free operation even with grasped objects. In this context, the robot must not only reach specific target positions but also follow a defined motion trajectory to, for example, remove dishes from the dishwasher and place them while executing a designated motion. Moreover, multiple obstacles, both unknown and cluttered, may be present along the robot's path.

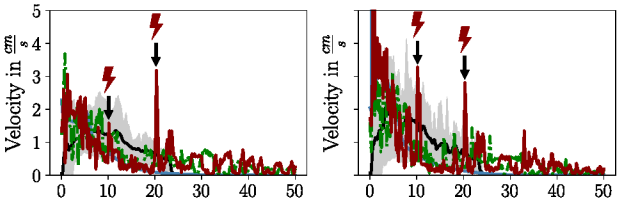
To model the obstacle, we sample a point cloud $\mathbf{P} = \{\mathbf{p}_i\}_{i=0}^{N}$ of the robot workspace on the fly using the depth camera mounted at the robot end-effector. To prevent the robot surface points from being represented in the point cloud, the point cloud is filtered, defining a new point cloud $\mathbf{P}^{+} = \{\mathbf{p_i} \in \mathbf{P} | \Gamma_{\mathcal{R}} > 0\}$. The sampled and filtered point clouds are merged incrementally and subsequently downsampled to a voxel size of 2cm. Using these processed point cloud data, an SDF function of the robot's surface is used for obstacle avoidance by means of the proposed methods SDC and SDDC.

We collect the training data for each skill using kinesthetic teaching. The following skills were recorded: PLACE CUP, PLACE PLATE, PLACE PLATE IN RACK, OPEN DOOR, GRASP DOORHANDLE, MOVE SINE. The recorded trajectories for these skills are shown in Fig. 16 in the Appendix. To execute a robot skill, the integrated output of the corresponding NCDS model is sent out as a reference to the robot joint impedance controller [46, 63], using a control frequency $f_{\text{jimp}} = 5$Hz.

2) Real-world Obstacle Avoidance Behavior: Given the set of contractive skills learned via NCDS, we evaluate our obstacle avoidance method in the tasks: OPEN DISHWASHER WITH



(a) Position trajectories



(b) Velocity time series: Position (left), Orientation (right)

Figure 12: Adaptation robustness test for the PLACE PLATE IN RACK skill performed on the Frank-Emika Panda robot using CDF and under perturbations at $10s$ and $21s$.

STACKED CUPS and PLACE PLATE. Specifically, we test the SDT pipeline with an RDF as distance function and the SDC method. The robot's implicit distance function is extended to include the plate's SDF that was previously learned via Equation (15). The experiment results are shown in Fig. 11. Specifically, Fig. 11b shows a successful obstacle avoidance behavior, without collisions with the stacked cups. Note that, by closely analyzing Fig. 11a, it can be observed that if the robot SDF is not extended with implicit representation of the grasped object a collision can occur. Figure 11c shows the difference among the reproduced trajectories when the plate is attached (where the demonstrations are also plotted as a reference). Notably, the robot avoids both the cups and dishwasher base, through an initial path deviation.

3) Skill Modulation Robustness: We assess the skill robustness using the PLACE PLATE IN RACK skill in the OPEN DISHWASHER WITH STACKED CUPS task. The disturbances that we introduce are: First, moving one joint of the robot at $10s$; and second, directly moving the end-effector at $21s$. The results are shown in Fig. 12, where it is possible to observe sudden velocity changes at $10s$ and $21s$ (see Fig. 12b), matching the aforementioned perturbations. The peak at $23s$ suggests oscillatory behavior caused by the disturbance at $21s$, while the smaller peak at $10s$ reflects the disturbance in joint space, which only affects the orientation of the end-effector. Figure 12a shows that disturbances vanish over time. Initial differences may stem from variations in configuration, but the perturbed path converges to the non-disturbed trajectory, suggesting effective contraction.

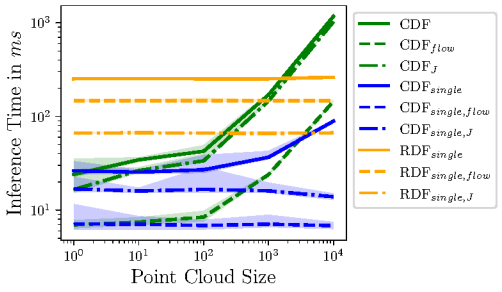


Figure 13: Comparison of the inference times for the SDC method under three different approaches for computing the robot SDF: (1) CDF evaluated at *all* points, (2) CDF evaluated at a *single closest* point, (3) RDF evaluated at a *single closest* point. We decompose the total inference time into two components: The flow inference time and the inference time for computing the Jacobian of the diffeomorphism.

4) Inference Time: To assess the real-time capability of the proposed SDC method, the computation time with respect to the point cloud size among different robot SDF functions is reported in Fig. 13). Note that for large point clouds, it is particularly beneficial to filter out only the nearest point from the input point cloud and then use this point to query the SDT. Using this strategy, the flow and Jacobian computation times remain constant, and therefore the overall inference time increases only marginally, mainly because the robot SDF must handle more query points. However, this additional overhead can be efficiently parallelized.

In contrast, when all points in the point cloud are used for inference, the computation time becomes unaffordable for real-time applications, particularly due to the increased computational load of the Jacobian calculation in the diffeomorphism. Furthermore, the CDF exhibits significantly faster inference times overall and is therefore preferable to the RDF. In addition to the foregoing results, the inference time of the robot SDF as a function of the ODE solver is provided in Table VI in the Appendix.

VII. DISCUSSION

Next we point out the limitations of the proposed approaches and elaborate on possible further research directions.

A. Limitations

1) Concave Obstacles: The experiments have shown that concave obstacle surfaces lead to local stability that limits the execution of robotic skills, indicating that only convex obstacles can be reliably avoided with the proposed methods. Although the minimal concave properties of the learned models, for example, local inaccuracies in the learned implicit distance function, did not show any local instability in the experiments, it could theoretically still occur.

2) Inference Time: To maintain the real-time capability of the framework, one is restricted by the size of the point cloud and the ODE solver used. This could result in less precise obstacle avoidance, especially when resources are limited.

