Co-SDF: Learning Signed-Distance-Correlated Neural Fields in Configuration Space Without Distance Supervision

Qisong Zheng[1] Sihui Li[1]

***Abstract*— The configuration space (C-space) of a manipulator provides the foundation for formulating motion planning problems, but its high-dimensional and non-convex geometry makes explicit modeling challenging. Signed distance fields (SDFs) are widely used in the workspace to enable gradient-based collision avoidance. Extending this notion to C-space is challenging, as obtaining ground-truth C-space distance-to-collision values for high-DoF systems is computationally expensive. We propose Co-SDF, a correlated signed distance neural representation defined directly in C-space. The model learns a continuous scalar field over joint configurations from binary collision labels without requiring explicit supervision of true C-space signed distance-to-collision. Each configuration is mapped to control-point embeddings via forward kinematics, and a 1-Lipschitz neural architecture trained with hinge Kantorovich–Rubinstein (hKR) and Eikonal regularization promotes smooth and distance-consistent behavior. Experiments with a 2-DoF planar arm and a 7-DoF manipulator show that the learned field aligns its zero level set with collision boundaries, preserves correlation with C-space distance-to-collision, and provides stable gradients for trajectory optimization.**

I. INTRODUCTION

The configuration space (C-space) formulation provides a powerful and unifying framework for motion planning [1]. In C-space, the robot's kinematics transform workspace obstacles into corresponding collision regions to support systematic collision checking and path generation. However, this workspace to C-space transformation can significantly increase geometric complexity, so that even geometrically simple obstacles can induce highly complex, nonlinear, and high-dimensional collision regions in the C-space of a high Degrees of Freedom (DoF) manipulator. As a result, it is difficult to model the C-space collision boundaries and distances explicitly.

To overcome this, many approaches leverage signed distance fields (SDFs) defined in the workspace. SDFs provide continuous obstacle distance information and enable efficient gradient-based collision avoidance in trajectory optimization methods [2], [3]. In recent years, there have been works that employ neural representations of SDF in the workspace [4], [5], [6]. These approaches generally require workspace distance supervision. However, extending this notion to C-space remains challenging. Constructing grid-based SDFs in high-dimensional C-spaces is computationally expensive and is subject to resolution limitations. Moreover, learning a C-space distance field through direct supervision is generally intractable, as obtaining accurate ground-truth C-space distance-to-collision for high DoF systems is itself computationally expensive.

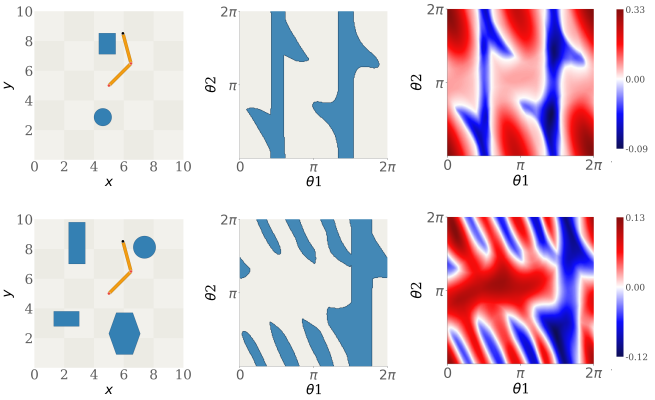


Fig. 1: A demo in two 2-DoF planar manipulator scenes. The left column shows the workspace. The middle column shows the ground-truth C-space. The right column shows the C-space distance field predicted by Co-SDF. The zero level set aligns with collision boundaries, and the predicted value correlates with C-space distance-to-collision.

In this work, we propose a neural signed distance representation of C-space geometry that correlates with C-space distance-to-collision without requiring explicit supervision of true C-space signed distance values. We call it a Correlated Signed Distance Fields in C-pace (Co-SDF). We achieve this by extending the 1-Lipschitz neural distance field framework [7], [8] to C-space modeling and designing the loss function that combines the original hinge Kantorovich–Rubinstein (hKR) objective with an additional Eikonal regularization term [9]. Under Lipschitz constraints, it makes sure the function cannot vary arbitrarily fast across the input space, and the learned function with binary collision labels forms a smooth surrogate whose magnitude correlates with proximity to the collision boundary. The hKR objective enforces global 1-Lipschitz continuity and bounds gradient magnitudes, preventing gradient explosion. The Eikonal regularization further encourages unit-norm gradients, mitigating gradient vanishing and promoting distance-consistent behavior. The resulting neural network model provides differentiable distance measures in C-space and can be directly integrated into optimization-based motion planning. Fig. 1 shows modeling results for a 2-DoF planar manipulator with the workspace layout, the ground-truth C-space, and the distance field predicted by Co-SDF. The learned field reconstructs both collision and free regions. The predicted zero-level set aligns with the true collision

[1]The authors are with the Department of Computer Science, Washington State University, 2710 Crimson Way, Richland, WA 99354, USA. Email: {qisong.zheng, sihui.li}@wsu.edu

boundary, and the output value correlates with distance-to-collision. Experiments with 7-DoF manipulators show high accuracy in collision checking, improved optimization stability, higher success rates, and smoother trajectories in cluttered environments.

II. RELATED WORK

A. Configuration-Space Modeling

Classical motion planning formulates the problem in C-space [1], [10]. The C-space is modeled as a manifold, such as $SE(2)$, $SE(3)$ for 2D or 3D free body, or a high-dimensional torus for articulated robots. In some cases, C_{obs} can be constructed using Minkowski operations. More generally, it can be represented as a semi-algebraic set [10]. For closed-chain systems, kinematic constraints correspond to polynomial equations. This formulation is mathematically precise. Although a precise mathematical characterization exists in theory, explicitly constructing and analyzing high-dimensional C-space geometry is computationally intractable in practice. Because exact representations are difficult to model, sampling-based planners [11], [12] rely on frequent collision checking during planning [13], [14]. Since collision checking are expansive, several works reduce this cost by learning proxy collision models [15], [16] to approximate collision checking in C-space. Specifically, DiffCo [16] outputs collision scores that correlate with workspace distance and provides gradients for optimization. In contrast, our neural representation produces a C-space distance correlated output by leveraging a 1-Lipschitz network architecture along with a specialized loss function.

