Learning Distance Functions for Robot Obstacle Avoidance

Master Degree in Artificial Intelligence and Robotics

Giuseppina Iannotti 1938436

Advisors: Prof. Giuseppe Oriolo, Dr. Navvab Kashiri

Co-Advisors: Dr. Andrea Monguzzi, Dr. Giuseppe Alfonso

Academic Year 2024/25

Table Of Contents

Problem Context

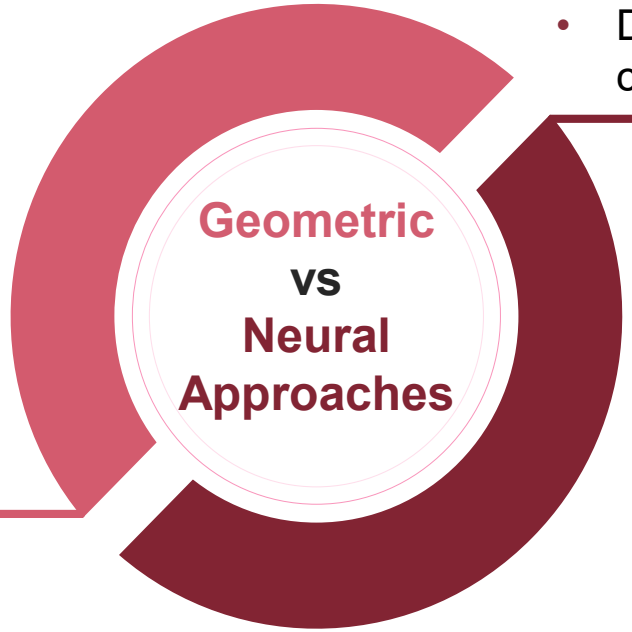
Robots must move safely in cluttered environments, reasoning about their proximity to surrounding objects. **Obstacle avoidance** ensures clearance is maintained, preventing both self-collisions and external contacts.

Geometric methods provide accurate distances but are computationally **expensive** and **non-differentiable.** Their poor scalability makes real-time planning and control particularly challenging.

We address these issues with a configuration-conditioned **Neural** Signed Distance Function for **mobile manipulators**. To train it, we first generate a dataset of analytic distances computed from the robot geometry. The resulting model provides a continuous, differentiable representation of proximity.

Existing Methods

- Distances from explicit robot/scene models
- High accuracy and interpretabily
- Computation grows with complexity
- Not always smoothly differentiable



- Proximity learned from data
- Continuous and differentiable functions
- Fast inference
- Dependent on data; less precise near contact

Robot Configuration & Workspace

- The robotic system considered is a mobile manipulator, composed of a holonomic mobile base and a 6-DOF serial arm manipulator mounted on top. Using a simplified kinematic model, the robot state is defined as

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_{\text{base}} \\ \mathbf{q}_{\text{arm}} \end{bmatrix} = \begin{bmatrix} (x \ y \ \psi)^\top \\ (q_1 \ q_2 \ q_3 \ q_4 \ q_5 \ q_6)^\top \end{bmatrix}$$

- The workspace is the space where the robot moves, i.e., $\mathbb{R}^3$
- We want to characterize the distance of a point (typically, a representative point on the obstacle) from the robot
- Distances are invariant with respect to rigid transformations, thus $x, y, \psi = 0$

Distance Definition

Unsigned Euclidean Distance

Given a point $\mathbf{p} \in \mathbb{R}^3$, the unsigned Euclidean distance between this point and the k-th link of the robot is defined as

$$\phi_k(\mathbf{p}, \mathbf{q}) = \min_{\mathbf{r} \in \mathcal{M}_k(\mathbf{q})} \|\mathbf{p} - \mathbf{r}\|_2, \quad k = 1, \ldots, N$$

where $\mathcal{M}_k(\mathbf{q})$ denotes the surface of the k-th link of the robot

Signed Distance

Since we are interested in the signed distance, we define

$$d_k(\mathbf{p}, \mathbf{q}) = \mathrm{sgn}_k(\mathbf{p}, \mathbf{q})\, \phi_k(\mathbf{p}, \mathbf{q})$$

where $\mathrm{sgn}_k(\mathbf{p}, \mathbf{q})$ is a sign function that determines whether the points lies inside or outside the k-th link