3) Discretization: The discretization of the system states could lead to a corner case. Specifically, if the sampling rate does not match the system dynamics, a collision might occur because the integrated state at the last time step could already lie within an obstacle. Although this issue is inherent to dynamical systems, it can be exacerbated if s_{grad} in Eq. (19) makes the barrier too steep. Furthermore, an overly steep barrier might force the system to operate at high velocities, potentially exceeding system safe limits.

B. Future Work

1) Concave Obstacles: In order to handle concave obstacles, Fourie et al. [13] propose representing obstacles with convex properties using isosurface tracking and manifold modulation to handle non-convex obstacles. However, contraction preservation remains unproven and warrants future investigation. In addition, the barrier function could be extended in such a way that the surface curvature is taken into account and has a stronger effect in non-convex regions in order to prevent local stability.

2) Dynamic Obstacles: In addition to the static obstacles examined so far, dynamic obstacles can also be considered. In general, these have no influence on the stability of the presented approach as long as the sampling rate is significantly higher than the dynamic of the obstacle. Learning the dynamics of the obstacle using an SDF is a substantial challenge and would have to be fundamentally investigated beforehand. If the dynamics $\mathbf{q}_o$ of the obstacles in configuration space are known or estimated, we could potentially incorporate this into a barrier function $b_{\text{swept},o}$, analogous to the swept features in Eqs. (20) and (21), to adapt obstacle avoidance as a function of the relative motion of the obstacle. This function amplifies when the contractive system moves opposite to the direction of the obstacle, but relaxes when moving in the same direction,

$$b_{\text{swept},o}\left(\mathbf{q}_{\text{SDT}}, \mathbf{q}_o\right) = \max\left(0, 1 - \frac{\dot{\mathbf{q}}_{\text{SDT}} \cdot \mathbf{q}_o}{|\dot{\mathbf{q}}_{\text{SDT}}|\ |\mathbf{q}_{\text{SDT}}|}\right). \quad (39)$$

However, the exact effects or alternative approaches would have to be investigated in future research.

3) Inference Time: To further optimize the computation time of the method for real-time use, it is recommended to consider only obstacles that are within the robot's operating space. Furthermore, incrementally learned SDF functions [45] could be leveraged to learn complex scenes as SDF and use them for the proposed framework, which could be particularly relevant for mobile robot use cases. In addition, it would also be conceivable to use a model that directly predicts the distance between two surfaces in order to achieve even more precise obstacle avoidance and to become independent of any point cloud; the paper by Goel and Tabib [14] could be used as a inspiration.

4) Control Barrier Functions: It may also be possible to extend our barrier function approach to include a Control Barrier Function (CBF) [2] that dynamically accounts for system inputs. The assumption is that the safe set defined by a CBF $h(\mathbf{x})$ [56, Eq. (5)]) coincides with the region outside obstacles exactly as given by the SDF. In other words, each obstacle boundary is defined by the zero level set $\Gamma_{\text{SDF}}(\mathbf{x}) = 0$ and similarly the safe-set boundary is defined by $h(\mathbf{x}) = 0$. This correspondence suggests that a CBF constructed from an SDF inherently embeds information regarding both the safe region and the direction of the boundary. We can therefore design a time dependent vector field guided by the gradient of the CBF, whose flow induces a diffeomorphism that repels the system from unsafe areas.

VIII. Conclusion

This paper introduces two novel contraction-preserving whole-body obstacle avoidance methods, namely the Signed Distance Field Differential Coordinate Change (SDDC) and the Signed Distance Field Coordinate Change (SDC). Both methods leverage learned implicit distance functions of the robot's surface, combining their gradients with barrier functions to define a flow that maps learned contractive skills onto an obstacle-avoiding manifold. Specifically, the SDDC relies on a differential coordinate change, while the SDC employs a standard coordinate transformation enabled by the constructed flow. Both approaches are particularly relevant for contractive robot skills or contractive dynamical systems in general, since most existing obstacle avoidance strategies fail to preserve the required stability properties. Furthermore, we propose new obstacle avoidance metrics that facilitate benchmarking and comparative evaluations across different methods. Experimental results show that the proposed approaches effectively preserve the underlying vector field and yield trajectories with minimal curvature changes. Real-world robot experiments confirm the practical applicability of the methods, underscoring the potential of the SDDC and SDC approaches for robust, contraction-preserving obstacle avoidance.

References

[1] Amin Abyaneh, Mahrokh Ghoddousi Boroujeni, Hsiu-Chin Lin, and Giancarlo Ferrari-Trecate. Contractive dynamical imitation policies for efficient out-of-sample recovery. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=lILEtkWOXD. 2

[2] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *European Control Conference (ECC)*, pages 3420–3431, 2019. URL https://doi.org/10.23919/ECC.2019.8796030. 14

[3] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57 (5):469–483, 2009. ISSN 0921-8890. URL https://doi.org/10.1016/j.robot.2008.10.024. 1

[4] Hadi Beik-Mohammadi, Søren Hauberg, Georgios Arvanitidis, Nadia Figueroa, Gerhard Neumann, and Leonel Rozo. Neural contractive dynamical systems. In *International Conference on Learning Representations*

(ICLR), 2024. URL https://openreview.net/forum?id=iAYIRHOYy8. 1, 2, 4, 8, 9

[5] Hadi Beik-Mohammadi, Søren Hauberg, Georgios Arvanitidis, Gerhard Neumann, and Leonel Rozo. Extended neural contractive dynamical systems: On multiple tasks and Riemannian safety regions, 2024. URL https://arxiv.org/abs/2411.11405. 2, 5

[6] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, page 359–370. AAAI Press, 1994. URL https://cdn.aaai.org/Workshops/1994/WS-94-03/WS94-03-031.pdf. 9, 20

[7] Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. *Robot Programming by Demonstration*, pages 1371–1394. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-30301-5. URL https://doi.org/10.1007/978-3-540-30301-5_60. 1

[8] Aude G. Billard, Sylvain Calinon, and Rüdiger Dillmann. *Learning from Humans*, pages 1995–2014. Springer International Publishing, Cham, 2016. ISBN 978-3-319-32552-1. URL https://doi.org/10.1007/978-3-319-32552-1_74. 1

[9] Caroline Blocher, Matteo Saveriano, and Dongheui Lee. Learning stable dynamical systems using contraction theory. In *International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. IEEE, 2017. doi: 10.1109/urai.2017.7992901. URL http://dx.doi.org/10.1109/URAI.2017.7992901. 1