B. Signed Distance Field Representations

SDFs encode signed geometric distance to surfaces. As a result, they are widely used for implicit surface modeling, shape reconstruction, and differentiable rendering in computer graphics [17], [18]. Beyond computer graphics, SDF-based representations have been adopted in robotics for mapping [19], [20], grasping [21], [22], and motion planning [2], [23]. Classical methods [2], [23] rely on memory-intensive grid-based workspace SDFs of environmental obstacles. Recent approaches replace discrete grids with learned neural SDFs in the workspace [4], [5], [6]. In these methods, joint configurations are mapped to the workspace via forward kinematics, and distance queries are performed in the workspace. Other work defines distance directly in C-space, for example, as the angular distance to contact manifolds [24], [25]. These methods explicitly regress C-space distance with ground-truth C-space distance supervision.

In contrast, we learn a Lipschitz-constrained neural distance field in C-space without explicit distance supervision. The model defines a continuous function over C-space that represents the signed distance to collision boundaries. Experimental results show that the learned field is strongly correlated with the C-space signed distance and provides smooth gradients for optimization-based motion planning.

III. METHOD

A. Problem Formulation and Overview

Let $\mathcal{C} \subset \mathbb{R}^d$ denote the C-space of a manipulator with d degrees of freedom. Each configuration $\mathbf{q} \in \mathcal{C}$ is assigned a binary label through robot–environment collision checking, which defines the free space $\mathcal{C}_{\text{free}}$ and the obstacle region $\mathcal{C}_{\text{obs}}$. The goal is to model this C-space: learn a neural C-space distance-correlated function $d(\mathbf{q}) : \mathcal{C} \to \mathbb{R}$, that implicitly represents collision boundaries with the model's zero-level set and provides smooth, approximate distance-to-collision information suitable for optimization.

The overall framework is illustrated in Fig. 2. Configurations are sampled and labeled in C-space using a collision checker (Sec. III-B). For each joint configuration q, forward kinematics maps q to the workspace positions of a set of predefined control points along the robot links. The concatenated control point coordinates serve as a geometric embedding for learning. This embedding makes collision-relevant geometry explicit, which eases learning compared to using raw joint angles alone (Sec. III-C). Then, we train a 1-Lipschitz neural network using the hKR loss and the Eikonal loss under binary collision supervision. Under the Lipschitz constraint, the hKR objective encourages the network output to vary linearly across the collision boundary, which has been shown to approximate the signed distance function up to a scaling factor [8]. As a result, the learned field exhibits distance-correlated behavior around the boundary (Sec. III-D and Sec. III-E).

B. Boundary-Densified Sampling in C-space

Learning a structured field in high-dimensional C-space requires coverage of the entire domain and sufficient resolution near the collision boundary. In a 7-DoF space, uniform sampling alone requires a high number of samples to capture the narrow transition regions between free and collision states. Therefore, we adopt a mixture sampling strategy. Configurations are partly sampled from a weighted distribution over joint limits and partly from a Gaussian perturbation process [26] to increase density near boundary regions. For Gaussian sampling, given a base configuration $\mathbf{q}_1$, we generate $\mathbf{q}_2 = \mathbf{q}_1 + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$. If $\mathbf{q}_1$ and $\mathbf{q}_2$ have different collision labels, the segment connecting them crosses the boundary. Such segments are retained, and midpoint evaluation is performed to obtain samples closer to the transition on both sides.

To account for local metric distortion induced by the forward-kinematics embedding Sec. III-C, we use the Jacobian volume term $\sqrt{\det(J^\top J)}$ for weighted sampling. This compensates for local metric distortion induced by the forward-kinematics embedding and promotes a geometry-consistent training distribution.

After sampling, for each configuration, we perform geometric-based collision checking to get binary labels $y(\mathbf{q}) \in \{-1, +1\}$, where -1 denotes a collision configuration and +1 denotes a collision-free configuration. No signed distance supervision in C-space or workspace is required.