Dataset and learning objective

The vector collecting the signed distances from all links is given by

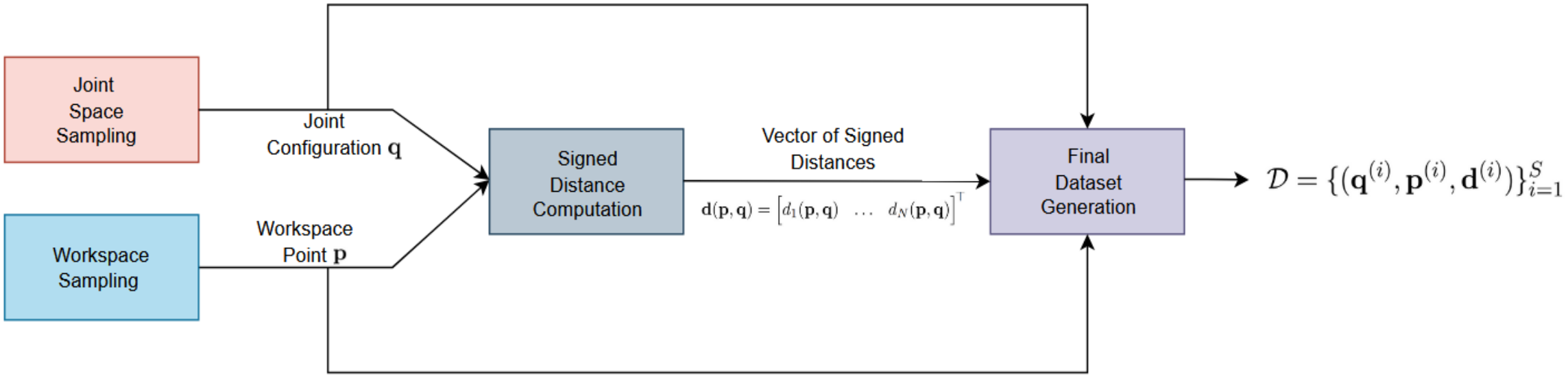
$$\mathbf{d}(\mathbf{p},\mathbf{q}) = \begin{bmatrix} d_1(\mathbf{p},\mathbf{q}) & d_2(\mathbf{p},\mathbf{q}) & \dots & d_N(\mathbf{p},\mathbf{q}) \end{bmatrix}^\top$$

A single dataset sample is expressed as

$$\xi = (\mathbf{q}, \mathbf{p}, \mathbf{d}(\mathbf{p},\mathbf{q}))$$

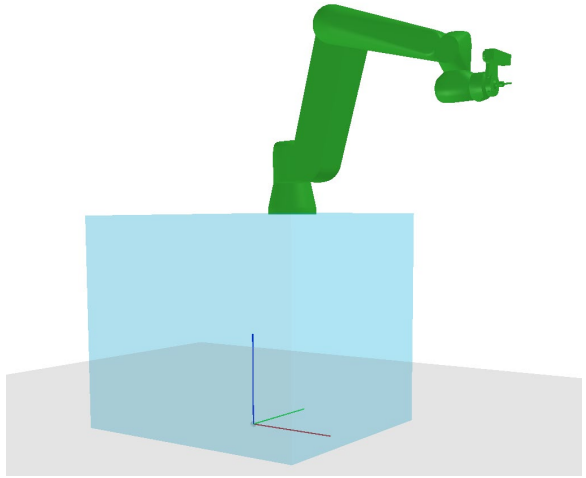
The whole dataset is used to train a neural model, that is designed to approximate the mapping between the robot configurations and the link-wise distances

Dataset Generation Pipeline



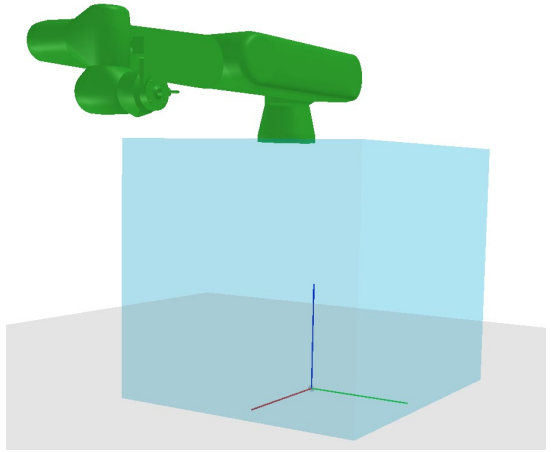
- N: number of links
- S: number of samples