[10] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf. 4

[11] Yiting Chen, Xiao Gao, Kunpeng Yao, Loïc Niederhauser, Yasemin Bekiroglu, and Aude Billard. Differentiable robot neural distance function for adaptive grasp synthesis on a unified robotic arm-hand system, 2023. URL https://arxiv.org/abs/2309.16085. 2, 3, 5

[12] Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics (T-RO)*, 39:1749–1767, 2022. URL http://dx.doi.org/10.1109/TRO.2022.3232542. 1

[13] Christopher K. Fourie, Nadia Figueroa, and Julie A. Shah. On-manifold strategies for reactive dynamical system modulation with nonconvex obstacles. *IEEE Transactions on Robotics (T-RO)*, 40:2390–2409, 2024. URL https://doi.org/10.1109/TRO.2024.3378179. 2, 14

[14] Kshitij Goel and Wennie Tabib. Distance and collision probability estimation from gaussian surface models, 2024. URL https://arxiv.org/abs/2402.00186. 14

[15] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning (ICML)*, pages 3789–3799. PMLR, 2020. URL https://proceedings.mlr.press/v119/gropp20a.html. 2, 5, 9, 12, 19

[16] Victor Hernandez Moreno, Steffen Jansing, Mikhail Polikarpov, Marc G. Carmichael, and Jochen Deuse. Obstacles and opportunities for learning from demonstration in practical industrial assembly: A systematic literature review. *Robotics and Computer-Integrated Manufacturing*, 86:102658, April 2024. ISSN 0736-5845. URL http://dx.doi.org/10.1016/j.rcim.2023.102658. 1

[17] Yanlong Huang, Leonel Rozo, João Silvério, and Darwin G Caldwell. Kernelized movement primitives. *International Journal of Robotics Research (IJRR)*, 38(7):833–852, 2019. URL https://doi.org/10.1177/0278364919846363. 1

[18] Lukas Huber, Aude Billard, and Jean-Jacques Slotine. Avoidance of convex and concave obstacles with convergence ensured through contraction. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1462–1469, April 2019. ISSN 2377-3774. URL http://dx.doi.org/10.1109/LRA.2019.2893676. 1, 2

[19] Lukas Huber, Jean-Jacques Slotine, and Aude Billard. Avoiding dense and dynamic obstacles in enclosed spaces: Application to moving in crowds. *IEEE Transactions on Robotics (T-RO)*, 38(5):3113–3132, 2022. doi: 10.1109/TRO.2022.3164789. 1, 2, 7, 8, 9, 10

[20] James E. Humphreys. *Introduction to Lie Algebras and Representation Theory*. Springer New York, 1972. ISBN 9781461263982. doi: 10.1007/978-1-4612-6398-2. URL http://dx.doi.org/10.1007/978-1-4612-6398-2. 4

[21] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25:328–373, 2013. URL https://doi.org/10.1162/NECO_a_00393. 1

[22] Muhammad Zubair Irshad, Mauro Comi, Yen-Chen Lin, Nick Heppert, Abhinav Valada, Rares Ambrus, Zsolt Kira, and Jonathan Tremblay. Neural fields in robotics: A survey, 2024. URL https://arxiv.org/abs/2410.20220. 2

[23] Sean Jaffe, Alexander Davydov, Deniz Lapsekili, Ambuj Singh, and Francesco Bullo. Learning neural contracting dynamics: Extended linearization and global guarantees. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=YYnP3Xpv3y. 2

[24] Bert Jüttler and Alf Felis. Least-squares fitting of algebraic spline surfaces. *Advances in Computational Mathematics*, 17(1/2):135–152, 2002. ISSN 1019-7168. doi: 10.1023/a:1015200504295. URL http://dx.doi.org/10.1023/A:1015200504295. 9, 19

[25] Seyed Mohammad Khansari-Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics (T-RO)*, 27(5):943–957, 2011. URL http://dx.doi.org/10.1109/TRO.2011.2159412. 1, 2

[26] Seyed Mohammad Khansari-Zadeh and Aude Billard. A dynamical system approach to realtime obstacle avoidance. *Autonomous Robots*, 32(4):433–454, March 2012. ISSN 1573-7527. URL http://dx.doi.org/10.1007/s10514-012-9287-y. 2

[27] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 500–505, 1985. URL http://dx.doi.org/10.1109/ROBOT.1985.1087247. 1, 9, 10

[28] Jin-Oh Kim and Pradeep K. Khosla. Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, 8(3):338–349, 1992. doi: 10.1109/70.143352. 2

[29] Holger Klein, Noémie Jaquier, Andre Meixner, and Tamim Asfour. On the design of region-avoiding metrics for collision-safe motion generation on Riemannian manifolds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2346–2353, 2023. URL http://dx.doi.org/10.1109/IROS55552.2023.10341266. 6

[30] Mikhail Koptev, Nadia Figueroa, and Aude Billard. Neural joint space implicit signed distance functions for reactive robot manipulator control. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):480–487, February 2023. ISSN 2377-3774. doi: 10.1109/lra.2022.3227860. URL http://dx.doi.org/10.1109/LRA.2022.3227860. 2, 3, 5

[31] Maria Koskinopoulou, Stylianos Piperakis, and Panos Trahanias. Learning from demonstration facilitates human-robot collaborative task execution. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 59–66, 2016. URL https://doi.org/10.1109/HRI.2016.7451734. 1

[32] John M. Lee. *Riemannian Manifolds*. Springer New York, 1997. ISBN 9780387227269. doi: 10.1007/b98852. URL http://dx.doi.org/10.1007/b98852. 3, 7

[33] John M. Lee. *Introduction to Smooth Manifolds*. Springer New York, 2012. ISBN 9781441999825. doi: 10.1007/978-1-4419-9982-5. URL http://dx.doi.org/10.1007/978-1-4419-9982-5. 4

[34] A. Lemme, Y. Meirovitch, M. Khansari-Zadeh, T. Flash, A. Billard, and J. J. Steil. Open-source benchmarking for learned reaching motion generation in robotics. *Paladyn, Journal of Behavioral Robotics*, 6(1), 2015. URL http://doi.org/10.1515/pjbr-2015-0002. 8, 10