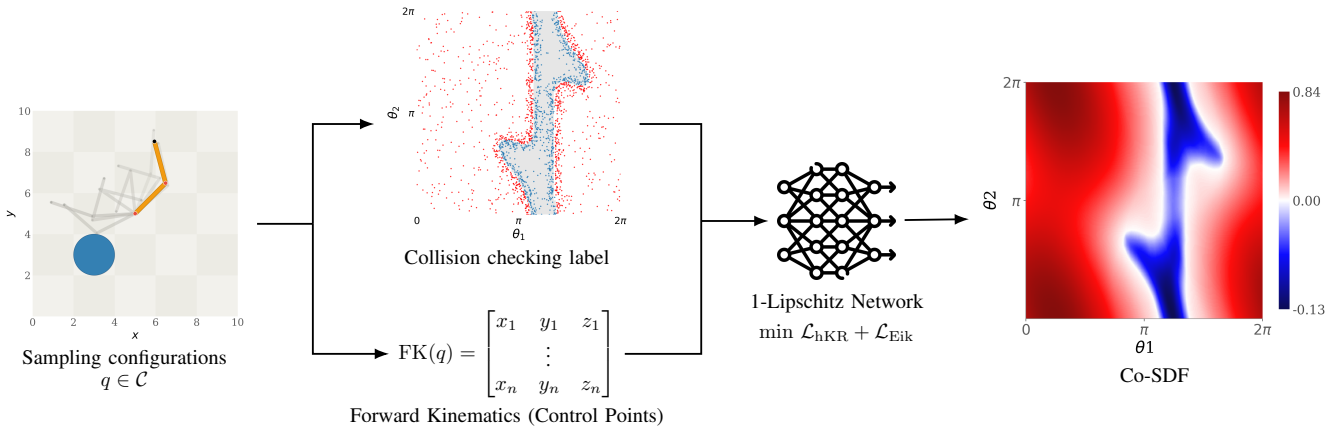


Fig. 2: Overview of the proposed method. Robot configurations are sampled in the C-space and collision states are determined by collision checking. Each configuration is mapped to workspace control points along the robot's body through forward kinematics. A 1-Lipschitz neural network is trained with hKR and Eikonal losses. The network learns a smooth field with the zero level set describing collision boundaries in C-space and correlated collision distance.

C. Geometric Embedding

From the sampling in Sec. III-B, each configuration $\mathbf{q} \in \mathcal{C}$ is associated with a binary collision label. Instead of directly feeding raw joint configurations into the neural network, we apply a deterministic geometric embedding based on forward kinematics. For each configuration q, predefined control points' positions along the robot links are computed and concatenated as the network input (see Fig. 2). A similar idea is also applied in [16].

Each configuration $\mathbf{q}$ is mapped to a set of workspace control points, $\phi(\mathbf{q}) : \mathcal{C} \to \mathbb{R}^{k\times 3}$, where $\phi(\mathbf{q})$ concatenates the Cartesian coordinates of control points placed at the geometric centers of the robot links, and k is the number of control points. The control point locations are fixed in each link frame and transformed to workspace coordinates via forward kinematics. In the 7-DoF Franka setting, two additional control points are added on the gripper to improve geometric coverage. The resulting embedding dimension is $k \times 3$. By incorporating FK embeddings, the network is relieved of the burden of implicitly learning forward kinematics, as this geometric relationship is encoded explicitly within the representation itself.

We then learn an implicit function, $g : \mathbb{R}^k \to \mathbb{R}$, and define the induced C-space field as $d(\mathbf{q}) := g(\phi(\mathbf{q}))$. The zero level set of d approximates the collision boundary in $\mathcal{C}$.

D. 1-Lipschitz Network Construction

In this section, we discuss how we model the implicit function $g(\phi)$. To model a distance-consistent structure, we constrain the neural network to be 1-Lipschitz. For any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^k$, $|g(\mathbf{b}) - g(\mathbf{a})| \leq \|\mathbf{b} - \mathbf{a}\|$. When the function is differentiable almost everywhere, this leads to the pointwise constraint $\|\nabla g(\mathbf{x})\| \leq 1$.

To enforce this constraint at the architectural level, we adopt the Semi-definite Programming Lipschitz Layer (SLL) [7]. Each residual layer is defined as $\mathbf{x} \mapsto \mathbf{x} - 2WT^{-1}\sigma(W^\top \mathbf{x} + \mathbf{b})$, where $\sigma(\cdot)$ denotes the ReLU activation function. The matrix $W \in \mathbb{R}^{k\times m}$ and the vector $\mathbf{b} \in \mathbb{R}^m$ are learnable parameters. The matrix $T \in \mathbb{R}^{k\times k}$ is a diagonal matrix constructed from W. This construction ensures that each residual layer satisfies the 1-Lipschitz condition. Continuous distance-field representations have been shown to be well suited for gradient-based motion planning and collision avoidance [4], [24]. Enforcing the 1-Lipschitz constraint guarantees bounded gradients and global continuity, limiting abrupt variations in C-space and promoting stable gradient-based optimization. More importantly, it preserves the geometric interpretation of the learned field as a distance-like function rather than an arbitrary classification score (refer to [8] for more details).

At the output, a normalized affine mapping is applied,

$$\mathbf{x} \mapsto \frac{\mathbf{w}^\top \mathbf{x}}{\|\mathbf{w}\|_2} + b, \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^k$ and $b \in \mathbb{R}$ are learnable parameters. We divide by $\|\mathbf{w}\|_2$ to control the scale of the linear mapping. This step ensures that the affine mapping satisfies the 1-Lipschitz condition.

E. Training Objective with hKR and Eikonal Regularization

Under the 1-Lipschitz architectural constraint in Sec. III-D, we train the implicit field using binary collision labels with a combined hKR loss and Eikonal loss.

The hKR loss consists of a Kantorovich-Rubinstein (KR) term and a hinge term [8]:

$$\mathcal{L}_{\text{hKR}} = \mathcal{L}_{\text{KR}} + \lambda \mathcal{L}^m_{\text{hinge}}. \tag{2}$$

We introduce a margin parameter $m > 0$ that controls the enforced functional separation between collision and free regions, and a weighting coefficient $\lambda > 0$ that balances the hinge term in the hKR objective. In all experiments, we set $m = 10^{-2}$ and manually schedule λ during training, progressively increasing it from 1 to 100 to stabilize optimization. By the Kantorovich–Rubinstein duality theorem [27], the Wasserstein-1 distance [28] admits a dual formulation over 1-Lipschitz functions. In the binary collision setting, the KR

term can be written as

$$\mathcal{L}_{\mathrm{KR}} = \int_{\mathcal{C}} - y(\mathbf{q})\, d(\mathbf{q})\, \rho(\mathbf{q}) d\mathbf{q}, \tag{3}$$

where y is the collision label, $\rho(\mathbf{q})$ denotes the empirical sampling distribution induced by mini-batch training. All loss terms are computed as expectations with respect to ρ. which encourages maximal separation between the two label distributions under the 1-Lipschitz constraint.