Geometric Representation



To generate the dataset efficiently, the robot geometry is simplified into a lightweight but conservative model that preserves its occupied volume while reducing computational complexity. In particular, to achieve this, we use:

- **Collision Meshes**
- **Base Abstraction**



Joint Space Sampling (1/3)

Objective

The goal of this phase is to generate a set of **self-collision-free** robot configurations that uniformly cover the robot workspace (actually, a box around the base configuration)

Sampling Strategy

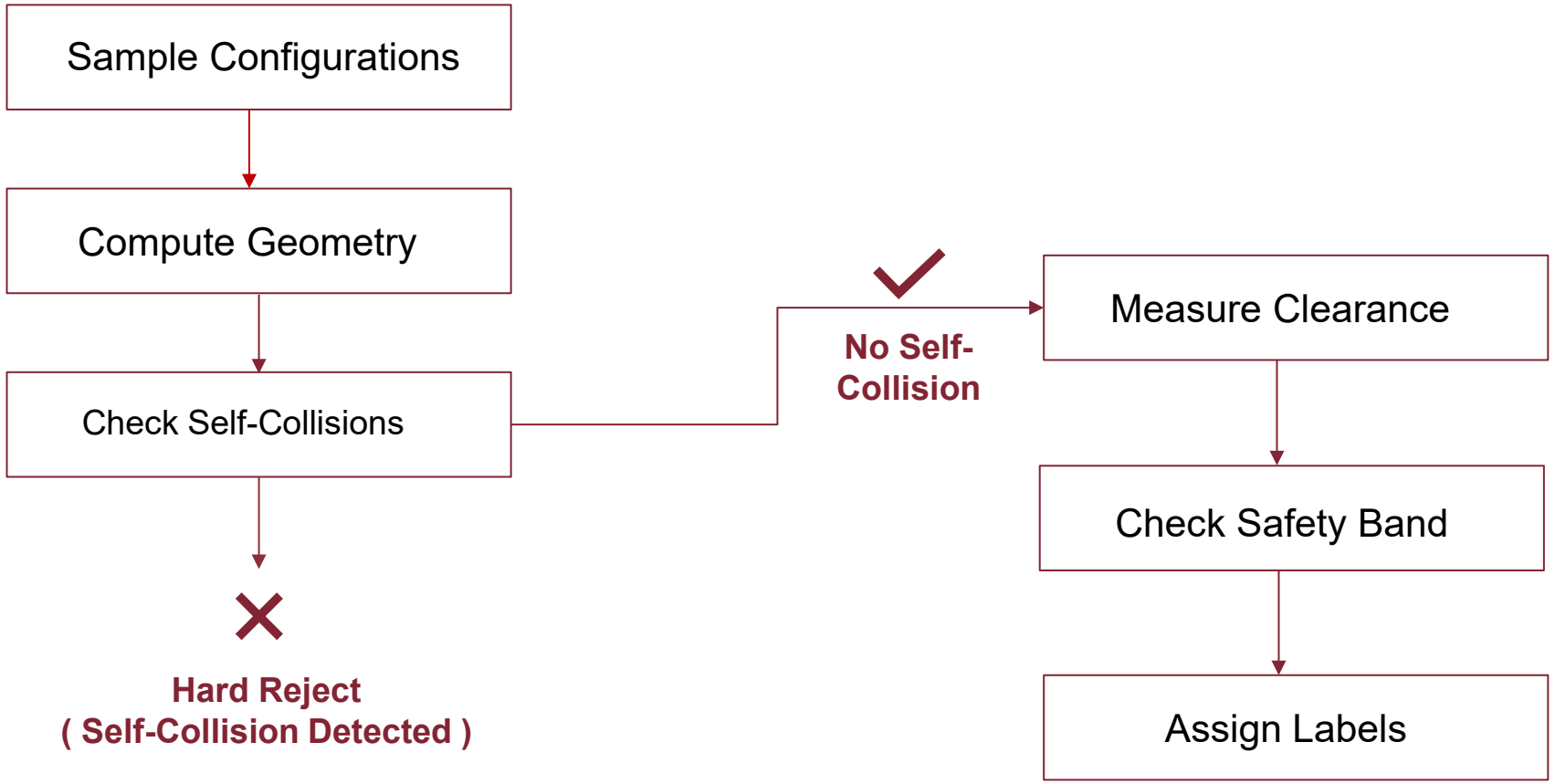
Each joint is sampled within its physical limits :