[35] Yiming Li, Xuemin Chi, Amirreza Razmjoo, and Sylvain Calinon. Configuration space distance fields for manipulation planning. In *Proc. Robotics: Science and Systems (R:SS)*, 2024. URL https://arxiv.org/abs/2406.01137. 3, 5, 9

[36] Yiming Li, Yan Zhang, Amirreza Razmjoo, and Sylvain Calinon. Representing robot geometry as distance fields: Applications to whole-body manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 15351–15357, 2024. URL http://dx.doi.org/10.1109/ICRA57147.2024.10611674. 2, 3, 5, 9, 12

[37] Puze Liu, Kuo Zhang, Davide Tateo, Snehal Jauhri, Jan Peters, and Georgia Chalvatzaki. Regularized deep signed distance fields for reactive motion generation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6673–6680, 2022. URL https://doi.org/10.1109/IROS47612.2022.9981456. 3

[38] Winfried Lohmiller and Jean-Jacques E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998. ISSN 0005-1098. URL https://doi.org/10.1016/S0005-1098(98)00019-3. 1, 3

[39] Jonathan Lorraine and Safwan Hossain. JacNet: Learning functions with structured Jacobians. *INNF Workshop at the International Conference on Machine Learning (ICML)*, 2019. URL https://invertibleworkshop.github.io/INNF_2019/accepted_papers/pdfs/INNF_2019_paper_10.pdf. 4

[40] Tomas Lozano-Perez. Robot programming. *Proceedings of the IEEE*, 71(7):821–841, 1983. doi: 10.1109/PROC.1983.12681. 1

[41] Ian R. Manchester and Jean-Jacques E. Slotine. Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design. *IEEE Transactions on Automatic Control*, 62(6):3046–3053, 2017. URL http://dx.doi.org/10.1109/TAC.2017.2668380. 3, 19

[42] Ante Marić, Yiming Li, and Sylvain Calinon. Online learning of piecewise polynomial signed distance fields for manipulation tasks, 2024. URL https://arxiv.org/abs/2401.07698. 4

[43] Nicolas Marticorena, Tobias Fischer, Jesse Haviland, and Niko Suenderhauf. Rmmi: Enhanced obstacle avoidance for reactive mobile manipulation using an implicit neural map, 2024. URL https://arxiv.org/abs/2408.16206. 3

[44] Klaus Neumann and Jochen J. Steil. Learning robot motions with stable dynamical systems under diffeomorphic transformations. *Robotics and Autonomous Systems (RAS)*, 70:1–15, 2015. ISSN 0921-8890. URL https://doi.org/10.1016/j.robot.2015.04.006. 2

[45] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. iSDF: Real-time neural signed distance fields for robot perception. In *Robotics: Science and Systems (R:SS)*, 2022. URL https://www.roboticsproceedings.org/rss18/p012.pdf. 2, 5, 9, 14, 19

[46] Christian Ott, Alin Albu-Schaffer, Andreas Kugi, and Gerd Hirzinger. On the passivity-based impedance control of flexible joint robots. *IEEE Transactions on Robotics (T-RO)*, 24(2):416–429, 2008. URL https://doi.org/10.1109/TRO.2008.915438. 12

[47] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, June 2019. URL http://dx.doi.org/10.1109/CVPR.2019.00025. 2, 4, 5, 6, 9, 12, 19

[48] Andrew Pressley. *Elementary Differential Geometry*. Springer London, 2010. ISBN 9781848828919. doi: 10.1007/978-1-84882-891-9. URL http://dx.doi.org/10.1007/978-1-84882-891-9. 8

[49] Eduard Pujol and Antonio Chica. Adaptive approximation of signed distance fields through piecewise continuous interpolation. *Computers and Graphics*, 114:337–346, 2023. ISSN 0097-8493. doi: https://doi.org/10.1016/j.cag.2023.06.020. URL https://www.sciencedirect.com/science/article/pii/S0097849323001139. 4

[50] Fabio Ramos and Lionel Ott. Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *The International Journal of Robotics Research (IJRR)*, 35(14):1717–1730, December 2016. ISSN 1741-3176. URL http://dx.doi.org/10.1177/0278364916684382. 9, 11, 12

[51] Muhammad Asif Rana, Anqi Li, Dieter Fox, Byron Boots, Fabio Ramos, and Nathan Ratliff. Euclideanizing flows: Diffeomorphic reduction for learning stable dynamical systems. In *Conference on Learning for Dynamics and Control (L4DC)*, pages 630–639, 2020. URL https://proceedings.mlr.press/v120/rana20a.html. 1, 2

[52] Harish Ravichandar and Ashwin Dani. Learning contracting nonlinear dynamics from human demonstration for robot motion planning. In *ASME Dynamic Systems, and Control Conference (DSCC)*. American Society of Mechanical Engineers, October 2015. doi: 10.1115/dscc2015-9870. URL http://dx.doi.org/10.1115/DSCC2015-9870. 6

[53] Harish Ravichandar, Athanasios S. Polydoros, Sonia Chernova, and Aude Billard. Recent advances in robot learning from demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1):297–330, May 2020. ISSN 2573-5144. doi: 10.1146/annurev-control-100819-063206. URL http://dx.doi.org/10.1146/annurev-control-100819-063206. 1

[54] Harish chaandar Ravichandar and Ashwin Dani. Learning position and orientation dynamics from demonstrations via contraction analysis. *Autonomous Robots*, 43(4):897–912, May 2018. ISSN 1573-7527. doi: 10.1007/s10514-018-9758-x. URL http://dx.doi.org/10.1007/s10514-018-9758-x. 1, 2

[55] Navid Rezazadeh, Maxwell Kolarich, Solmaz S. Kia, and Negar Mehr. Learning contraction policies from offline data. *IEEE Robotics and Automation Letters (RA-L)*, 7 (2):2905–2912, 2022. URL https://doi.org/10.1109/LRA.2022.3145100. 2

[56] Ivan Dario Jimenez Rodriguez, Noel Csomay-Shanklin, Yisong Yue, and Aaron D. Ames. Neural gaits: Learning bipedal locomotion via control barrier functions and zero dynamics policies. In *Learning for Dynamics and Control Conference (L4DC)*, volume 168, pages 1060–1072, 2022. URL https://proceedings.mlr.press/v168/rodriguez22a.html. 14