However, optimizing the KR objective alone is insufficient [29]. Therefore, a hinge term is added to the loss [8].

$$\mathcal{L}^{m}_{\mathrm{hinge}} = \int_{\mathcal{C}} \max(0,\ m - y(\mathbf{q})\, d(\mathbf{q}))\ \rho(\mathbf{q}) d\mathbf{q}, \tag{4}$$

The margin parameter m showed in (2) controls the functional separation between classes. Together, they determine how supervision is distributed in C-space, influencing boundary alignment and geometric consistency. The hinge term enforces a functional margin around the zero level set, pushing samples away from the decision boundary and improving boundary alignment.

To further encourage distance-consistent behavior, we include an Eikonal regularization term [9]:

$$\mathcal{L}_{\mathrm{Eik}} = \int_{\mathcal{C}} \left(\|\nabla d(\mathbf{q})\|_2 - 1\right)^2 \rho(\mathbf{q}) d\mathbf{q}. \tag{5}$$

While the 1-Lipschitz constraint enforces the upper bound $\|\nabla d(q)\| \leq 1$, it does not prevent vanishing gradients. The Eikonal regularization instead encourages $\|\nabla d(q)\| \approx 1$, thereby promoting distance-consistent behavior and more uniform level-set spacing in C-space.

The total loss is defined as $\mathcal{L}_{total} = \alpha_1 \mathcal{L}_{\mathrm{hKR}} + \alpha_2\, \mathcal{L}_{\mathrm{Eik}}$, We set α_1 = 1.0, α_2 = 0.1. Under the 1-Lipschitz architecture, the hKR objective enforces structured separation between free space and obstacle regions, and the Eikonal regularizer promotes distance-consistent gradients. Their combination enables learning a smooth C-space field directly from binary collision labels. Our approach does not require ground-truth signed distance supervision. This enables learning a distance-consistent representation in C-space even when exact geometric distance information is unavailable or computationally intractable in high-dimensional settings. Sec. IV-A.1 shows the advantages of our model design when comparing with others that do not use a 1-Lipschitz architecture.

IV. EXPERIMENT

Our model is trained in PyTorch on an NVIDIA PNY RTX 5080 GPU (16GB GDDR7). All experiments are conducted on an 8-Core AMD Ryzen 7 9700 Processor. We evaluate our model on a 2-DoF articulated manipulator as a demonstrative example and on a 7-DoF Franka Research 3 [30] manipulator to validate its effectiveness in high-dimensional C-space. 7-DoF simulation and ground-truth collision checking are implemented in MuJoCo [31].

We use 5,000 training configurations for the 2-DoF planar setting and 500,000 training configurations for the 7-DoF Franka setting. For the 2-DoF setting, Co-SDF is parameterized as a 10-layer 1-Lipschitz network with hidden dimension 256, trained for 200 epochs using Adam with learning rate 5×10^{-4} and batch size 200, requiring approximately 30 seconds of training. For the 7-DoF setting, we use a 20-layer 1-Lipschitz network with hidden dimension 512, trained with the same optimization hyperparameters, with a training time of approximately 1.5 hours.

In the following, we evaluate the proposed Co-SDF representation from four complementary perspectives. First, we assess whether Co-SDF captures a meaningful geometric structure in C-space by analyzing its collision checking accuracy and its distance-like behavior. Second, we examine the ability of Co-SDF to guide configurations out of collision regions using both gradient-based and sampling-based strategies. Finally, we demonstrate how Co-SDF can be integrated as a differentiable feasibility constraint in trajectory optimization for high-DoF robotic manipulators.

A. Evaluation of the Learned C-Space Distance Field

This section evaluates whether the proposed Co-SDF captures meaningful C-space structure. We conduct the evaluation in two settings: (i) a low-dimensional 2-DoF setting to compare with two other neural models to show Co-SDF's C-space distance correlation advantages. (ii) a 7-DoF Franka FR3 setting to assess the scalability and effectiveness of the model in high dimensions.

1) 2-DoF Planar Manipulator: Distance Correlation Comparison and Analysis: We evaluate Co-SDF on a 2-DoF planar manipulator as shown in Fig. 1. To show the effectiveness of our network architecture and loss function design, we compare with two other neural models, MLP [17] and SIREN [32]. Both baselines follow their standard implicit field formulations, which do not have 1-Lipschitz constraints, and are trained with MSE on binary occupancy labels together with Eikonal regularization. All models are trained under the same binary labels, and no ground-truth distance values are used during training.

We sample $N = 5{,}000$ configurations for testing, and obtain ground-truth distances by constructing a grid-based C-space SDF computed on a 256×256 joint-angle grid. All distances are measured in radians. We use Spearman correlation to evaluate whether Co-SDF preserves the relative ranking of configurations, and Pearson correlation to assess the degree of linear agreement as well as scale and bias alignment with the ground truth.

Table I reports correlation and classification metrics on the testing configurations. All models achieve a similar high classification accuracy. In contrast, the Spearman and Pearson correlations reveal a clear difference. Our model's predictions are highly rank-correlated with the ground-truth grid-based C-space signed distance-to-collision and maintain substantial linear agreement. MLP and SIREN show significantly lower rank consistency and linear agreement. These results indicate that all models can separate valid and invalid configurations, but only our model preserves a strong correlation with C-space collision distance. In general, the gap between the

TABLE I: Correlation and classification metrics in the 2-DoF setting with 2 and 4 obstacles. All metrics are computed over $5,000$ randomly sampled testing configurations.