$$\mathcal{Q}_{\text{arm}} = \left\{ \mathbf{q}_{\text{arm}} \in \mathbb{R}^6 \;\middle|\; q_i^{\min} \leq q_i \leq q_i^{\max},\ i = 1, \ldots, 6 \right\}$$

To obtain a uniform coverage, we use a scrambled Sobol sequence, mapping each Sobol point $\mathbf{s}^{(k)} \in [0,1]^6$ to the configuration sample $\mathbf{q}_{\text{arm}}^{(k)}$ through an affine transformation that complies with the joint limits

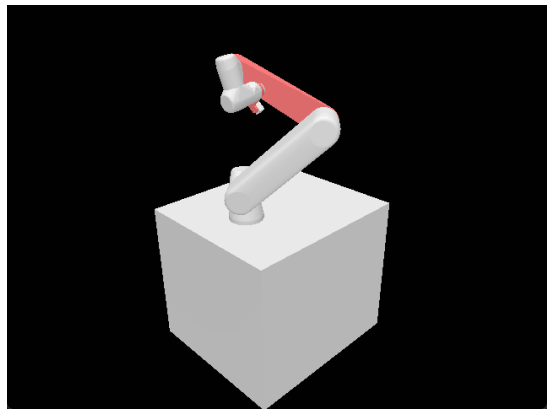
$$\mathbf{q}_{\text{arm}}^{(k)} = \mathbf{q}^{\min} + (\mathbf{q}^{\max} - \mathbf{q}^{\min}) \odot \mathbf{s}^{(k)}$$

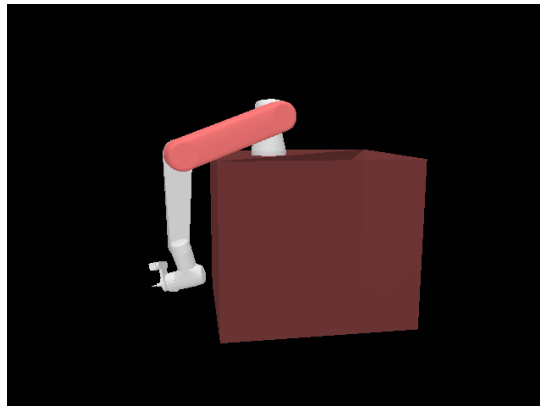
Joint Space Sampling (2/3)



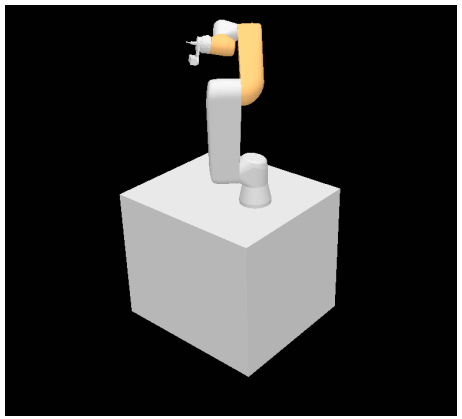
Joint Space Sampling (3/3)

