Appendix for “PairwiseNet: Pairwise Collision Distance Learning for High-dof Robot Systems”

Anonymous Author(s)
Affiliation
Address
`email`

A Further Experimental Details

A.1 Capsule-based Bounding Volume Method for a Panda Robot Arm

We construct capsule-shaped collision primitives for a Panda robot arm as a baseline collision distance estimation method (see Figure 1). Our approach follows a similar methodology to [1], which formulates an optimization problem as follows:

$$\min_{a_i, b_i, r_i} \quad ||a_i - b_i|| \pi r_i^2 + \frac{4}{3}\pi r_i^3 \tag{1}$$

$$\text{s.t.} \quad \text{dist}(p, \overline{a_i b_i}) \leq r_i, \quad \text{for all } p \in \mathcal{M}_i \tag{2}$$

Here, i denotes the link index of the Panda robot arm, $\mathcal{M}_i$ represents the vertices of the i^{th} link mesh, and a_i, b_i, and r_i refer to the two endpoints and the radius of a capsule, respectively, and $\overline{a_i b_i}$ represents the line segment connecting the two endpoints. This formulation results in the creation of minimal volume capsules that encapsulate all vertices of the link meshes. The collision distance of the multi-arm robot systems can be estimated through the minimum distance calculation between capsules.

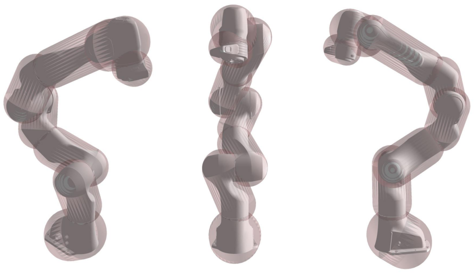

Figure 1: Illustrations of the Panda robot arm with capsule-shape collision primitives.

A.2 The Collision-free Guaranteed Threshold

The collision-free guaranteed threshold ϵ_{safe} refers to a predefined distance value that is established in collision distance estimation methods. This threshold is set to ensure that during testing or actual operation, the estimated collision distance remains above this threshold for all valid configurations or movements of the robot system. In other words, if the estimated collision distance between the robot and any obstacles remains above the collision-free guaranteed threshold ($\hat{d}_{\text{col}}(q) > \epsilon_{\text{safe}}$), it is considered safe and collision-free. In our experiments, we set the collision-free guaranteed threshold to the least conservative value that allows us to classify all collision configurations in the test dataset as collisions. These thresholds are then utilized for measuring the Safe-FPR.

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

Table 1: The Collision-free Guaranteed Thresholds

	Two arms	Three arms	Four arms
Capsule	0.0	0.0	0.0
JointNN	0.2111	0.2015	0.2231
PosNN	0.1141	0.1189	0.1756
jointNERF	0.1661	0.1734	0.2001
ClearanceNet	0.2840	0.3713	0.4944
DiffCo	-1.2789	-1.4672	-0.9535
PairwiseNet	0.0150	0.0152	0.0184

A.3 Hyperparameters

Table 2 shows hyperparameters employed in our experiments.

Table 2: Hyperparameters

hyperparameter	value
batch size, learning rate, epoch for PairwiseNet	1000, 1e-3, 2000
batch size, learning rate, epoch for ClearanceNet	191, 1.75e-4, 400
batch size, learning rate, epoch for other NN baselines	10000, 1e-3, 10000
k for k-nearest neighbor of EdgeConv layers	5
# of points in the point cloud data of an shape element	100
hidden nodes of EdgeConv layers	64

B Additional Experimental Results

B.1 Inference Time of PairwiseNet

PairwiseNet has an efficient inference strategy that infers solely the regressor network without the need to path through the encoder, which contains complex EdgeConv layers. Additionally, the batch calculation capability of the neural network structure further enhances the efficiency of PairwiseNet when estimating the global collision distance, even with a large number of element pairs in the system. Table 3 provides the inference time of PairwiseNet for different environments. Notably, even with an increase in the number of element pairs to 384 in a four-arm system, PairwiseNet maintains a reasonably efficient inference time of 0.542 ms on CPU and 0.347 ms on GPU. These inference times highlight the ability of PairwiseNet to deliver fast collision distance estimation, ensuring practicality and real-time applicability in various robotic environments.

Table 3: Inference time of PairwiseNet

Env.	Two arms	Three arms	Four arms	One arm with obstacles
# of element pairs	64	192	384	145
Inference time (CPU) (ms)	0.201	0.323	0.542	0.331
Inference time (GPU) (ms)	0.235	0.284	0.347	0.257

B.2 Collision-free Path Planning with PairwiseNet

The collision-free trajectory used in our experiments is the outcome of path planning process utilizing the trained PairwiseNet model. Similar to the approaches presented in [2, 3], the trajectory planning process can be formulated as the following optimization problem:

$$\begin{aligned} \min_{\tau} \quad & \int_0^T \dot{\tau}(t)dt & (3) \\ \text{s.t.} \quad & \hat{d}_{\text{col}}(\tau(t)) > \epsilon, \quad t \in [0, T] & (4) \\ & \tau(0) = q_{\text{start}} & (5) \\ & \tau(T) = q_{\text{end}}. & (6) \end{aligned}$$

where τ represents the trajectory in the joint configuration space, $\tau(t) \in \mathbb{R}^{N_{\text{dof}}}$, and q_{start} and q_{end} are the start and end joint configurations, respectively, and ϵ is the collision-free guaranteed threshold of PairwiseNet. Auto-differentiation and the Adam optimizer in PyTorch [4] are employed for the optimization process. To enhance computational efficiency and trajectory smoothness, we utilize cubic spline parameterization for the trajectory function $\tau(t)$. Additionally, the supplementary materials include a video file showcasing another example of path planning with PairwiseNet.

References

[1] A. El Khoury, F. Lamiraux, and M. Taix. Optimal motion planning for humanoid robots. In *2013 IEEE international conference on robotics and automation*, pages 3136–3141. IEEE, 2013.

[2] Y. Zhi, N. Das, and M. Yip. Diffco: Autodifferentiable proxy collision detection with multiclass labels for safety-aware trajectory optimization. *IEEE Transactions on Robotics*, 2022.

[3] Y. Kim, J. Kim, and D. Park. Graphdistnet: A graph-based collision-distance estimator for gradient-based trajectory optimization. *IEEE Robotics and Automation Letters*, 7(4):11118–11125, 2022.

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.