[57] Leonel Rozo, Sylvain Calinon, Darwin G. Caldwell, Pablo Jiménez, and Carme Torras. Learning physical collaborative robot behaviors from human demonstrations. *IEEE Transactions on Robotics (T-RO)*, 32(3):513–527, 2016. URL https://doi.org/10.1109/TRO.2016.2540623. 1

[58] Leonel Rozo, Heni Ben Amor, Sylvain Calinon, Anca Dragan, and Dongheui Lee. Special issue on learning for human–robot collaboration. *Autonomous Robots*, 42(5):953–956, 2018. URL https://doi.org/10.1007/s10514-018-9756-z. 1

[59] Leonel Rozo, Andras G. Kupcsik, Philipp Schillinger, Meng Guo, Robert Krug, Niels van Duijkeren, Markus Spies, Patrick Kesper, Sabrina Hoppe, Hanna Ziesche, Mathias Bürger, and Kai O. Arras. The e-bike motor assembly: Towards advanced robotic manipulation for flexible manufacturing. *Robotics and Computer-Integrated Manufacturing*, 85:102637, 2024. ISSN 0736-5845. doi: https://doi.org/10.1016/j.rcim.2023.102637. URL https://www.sciencedirect.com/science/article/pii/S0736584523001126. 1

[60] Stefan Schaal. Learning from demonstration. In *Neural Information Processing Systems (NeurIPS)*, 1996. URL https://proceedings.neurips.cc/paper_files/paper/1996/file/68d13cf26c4b4f4f932e3eff990093ba-Paper.pdf. 1

[61] John W. Simpson-Porco and Francesco Bullo. Contraction theory on riemannian manifolds. *Systems & Control Letters*, 65:74–80, 2014. ISSN 0167-6911. URL https://doi.org/10.1016/j.sysconle.2013.12.016. 3

[62] Sumeet Singh, Spencer M Richards, Vikas Sindhwani, Jean-Jacques E Slotine, and Marco Pavone. Learning stabilizable nonlinear dynamics with contraction-based regularization. *The International Journal of Robotics Research (IJRR)*, 40(10-11):1123–1150, 2021. URL https://doi.org/10.1177/0278364920949931. 1

[63] M. W. Spong. Modeling and control of elastic joint robots. *Journal of Dynamic Systems, Measurement, and Control*, 109(4):310–318, December 1987. ISSN 1528-9028. URL http://dx.doi.org/10.1115/1.3143860. 12

[64] Hang Su, Andrea Mariani, Salih Ertug Ovur, Arianna Menciassi, Giancarlo Ferrigno, and Elena De Momi. Toward teaching by demonstration for robot-assisted minimally invasive surgery. *IEEE Transactions on Automation Science and Engineering*, 18(2):484–494, 2021. URL https://doi.org/10.1109/TASE.2020.3045655. 1

[65] Dawei Sun, Susmit Jha, and Chuchu Fan. Learning certified control using contraction metric. In *Conference on Robot Learning (CoRL)*, pages 1519–1539, 2020. URL https://proceedings.mlr.press/v155/sun21b.html. 2

[66] Hiroyasu Tsukamoto and Soon-Jo Chung. Neural contraction metrics for robust estimation and control: A convex optimization approach. *IEEE Control Systems Letters*, 5(1):211–216, 2021. URL https://doi.org/10.1109/LCSYS.2020.3001646. 2

[67] Hiroyasu Tsukamoto, Soon-Jo Chung, and Jean-Jacques E. Slotine. Contraction theory for nonlinear stability analysis and learning-based control: A tutorial

overview. *Annual Reviews in Control*, 52:135–169, 2021. ISSN 1367-5788. 1, 3

[68] J. Urain, M. Ginesi, D. Tateo, and J. Peters. Imitationflow: Learning deep stable stochastic dynamic systems by normalizing flows. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 5231–5237. IEEE, 2020. URL https://doi.org/10.1109/IROS45743.2020.9341035. 1, 2

[69] Tim Welschehold, Christian Dornhege, and Wolfram Burgard. Learning manipulation actions from human demonstrations. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3772–3777, 2016. URL https://doi.org/10.1109/IROS.2016.7759555. 1

[70] Jiechao Zhang, Hadi Beik Mohammadi, and Leonel Rozo. Learning Riemannian stable dynamical systems via diffeomorphisms. In *Conference on Robot Learning (CoRL)*, 2022. URL https://openreview.net/forum?id=o8dLx8OVcNk. 2

[71] Weiming Zhi, Tin Lai, Lionel Ott, and Fabio Ramos. Diffeomorphic transforms for generalised imitation learning. In *Learning for Dynamics and Control Conference (L4DC)*, pages 508–519, 2022. URL https://proceedings.mlr.press/v168/zhi22a.html. 1, 2, 8, 9, 10, 11

APPENDIX

A. Signed Distance Fields

1) Loss Functions: According to Park et al. [47], learned SDF methods with parameters $\boldsymbol{\theta}$ can be interpreted as a decision boundary that assigns points to either the inside $\mathcal{H}^i$ or outside $\mathcal{H}^f$ of an object, while additionally providing the distance to the surface. To achieve these properties, an appropriate loss function must be chosen. Park et al. [47] propose using an $\mathcal{L}_1$ reconstruction loss,

$$\mathcal{L}_{\text{SDF}}(\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})) = \left|\text{clamp}\left(\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta}),\delta\right) - \text{clamp}(s,\delta)\right|, \tag{40}$$

where $\text{clamp}(x,\delta) := \min(\delta, \max(-\delta, x))$ limits the values within the range $[-\delta,\delta]$, and s defines a boundary for the region over which the SDF function should be defined.