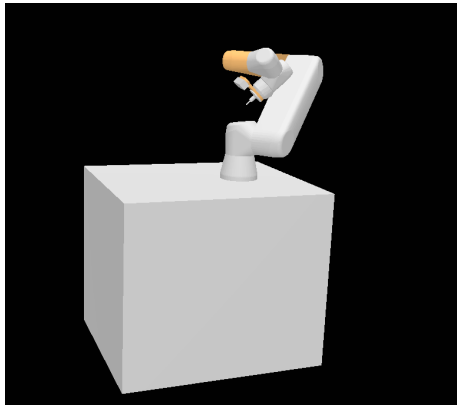
Reject



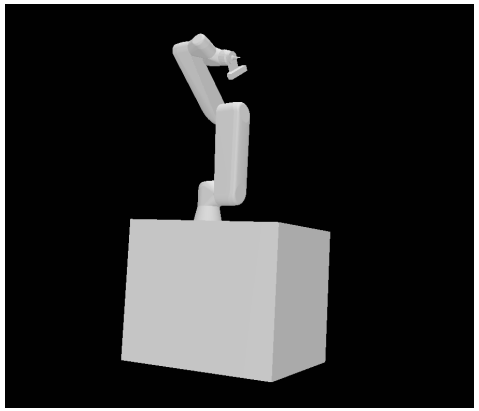


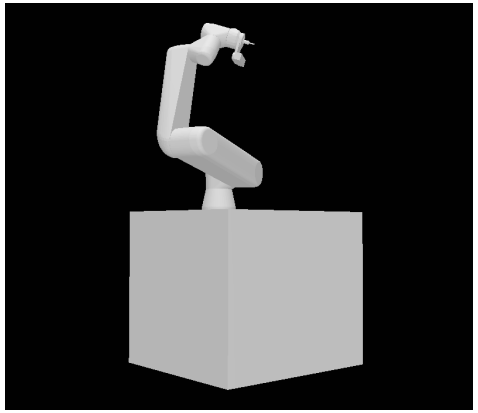
Near





Safe





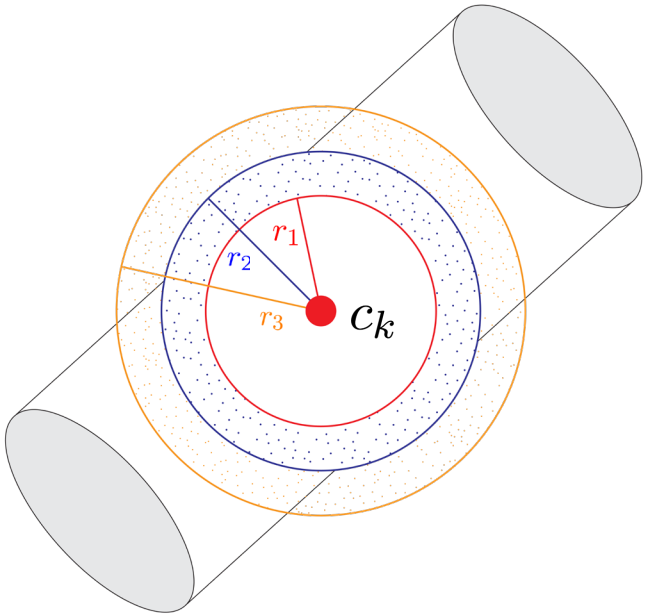
Workspace Sampling

Objective

The objective of this procedure is to establish a sampling scheme that tries to guarantee **uniform** and **scalable coverage** of the space surrounding each robot link — from the millimetric regions closest to the surface, where safety is most critical, to the larger volumes of the free workspace

Implementation

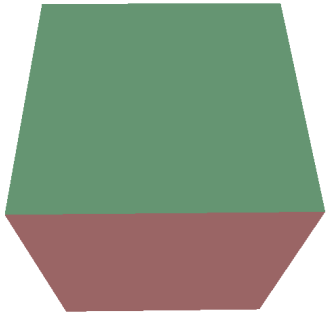
For each link, workspace points are uniformly sampled from **spherical shells**, of increasing radius, centered at the link barycenter c_k



Exposed Masks and Point Discarding

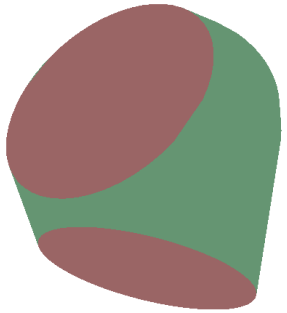
Ground Filter

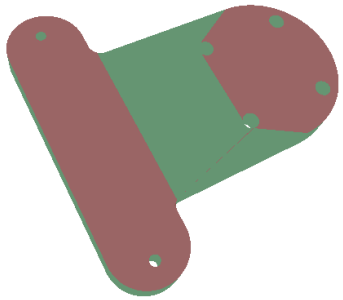
We remove faces located **near the ground** plane and **oriented downward**, such as the underside of the base. They are identified by checking the face centroid height and the direction of its normal



Joint Contact Filter

We exclude faces around mechanical joints, where two links can be very close with opposite normals. They are detected by measuring the distance to the **adjacent link** and verifying the **relative orientation of the normals**.