Model	Spearman	Pearson	F1	Acc%	FNR%	FPR%
2 obstacles						
Co-SDF	0.941	0.903	0.986	98.06	1.62	2.75
MLP	0.567	0.772	0.989	98.38	0.95	3.31
SIREN	0.663	0.775	0.992	98.82	0.45	3.03
4 obstacles						
Co-SDF	0.964	0.924	0.992	98.94	0.75	1.67
MLP	0.816	0.755	0.989	98.58	0.84	2.56
SIREN	0.666	0.739	0.995	99.34	0.57	0.83

TABLE II: Correlation and classification metrics in the 7-DoF Franka FR3 setting with 10 and 20 obstacles. All metrics are computed over $5{,}000$ testing configurations.

Obstacles	Spearman	Pearson	F1	Acc%	FNR%	FPR%
10	0.948	0.854	0.950	96.74	4.55	2.65
20	0.956	0.879	0.946	95.70	4.91	3.90

accuracy of the classification and the correlation of distance highlights the difference between the feasibility of learning and the geometry of learning. Although all methods are trained only with binary supervision, our model recovers a distance-like geometric structure in C-space.

2) 7-DoF Franka FR3 Manipulator: Distance Correlation and Accuracy in High-Dimension: We randomly sample $5{,}000$ configurations for testing. We use a random sampling strategy to obtain approximate ground-truth signed distances for correlation analysis. For each configuration q, we first determine its collision state. We then apply local random perturbations in C-space. When a perturbed configuration exhibits a different collision state, the segment between the two configurations must cross the collision boundary. We perform binary search along this segment to locate the boundary point and compute the ℓ_2 distance between q and the boundary point. We then select the closest point on the collision boundary to the original configuration among all detected crossings. The distance is positive in free space and negative in collision, measured in radians. We use this distance as the approximated ground truth.

We evaluate the trained Co-SDF on the same configurations and compute the Spearman and Pearson correlation between the Co-SDF outputs and the ground truth. The results shown in Table II indicate a strong positive correlation in the 7D C-space, demonstrating that the learned field preserves the rank ordering induced by distance to the collision boundary in high dimension.

B. Local Escape from In-Collision Configurations

This experiment studies a local escape task from in-collision configurations in C-space. The goal is to evaluate whether the learned fields provide meaningful and reliable local gradient guidance. Given an initial configuration q in collision, the objective is to reach a nearby collision-free configuration. The learned field $d(q)$ is defined such that zero corresponds to the collision boundary and positive values indicate collision-free configurations. An escape search is considered successful and is terminated if the final configuration is verified to be collision-free by ground-truth collision checking within $T_{\max} = 200$ steps. We set experiments with 2, 4, 6 obstacles in 2-DoF experiment and 10, 20, 30 obstacles in the 7-DoF setting. We employ two local escape strategies and use them across all models.

a) Gradient-Based Escape: We formulate the escape process as a local optimization problem by minimizing $C(q) = \mathrm{ReLU}(-d(q))$. We update the configuration using Adam [33] with learning rate $\alpha = 0.01$.

b) Sampling-Based Escape: As a non-differentiable baseline, we generate $K = 200$ locally perturbed candidates at each step with step size $\sigma = 0.01$. We select the candidate with the largest value predicted by $d(q)$ in each step.

We test $1,000$ samples in each scene. We evaluate escape performance using the success rate (whether escaped at maximum step), the number of escape steps, and the computation time for all samples. We also report the average C-space distance between escaped configuration and sampled collision configuration, and the mean absolute error (MAE) of this distance with ground-truth (based on grid-based C-space SDF in 2D and random sampling strategy in 7D, same as in Sec. IV-A.1). These results are computed only for successful escapes.

As shown in Table III, in the 2-DoF setting, we compare Co-SDF with the implicit fields based on standard MLP [17] and SIREN [32]. Co-SDF achieves consistently higher ground-truth success rates and requires fewer escape steps. Co-SDF also has a lower escape time in most cases. These results show that with the 1-Lipschitz constraint and the hKR loss, the learned field preserves stable gradient magnitudes in C-space, enabling reliable ascent directions toward free space during optimization.

When comparing C-space distance and MAE, Co-SDF's average C-space distance is the largest among successful escapes, meaning the valid gradient is available in the entire C-space. Co-SDF also has a smaller MAE when considering the average distance. In contrast, the low average distance and MAE of SIREN indicate that it is only valid near the collision boundary. When configurations lie deeply inside obstacles, its gradients become less informative, and escape becomes difficult. In the 2 obstacles scene, MLP tends to overfit the collision boundary, resulting in a low escape success rate. In the 4 obstacles and 6 obstacles scenes, MLP has a considerably large MAE.

In the 7-DoF setting (See Table IV), escapes with Co-SDF also maintain a similar high success rate and relatively low MAE. In both the 2-DoF and 7-DoF settings, Co-SDF provides reliable local guidance for recovering from in-collision configurations. These results suggest that Co-SDF is suited as a reactive safety primitive for local collision recovery in high-dimensional C-spaces and can also provide useful estimates of penetration depth.

TABLE III: ESCAPE benchmark results ($N = 1,000$) in the 2-DoF setting with 2, 4, and 6 obstacles.