Furthermore, Gropp et al. [15] suggested using the Eikonal loss,

$$\mathcal{L}_{\text{eik}}(\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})) = \left|\|\nabla_{\mathbf{x}}\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})\| - 1\right|, \tag{41}$$

which encourages the SDF to maintain equidistance properties across the domain, meaning that for any point in space, the value of the SDF corresponds to the shortest distance to the surface $\mathcal{H}^s$ [15]. In addition, Ortiz et al. [45] introduced a gradient-based loss,

$$\mathcal{L}_{\text{grad}}(\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta}),\mathbf{g}) = 1 - \frac{\nabla_{\mathbf{x}}\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})\cdot\mathbf{g}}{\|\nabla_{\mathbf{x}}\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})\|\|\mathbf{g}\|}, \tag{42}$$

which penalizes deviations of the gradient of the SDF from the surface normal by computing the cosine distance between the predicted gradient and the true normal vector $\mathbf{g}$. Jüttler and Felis [24] additionally introduced the concept of tension loss, which enhances the smoothness of the function.

$$\mathcal{L}_{\text{tension}}(\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})) = \|\nabla^2_{\mathbf{x}}\Gamma_{\text{SDF}}(\mathbf{x};\boldsymbol{\theta})\|^2 \tag{43}$$

B. Contraction-preserving Obstacle Avoidance

1) Multi-modal demonstrations: Figure 14 displays the evaluation of the SDC on a multi-modal LASA dataset skill. Under a high contraction rate, the green trajectory remains in its initial modal state. Conversely, a reduced contraction rate, as evidenced by the purple trajectory, can induce a mode switch.

2) Flow Solver: Table V shows a overview of different types of solvers that can be used to solve the flow in Eq. (24).

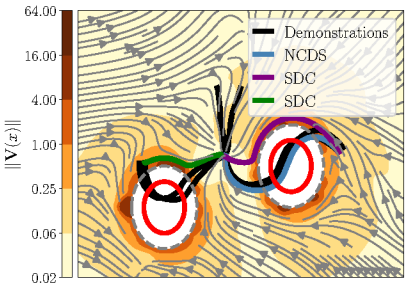


Figure 14: Obstacle avoidance with SDC using an inverse barrier with swept features on an NCDS model trained on a multi-modal LASA dataset skill. Two circular obstacles are depicted by red lines.

Table V: Overview of various ODE solvers

ODE Solver	Description	Pros	Cons	Complexity
Convex	One-step solution of flow	+ Fastest solver	- Only for convex surfaces	$\mathcal{O}(1)$
Euler method	Numerical first-order ODE solver	+ Any curvature + Fast	- Very imprecise	$\mathcal{O}(N)$
Runge-Kutta (RK4)	Numerical fourth-order ODE solver	+ Any curvature + Precise	- Slowest method	$\mathcal{O}(4N)$

3) Reactive Obstacle Avoidance: The following is a proof which shows that reactive obstacle avoidance as in Eq. (23) can violate contraction stability.

Proof: Consider the obstacle-avoidance term as described in Equation (23), we need to show the contraction conditions as described in Theorem 2. Therefore, taking the derivative of $\hat{f}_{\text{c}}$ leads to,

$$\frac{\partial \hat{f}_{\text{c}}}{\partial \mathbf{q}} = \frac{\partial}{\partial \mathbf{q}}\left(f_{\text{c}}(\mathbf{q}) - b(\mathbf{x},\mathbf{q})\nabla_{\mathbf{q}}\Gamma_{\text{SDF}}\right) = \mathbf{F} - \mathbf{H}_{\text{SDF}}(\mathbf{x},\mathbf{q}), \tag{44}$$

where,

$$\mathbf{H}_{\text{SDF}}(\mathbf{x},\mathbf{q}) = \frac{\partial^2\left(b(\mathbf{x},\mathbf{q})\Gamma_{\text{SDF}}(\mathbf{x},\mathbf{q})\right)}{\partial \mathbf{q}^2}, \tag{45}$$

is the Hessian of the SDF with the proposed barrier function.

Substituting $\frac{\partial \hat{f}_{\text{c}}}{\partial \mathbf{q}}$ into Theorem 2, we must satisfy,

$$\dot{\mathbf{G}} + \mathbf{G}\frac{\partial \hat{f}_{\text{c}}}{\partial \mathbf{q}} + \frac{\partial \hat{f}_{\text{c}}^{\top}}{\partial \mathbf{q}}\mathbf{G} \preceq -2\alpha\mathbf{G}. \tag{46}$$

Multiplying out all terms using Equation (23) leads to,

$$\dot{\mathbf{G}} + \mathbf{G}\mathbf{F} + \mathbf{F}^{\top}\mathbf{G} + \mathbf{G}\mathbf{H}_{\text{SDF}} + \mathbf{H}_{\text{SDF}}^{\top}\mathbf{G} \preceq -2\alpha\mathbf{G}. \tag{47}$$

Since f_{c} is contractive by design, we known from Theorem 2 that,

$$\dot{\mathbf{G}} + \mathbf{G}\frac{\partial f_{\text{c}}}{\partial \mathbf{q}} + \frac{\partial f_{\text{c}}^{\top}}{\partial \mathbf{q}}\mathbf{G} \preceq -2\alpha\mathbf{G}, \tag{48}$$

so it remains to show that,

$$\mathbf{G}\mathbf{H}_{\text{SDF}} + \mathbf{H}_{\text{SDF}}^{\top}\mathbf{G} \preceq \mathbf{0}. \tag{49}$$

If $\mathbf{H}_{\text{SDF}}$ has positive eigenvalues, then $\mathbf{G}\mathbf{H}_{\text{SDF}} + \mathbf{H}_{\text{SDF}}^{\top}\mathbf{G}$ can be positive definite, thus violating contraction. As Γ_{SDF} may have any shape, we cannot avoid having positive eigenvalues in $\mathbf{H}_{\text{SDF}}$ and thus violate the contraction condition. ■

4) Signed Distance Field Diffeomorphic Transform: In the following, we prove that the SDC preserves the contraction property and that the contraction metric is equivalent to the Riemannian metric of the obstacle-avoiding manifold $\mathcal{Y}$. The proof is inspired by Manchester and Slotine [41].

Proof:
Consider a contractive system,

$$\dot{\mathbf{q}} = f_{\mathrm{c}}(\mathbf{q}), \tag{50}$$

which can be described via its differential dynamics,

$$\dot{\delta}_{\mathbf{q}} = \mathbf{A}(\mathbf{q}, t)\delta_{\mathbf{q}} \quad \text{s.t.} \quad \mathbf{A} := \frac{\partial f_{\mathrm{c}}}{\partial \mathbf{q}}. \tag{51}$$

Given the differential dynamics, a differential coordinate change $\delta_{\mathbf{q}} = \mathbf{J}_\psi \delta_{\mathbf{y}}$ can be introduced based on the smooth diffeomorphism ψ, defined by the flow in Equation 24. By applying the coordinate change $\delta_{\mathbf{q}} = \mathbf{J}_\psi \delta_{\mathbf{y}}$ to the differential contractive system, this results in,

$$\dot{\delta}_{\mathbf{y}} = \mathbf{A}_{\mathbf{y}}(\mathbf{y}, t)\delta_{\mathbf{y}}. \tag{52}$$

Given $\delta_{\mathbf{q}} = \mathbf{J}_\psi \delta_{\mathbf{y}}$, $\dot{\delta}_{\mathbf{y}}$ can be calculated via,

$$\dot{\delta}_{\mathbf{q}} = \mathbf{J}_\psi \dot{\delta}_{\mathbf{y}} = \dot{\mathbf{J}_\psi} \delta_{\mathbf{y}} + \mathbf{J}_\psi \dot{\delta}_{\mathbf{y}}. \tag{53}$$

Given Equation 51 and the coordinate changes $\delta_{\mathbf{q}} = \mathbf{J}_\psi \delta_{\mathbf{y}}$, Equation (53) can be transformed as follows,

$$\dot{\delta}_{\mathbf{y}} = \underbrace{\left(-\mathbf{J}_\psi^{-1}\dot{\mathbf{J}}_\psi + \mathbf{J}_\psi^{-1}\mathbf{A}\mathbf{J}_\psi\right)}_{\mathbf{A}_y} \delta_{\mathbf{y}}. \tag{54}$$

Note that the contraction metric $\mathbf{G}_{\mathbf{q}}$ of f_{c} is given by $\mathbf{G}_{\mathbf{q}} = \mathbf{I}$, since the contractive system f_{c} is defined in Euclidean space. For the contraction metric $\mathbf{G}_{\mathbf{y}}$ of the dynamical system mapped into the obstacle-avoiding manifold we choose,

$$\mathbf{G}_{\mathbf{y}} = \mathbf{J}_\psi^\top \mathbf{G}_{\mathbf{q}} \mathbf{J}_\psi = \mathbf{J}_\psi^\top \mathbf{J}_\psi, \tag{55}$$

where $\mathbf{J}_\psi$ is the the Jacobian of the flow ψ.

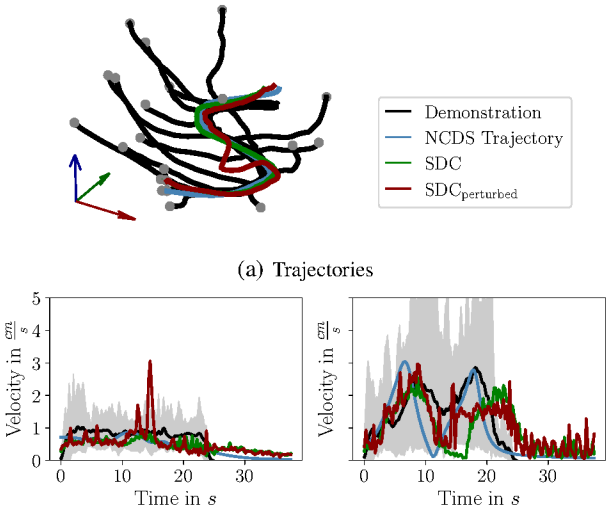
The derivation of the metric results in,

$$\dot{\mathbf{G}}_{\mathbf{y}} = \dot{\mathbf{J}}^\top \mathbf{J} + \mathbf{J}^\top \dot{\mathbf{J}}. \tag{56}$$

Given $\mathbf{G}_{\mathbf{y}}, \dot{\mathbf{G}}_{\mathbf{y}}, \mathbf{A}_{\mathbf{y}}$, the contraction condition from Theorem 2 can be shown in the following,

$$\begin{aligned}
&\dot{\mathbf{G}}_{\mathbf{y}} + \mathbf{A}_{\mathbf{y}}^\top \mathbf{G}_{\mathbf{y}} + \mathbf{G}_{\mathbf{y}} \mathbf{A}_{\mathbf{y}} + 2\alpha \mathbf{G}_{\mathbf{y}} \prec \mathbf{0}, \\
\Leftrightarrow &\dot{\mathbf{J}}\psi^\top \mathbf{J}_\psi + \mathbf{J}\psi^\top \dot{\mathbf{J}}_\psi + \mathbf{J}_\psi^\top \mathbf{J}_\psi (\mathbf{J}_\psi^{-1} A_c \mathbf{J}_\psi - \mathbf{J}_\psi^{-1}\dot{\mathbf{J}}_\psi) + \\
&(\mathbf{J}_\psi^{-1} A_c \mathbf{J}_\psi - \mathbf{J}_\psi^{-1}\dot{\mathbf{J}}_\psi)^\top \mathbf{J}_\psi^\top \mathbf{J}_\psi + 2\alpha \mathbf{J}_\psi^\top \mathbf{J}_\psi \preceq \mathbf{0}, \\
\Leftrightarrow &\dot{\mathbf{J}}\psi^\top \mathbf{J}_\psi + \mathbf{J}\psi^\top \dot{\mathbf{J}}_\psi + \mathbf{J}_\psi^\top A_c \mathbf{J}_\psi - \mathbf{J}_\psi^\top \dot{\mathbf{J}}_\psi + \mathbf{J}_\psi^\top A_c^\top \mathbf{J}_\psi - \\
&\dot{\mathbf{J}}_\psi^\top \mathbf{J}_\psi + 2\alpha \mathbf{J}_\psi^\top \mathbf{J}_\psi \preceq \mathbf{0}, \\
\Leftrightarrow &\mathbf{J}_\psi^\top (\mathbf{A} + \mathbf{A}^\top + 2\alpha \mathbf{I}) \mathbf{J}_\psi \preceq \mathbf{0}.
\end{aligned} \tag{57}$$

It is known from matrix calculations that the definiteness of a matrix $\mathbf{A}$ under $\mathbf{J}_\psi^\top \mathbf{A} \mathbf{J}_\psi$ is preserved. Therefore, only the term $(\mathbf{A} + \mathbf{A}^\top + 2\alpha\mathbf{I})$ must be negative definite. This property holds by the construction of the contractive system, implying that the entire SDC system is also contractive. ■



(a) Trajectories

(b) Velocities Series - Position (left), Orientation (right)