	2 obstacles					4 obstacles					6 obstacles				
Method	Success(%)	Steps	MAE	Dist(rad)	Time(ms)	Success(%)	Steps	MAE	Dist(rad)	Time(ms)	Success(%)	Steps	MAE	Dist(rad)	Time(ms)
Co-SDF Grad	100.0	20.8	0.0590	0.256	84.83	97.9	27.8	0.0553	0.277	115.17	100.0	23.0	0.0512	0.270	94.31
Co-SDF Samp	100.0	9.6	0.0344	0.230	18.71	100.0	10.7	0.0360	0.256	21.14	100.0	10.5	0.0311	0.247	20.92
SIREN Grad	20.4	161.2	0.0240	0.068	163.17	18.5	164.3	0.0156	0.073	166.11	23.8	154.8	0.0184	0.086	158.84
SIREN Samp	22.4	156.2	0.0232	0.072	73.22	20.9	159.0	0.0179	0.079	75.45	24.8	151.4	0.0183	0.085	73.28
MLP Grad	23.7	159.4	0.0574	0.147	167.98	77.8	74.6	0.1136	0.324	77.95	68.2	87.9	0.0918	0.281	93.16
MLP Samp	28.7	145.9	0.0454	0.130	71.34	79.2	53.7	0.1013	0.314	26.17	68.4	72.4	0.0712	0.257	36.06

TABLE IV: ESCAPE benchmark results ($N = 1,000$) in the 7-DoF setting with 10, 20, and 30 obstacles.

	obstacle 10					obstacle 20					obstacle 30				
Method	Success(%)	Steps	MAE	Dist(rad)	Time(ms)	Success(%)	Steps	MAE	Dist(rad)	Time(ms)	Success(%)	Steps	MAE	Dist(rad)	Time(ms)
Gradient	99.7	15.2	0.059	0.251	158.54	97.0	25.4	0.065	0.270	269.37	98.3	22.6	0.072	0.270	234.70
Sampling	100.0	8.1	0.064	0.195	61.43	99.9	9.6	0.073	0.216	74.24	99.9	9.3	0.072	0.212	68.73

C. Trajectory Optimization with Co-SDF

We experiment with a 7-DoF Franka Research 3 (FR3) manipulator operating in randomized cluttered environment scenes and a bookshelf scene (Fig. 3) to evaluate its performance in real planning problems. In the cluttered environment, we randomly select a collision-free start and the goal configuration. In the bookshelf scene, the robot is required to move between different shelf levels through narrow openings. This scene introduces confined spaces and highly nonconvex geometry, making trajectory feasibility and solver stability more challenging than in randomly generated obstacle scenes.

1) Optimization Problem Setup: We evaluate Co-SDF in a trajectory optimization problem formulation. A trajectory is parameterized as a sequence of n discrete joint-space waypoints, $Q = \{q_0, q_1, \ldots, q_{n-1}\}$, $q_0 = q_{\text{start}}$, $q_{n-1} = q_{\text{goal}}$, where each configuration $q \in \mathbb{R}^7$ lies in the joint space of the robot.

For each planning query, we first use a sampling-based planner (RRT/RRT-Connect) [11], [34] in OMPL [35] to get the initial trajectory $Q^{(0)}$ for optimization. The resulting path provides connectivity but is not optimized for smoothness and may contain collisions.

Given the initial trajectory $Q^{(0)}$, we optimize the waypoint sequence $Q = \{q_0, \ldots, q_{n-1}\}$ by solving the following nonlinear program:

$$\begin{aligned} \min_{Q} \;& C_{\text{smooth}}(Q) + C_{\text{clr}}(Q) \\ \text{s.t.} \;& q_0 = q_{\text{start}},\; q_{n-1} = q_{\text{goal}}, \\ & q_{\min} \le q_i \le q_{\max}, \\ & d(q_i) \ge \epsilon, \\ & -v_{\max}\Delta t \le q_i - q_{i-1} \le v_{\max}\Delta t, \\ & -a_{\max}\Delta t^2 \le q_{i+1} - 2q_i + q_{i-1} \le a_{\max}\Delta t^2. \end{aligned} \tag{6}$$

In all experiments, we use $n = 30$ waypoints with $\Delta t = 0.1$, set $v_{\max} = 1.5$ and $a_{\max} = 3.0$, choose safety margin $\epsilon = 0.02$. Starting from $Q^{(0)}$, we solve (6) using IPOPT [36] and SLSQP [37]. For IPOPT, the optimization problem is formulated in CasADi [38], and Co-SDF $d(\cdot)$ is wrapped via L4CasADi [39] as a differentiable CasADi function. For SLSQP, the same objective and constraints are evaluated through NLopt [40], with gradients computed using PyTorch autograd. We limit the maximum number of solver iterations to 150.

The smoothness cost is defined as

$$\begin{aligned} C_{\text{smooth}}(Q) = \frac{1}{n}\Bigg(& \sum_{i=1}^{n-1} \|q_i - q_{i-1}\|_2^2 \\ & + \lambda \sum_{i=1}^{n-2} \|q_{i+1} - 2q_i + q_{i-1}\|_2^2 \Bigg), \end{aligned} \tag{7}$$

with $\lambda = 20$.

The clearance term is

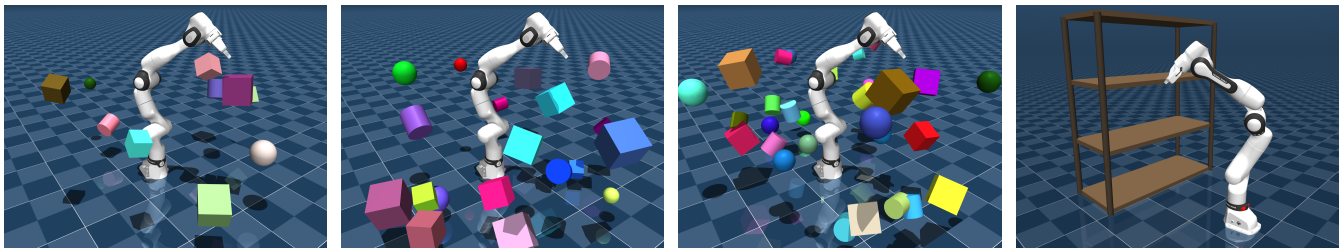
$$C_{\text{clr}}(Q) = w_{\text{clr}} \sum_{i=0}^{n-1} \exp\left(\min\left(50,\; -\frac{d(q_i)}{\epsilon}\right)\right), \tag{8}$$

with the weight of clearance $w_{\text{clr}} = 1.0$.