Figure 15: Move Sine skill performed on the robot with external perturbation

C. Experiments

1) Evaluation Metrics: We use the DTWD [6] defined as,

$$\begin{aligned}
&\mathrm{DTWD}\big(\tau_{\text{base}}(\mathbf{x}), \tau_m(\mathbf{x})\big) = \\
&\sum_{j \in l(\tau_{\text{base}}(\mathbf{x}))} \min_{i \in l(\tau_m(\mathbf{x}))} d\big(\tau_{i,\text{base}}(\mathbf{x}), \tau_{j,m}(\mathbf{x})\big) + \\
&\sum_{i \in l(\tau_m(\mathbf{x}))} \min_{j \in l(\tau_{\text{base}}(\mathbf{x}))} d\big(\tau_{i,\text{base}}(\mathbf{x}), \tau_{j,m}(\mathbf{x})\big),
\end{aligned} \tag{58}$$

measuring the similarity between two trajectories, which may vary in length and/or velocity [6]. In our context, τ_{base} denotes the base trajectory of length $l(\tau_{\text{base}})$, and it is compared to a collision-free modulated trajectory τ_m of length $l(\tau_m)$. A low DTWD indicates that the original vector field was slightly modulated by the obstacle avoidance term.

The minimum distance between the robot surface and the obstacle over the course of the skill execution is defined as follows,

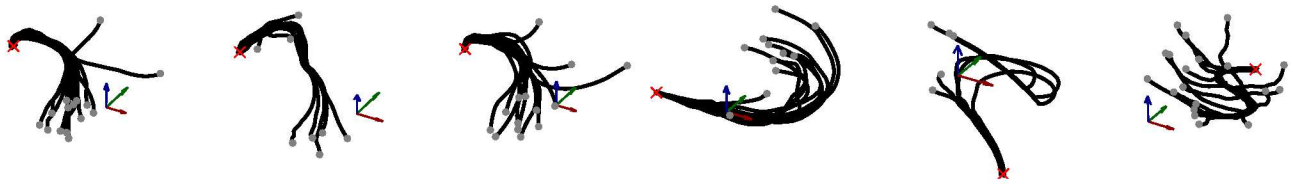
$$D_{\text{min}} = \min_{\mathbf{x} \in \tau_m} \Gamma_{\text{SDF}}(\mathbf{x}). \tag{59}$$

The *MJ* metric is defined as,

$$\text{MJ} = \left| \max_{\mathbf{x} \in \tau_{\text{base}}} \| \dddot{\tau}_{\text{base}}(\mathbf{x}) \| - \max_{\mathbf{x} \in \tau_m} \| \dddot{\tau}_m(\mathbf{x}) \| \right|. \tag{60}$$

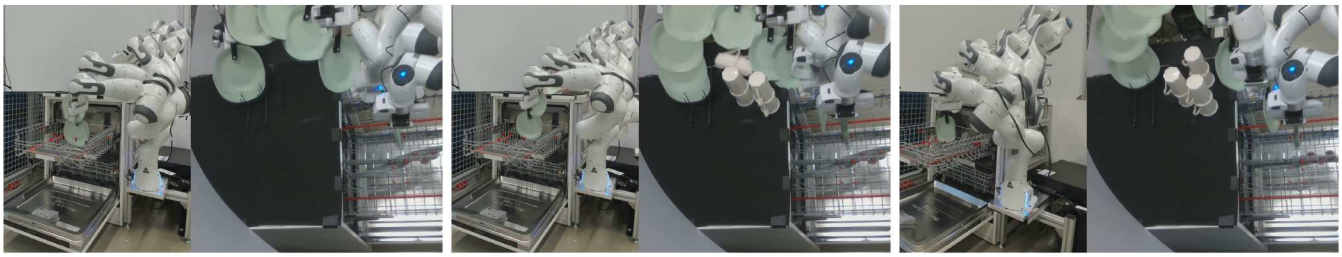
In our context, τ_{base} denotes the base trajectory, and it is compared to a collision-free modulated trajectory τ_m. A high MJ indicates that the modulated vector field has a stronger jerk than in the original skill motion profile.

2) Dishwasher Dataset: Figure 16 shows all the demonstrated skills used for the interaction with the dishwasher. Note that the last state in the demonstration is repeated $N = 20$ times to ensure that the contractive model learns a dynamic that reaches a target state at the end of the trajectory. If the goal is to learn an oscillating motion or a motion without a stable final state, this is not necessary.



(a) Place Plate (b) Place Plate in Rack (c) Place Cup (d) Grasp Doorhandle (e) Open Door (f) Move Sine

Figure 16: Overview of all 6 motion sequences in the dishwasher dataset. Demonstrations are displayed as black solid trajectories, with start and end points represented as red crosses and gray points, respectively.



(a) Motion sequence without obstacle avoidance (b) Obstacle avoidance without plate SDF (c) Obstacle avoidance with plate SDF

Figure 17: EMPTYING THE DISHWASHER task: A series of overlapped snapshots shows the PLACE PLATE IN RACK skill performed by the Franka-Emika Panda robot using an CDF with grasped plate in the right panel and without the grasped plate in the middle panel. The left plot shows the execution of the skill without obstacles.

3) Robustness Test: In the following, the contraction behavior and robustness of the *Move Sine* skill is tested by introducing a manual perturbation during skill execution. The result is shown in Fig. 15. The disturbances are observed with a velocity spike (Fig. 15b), a position deflection (Fig. 15a) and a slight orientation offset (Fig. 15b). The successful stabilized disturbance in Fig. 15a shows the robustness and contraction stability of the learned model.

4) Real World Skill Execution: Fig. 17a shows the robot successfully executing the *Place Plate in Rack* skill.

5) Real World Obstacle Avoidance: Further experiments in Fig. 17b and Fig. 17c demonstrate the need to extend the robot's SDF with the SDF of the grasped object.

6) Real World Inference time: Table VI shows the inference times of ODE solvers during real-world experiments, which approximately match the complexities $\mathcal{O}$ listed in Table V of the paper's appendix. Clearly, the choice of the SDF is most critical.

Solvers / **SDC**	**RDF**	**CDF**
Convex	20.9	2.4
Euler	41.1	4.5
Runge-Kutta	91.2	9.3

Table VI: Comparison of the inference time t_{step} (in ms) required for a diffeomorphism step ψ for the SDC with 1000 points, across different solvers.