All optimized trajectories are validated by high-resolution ground-truth collision checks. Trajectories are interpolated and uniformly resampled to 512 configurations for evaluation. All metrics, including collision checking for success rate, path length, and smoothness (normalized second- order finite difference, with the mean Sm_{mean} and the 95th percentile Sm_{p95}), are computed at this fixed resolution. A trajectory is regarded as successful only if all interpolated configurations are collision-free.

2) Results: We evaluated each method in 100 planning queries in random obstacle scenes with 10, 20 or 30 obstacles and the structured bookshelf scene. The same initialization and waypoint count are used for all methods. We compare Co-SDF-based trajectory optimization with a workspace-based neural SDF method [17], [5] and DiffCo [16] under identical objectives, constraints, and solver settings. Fig. 4 shows an example optimized path with Co-SDF from the initial trajectory.

The results are reported in Table V. Co-SDF achieves better results across all metrics. With IPOPT, Co-SDF achieves higher success rates in 10 and 20 obstacles scenes, and produces shorter trajectories with better smoothness values and lower average solve time. When using SLSQP, success rates remain comparable, while solve times increase and smoothness values are higher compared to IPOPT. In more complicated scenes, such as the 30 obstacles scene and the structured bookshelf scene, NN-SDF and DiffCo shows unstable performance. The constrained geometry makes trajectory optimization highly sensitive to gradient quality, especially in more complicated scenes. The limited resampling

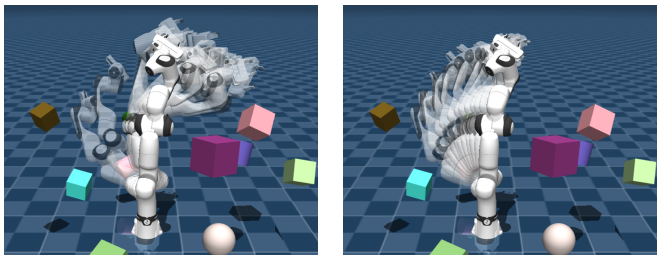


(a) 10 obstacles (b) 20 obstacles (c) 30 obstacles (d) bookshelf

Fig. 3: Example cluttered 10, 20, 30 obstacles scene and a bookshelf scene used in the trajectory optimization.

TABLE V: 7-DoF Franka FR3 trajectory optimization comparison in cluttered obstacle scenes with 10/20/30 obstacles and a structured bookshelf scene.

	obstacle 10					obstacle 20				
Method	Success Rate (%)	Len	Sm_{mean}	Sm_{p95}	Time (s)	Success Rate (%)	Len	Sm_{mean}	Sm_{p95}	Time (s)
Co-SDF-ipopt	**97.0**	5.775	12.82	108.69	1.59	**97.0**	5.853	15.69	120.72	2.46
Co-SDF-slsqp	**97.0**	6.730	36.31	362.57	11.39	92.0	6.831	39.61	415.92	12.04
DiffCo-ipopt	75.0	**5.552**	**9.54**	**69.93**	**0.72**	69.0	**5.559**	**11.12**	**84.16**	**1.00**
DiffCo-slsqp	73.0	6.500	36.60	296.74	5.08	67.0	6.864	62.64	440.13	4.74
NN-SDF-ipopt	58.0	6.534	30.48	314.19	20.46	51.0	6.197	24.60	206.01	38.40
NN-SDF-slsqp	79.0	7.951	126.57	768.09	15.97	72.0	7.674	109.03	707.47	18.96
	obstacle 30					bookshelf				
Method	Success Rate (%)	Len	Sm_{mean}	Sm_{p95}	Time (s)	Success Rate (%)	Len	Sm_{mean}	Sm_{p95}	Time (s)
Co-SDF-ipopt	86.0	5.873	14.94	114.70	2.59	74.0	6.141	18.43	146.08	3.79
Co-SDF-slsqp	**88.0**	6.888	39.24	377.10	17.02	**86.0**	7.885	45.17	414.46	29.06
DiffCo-ipopt	60.0	**5.770**	**12.34**	**87.84**	**2.15**	17.0	**5.892**	**15.16**	**119.48**	**0.63**
DiffCo-slsqp	57.0	7.372	79.93	618.81	6.16	20.0	8.630	122.04	494.20	8.51
NN-SDF-ipopt	49.0	6.316	29.48	292.79	15.38	2.0	6.609	25.40	200.56	45.22
NN-SDF-slsqp	63.0	8.508	150.07	1035.31	12.46	21.0	9.905	147.12	636.08	27.64



(a) Initial (b) Co-SDF

Fig. 4: Trajectory optimization results in a 7-DoF random obstacle setting. Co-SDF produces a smooth trajectory from the initial trajectory.

budget further amplifies this sensitivity. The workspace-based method computes distances in task space through forward kinematics during optimization, whereas Co-SDF defines a scalar distance function directly in the C-space. DiffCo learns collision scores that correlate with workspace distance. This difference in representation influences the numerical behavior of the solver in cluttered scenes. Results suggest that Co-SDF provides a more stable and geometrically consistent collision representation in C-space.

We also experimented with using Co-SDF in sampling-based planning as a replacement for the collision checker. However, due to approximation errors near the decision boundary and the strict accuracy requirements of sampling-based planners along the entire path, small boundary deviations can significantly affect success rates. In contrast, trajectory optimization benefits from continuous gradient information and can iteratively correct local collisions. These observations highlight that Co-SDF is particularly well-suited for gradient-based motion optimization rather than discrete feasibility checking. Also, a single collision check takes about 4.0 ms with FK embedding when evaluated without batching for sampling-based planning. With batch checking, the speed will be faster according to the level of parallelization on GPU.

V. CONCLUSION AND FUTURE WORK

This paper studies distance modeling directly in C-space for motion planning and proposes Co-SDF, a 1-Lipschitz neural field trained using only binary collision labels. The model incorporates boundary-densified sampling and a forward-kinematics control-point embedding to capture geometric structure and is optimized using hKR and Eikonal objectives.

Experiments validate the method from three complementary perspectives. First, in both 2-DoF and 7-DoF settings, Co-SDF recovers a distance-consistent geometric structure from binary supervision and preserves the rank ordering induced by proximity to the collision boundary. Second, in local escape tasks, Co-SDF provides more stable gradients near collision regions, improving recovery success and reducing joint displacement. Finally, in 7-DoF trajectory optimization, Co-SDF improves the success rate and produces shorter and smoother trajectories under identical initialization and solver

settings. The advantage becomes more evident in structured narrow-passage environments, where optimization is highly sensitive to gradient quality.

This method is scene-specific. Future work will investigate scene generalization in dynamic environments, aiming to learn C-space distance representations that adapt to varying obstacle layouts without retraining. An important extension is to explicitly incorporate self-collision modeling within the learned C-space field. We also plan to deploy the proposed framework on real robotic platforms for hardware validation.

REFERENCES

[1] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE transactions on computers*, vol. 32, no. 02, pp. 108–120, 1983.

[2] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 489–494.

[3] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[4] Y. Li, Y. Zhang, A. Razmjoo, and S. Calinon, "Representing robot geometry as distance fields: Applications to whole-body manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 15 351–15 357.

[5] B. Liu, G. Jiang, F. Zhao, and X. Mei, "Collision-free motion generation based on stochastic optimization and composite signed distance field networks of articulated robot," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7082–7089, 2023.

[6] M. Koptev, N. Figueroa, and A. Billard, "Neural joint space implicit signed distance functions for reactive robot manipulator control," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 480–487, 2022.

[7] A. Araujo, A. Havens, B. Delattre, A. Allauzen, and B. Hu, "A unified algebraic perspective on lipschitz neural networks," *arXiv preprint arXiv:2303.03169*, 2023.

[8] G. Coiffier and L. Béthune, "1-lipschitz neural distance fields," in *Computer Graphics Forum*, vol. 43, no. 5. Wiley Online Library, 2024, p. e15128.

[9] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," *arXiv preprint arXiv:2002.10099*, 2020.

[10] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[11] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[12] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[13] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 2002.

[14] J. Pan, S. Chitta, and D. Manocha, "Fcl: A general purpose library for collision and proximity queries," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 3859–3866.

[15] N. Das and M. Yip, "Learning-based proxy collision detection for robot motion planning applications," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1096–1114, 2020.

[16] Y. Zhi, N. Das, and M. Yip, "Diffco: Autodifferentiable proxy collision detection with multiclass labels for safety-aware trajectory optimization," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 2668–2685, 2022.

[17] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.

[18] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans, "Reconstruction and representation of 3d objects with radial basis functions," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 67–76.

[19] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam, "isdf: Real-time neural signed distance fields for robot perception," *arXiv preprint arXiv:2204.02296*, 2022.

[20] L. Wiesmann, T. Guadagnino, I. Vizzo, N. Zimmerman, Y. Pan, H. Kuang, J. Behley, and C. Stachniss, "Locndf: Neural distance field mapping for robot localization," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4999–5006, 2023.

[21] T. Weng, D. Held, F. Meier, and M. Mukadam, "Neural grasp distance fields for robot manipulation," *arXiv preprint arXiv:2211.02647*, 2022.

[22] W. Xu, W. Guo, X. Shi, X. Sheng, and X. Zhu, "Fast force-closure grasp synthesis with learning-based sampling," *IEEE Robotics and Automation Letters*, vol. 8, no. 7, pp. 4275–4282, 2023.

[23] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization." in *Robotics: science and systems*, vol. 9, no. 1. Berlin, Germany, 2013, pp. 1–10.

[24] Y. Li, X. Chi, A. Razmjoo, and S. Calinon, "Configuration space distance fields for manipulation planning," *arXiv preprint arXiv:2406.01137*, 2024.

[25] M. Li, Y. Zhou, H. Ying, and F. R. Yu, "Cdflow: Generative gradient flows for configuration space distance fields via neural odes," *arXiv preprint arXiv:2509.13771*, 2025.

[26] V. Boor, M. H. Overmars, and A. F. Van Der Stappen, "The gaussian sampling strategy for probabilistic roadmap planners," in *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1018–1023.

[27] C. Villani *et al.*, *Optimal transport: old and new*. Springer, 2009, vol. 338.

[28] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017. [Online]. Available: https://arxiv.org/abs/1701.07875

[29] M. Serrurier, F. Mamalet, A. González-Sanz, T. Boissin, J.-M. Loubes, and E. Del Barrio, "Achieving robustness in classification using optimal transport with hinge regularization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 505–514.

[30] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jähne, L. Hausperger, and S. Haddadin, "The franka emika robot: A reference platform for robotics research and education," *IEEE Robotics & Automation Magazine*, vol. 29, no. 2, pp. 46–64, 2022.

[31] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

[32] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.

[33] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[34] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[35] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[36] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[37] D. Kraft, "Algorithm 733: Tomp–fortran modules for optimal control calculations," *ACM Transactions on Mathematical Software (TOMS)*, vol. 20, no. 3, pp. 262–281, 1994.

[38] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[39] T. Salzmann, J. Arrizabalaga, J. Andersson, M. Pavone, and M. Ryll, "Learning for casadi: Data-driven models in numerical optimization," in *6th Annual Learning for Dynamics & Control Conference*. PMLR, 2024, pp. 541–553.

[40] S. G. Johnson, "The NLopt nonlinear-optimization package," https://github.com/stevengj/nlopt, 2007.