Distance Computation

Unsigned Euclidean Distance

To find the closest point on the robot surface from a point in the space, we:

1. Build a **tree of boxes** that groups the triangles of the robot's surface
2. Prune **distant** triangles using the tree
3. Compute the **closest point** on the remaining triangles

Signed Distance

The **generalized winding number** is used to determine whether a point lies **inside** or **outside** a solid surface

- It quantifies how much the surface encloses a point in space
- For each point:
  - Winding number ≈ 1 $\rightarrow$ inside the robot link
  - Winding number ≈ 0 $\rightarrow$ outside

Dataset Ownership and Balancing

Ownership

Formally, for each point $\mathbf{p}$ and robot configuration $\mathbf{q}$, we have access to the vector of link-wise signed distances. The critical distance is defined as the minimum absolute distance

$$d_{\min}(\mathbf{p}, \mathbf{q}) = \min_i \ |d_i(\mathbf{p}, \mathbf{q})|$$

and the link realizing this minimum is designated as the owner of the point

$$i^*(\mathbf{p}, \mathbf{q}) = \arg\min_i \ |d_i(\mathbf{p}, \mathbf{q})|$$

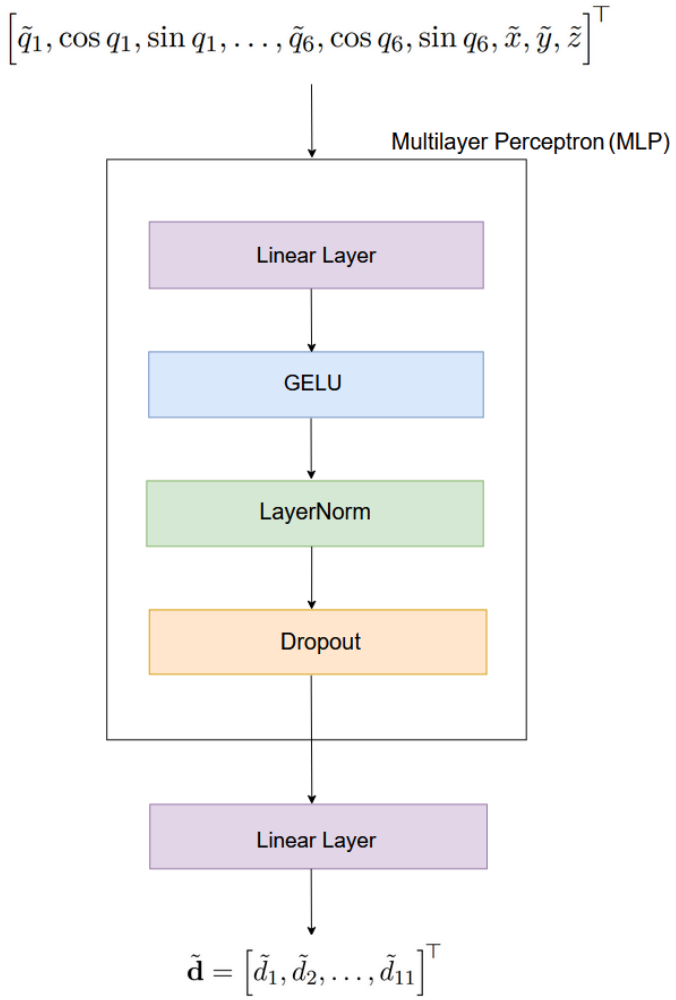
Balancing

Once ownership has been established, dataset balancing is carried out w.r.t. the owner distance

Points are collected into distance bands:

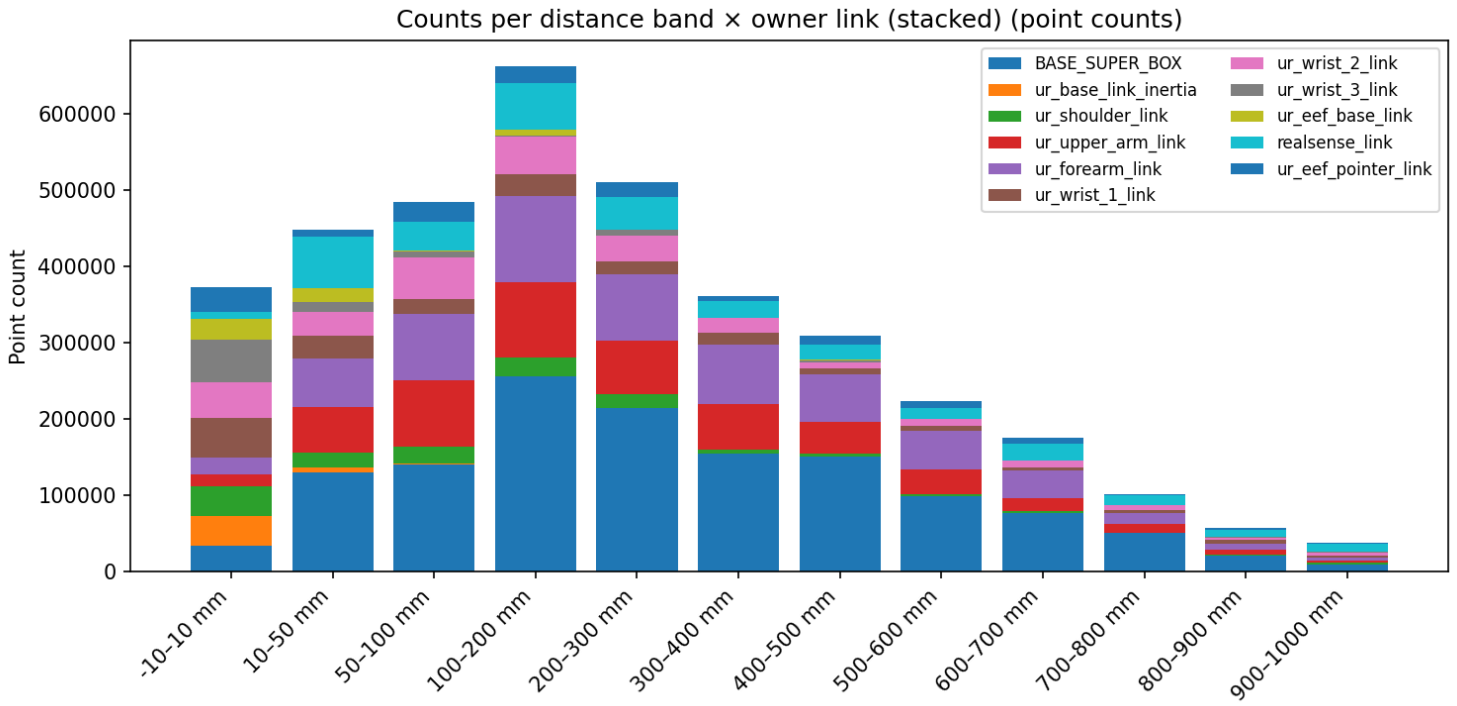
- **Very Near:** [-10, 10] mm
- **Near:** [10, 200] mm
- **Mid:** [200, 700] mm
- **Far:** [700, 1000] mm

Network Architecture



- Inputs and distances are transformed to stabilize training
- A fully connected MLP with 5 hidden layers x 256 neurons is implemented
- A **stratified sampling strategy** is adopted to ensure coverage of the bands in each batch
- The loss function is composed of a **Regression term**, weighted per link, and a **Sign Penalty term**

About the Dataset



- **Size :** ~4.1M query samples, each annotated with signed distances (in mm) to the 11 links of the robot
- **Distribution:** long-tailed, about ~92% of samples within 1 m of the robot, with progressively fewer samples beyond 1 m
- **Splits:** 80/10/10 train/validation/test, split by configuration

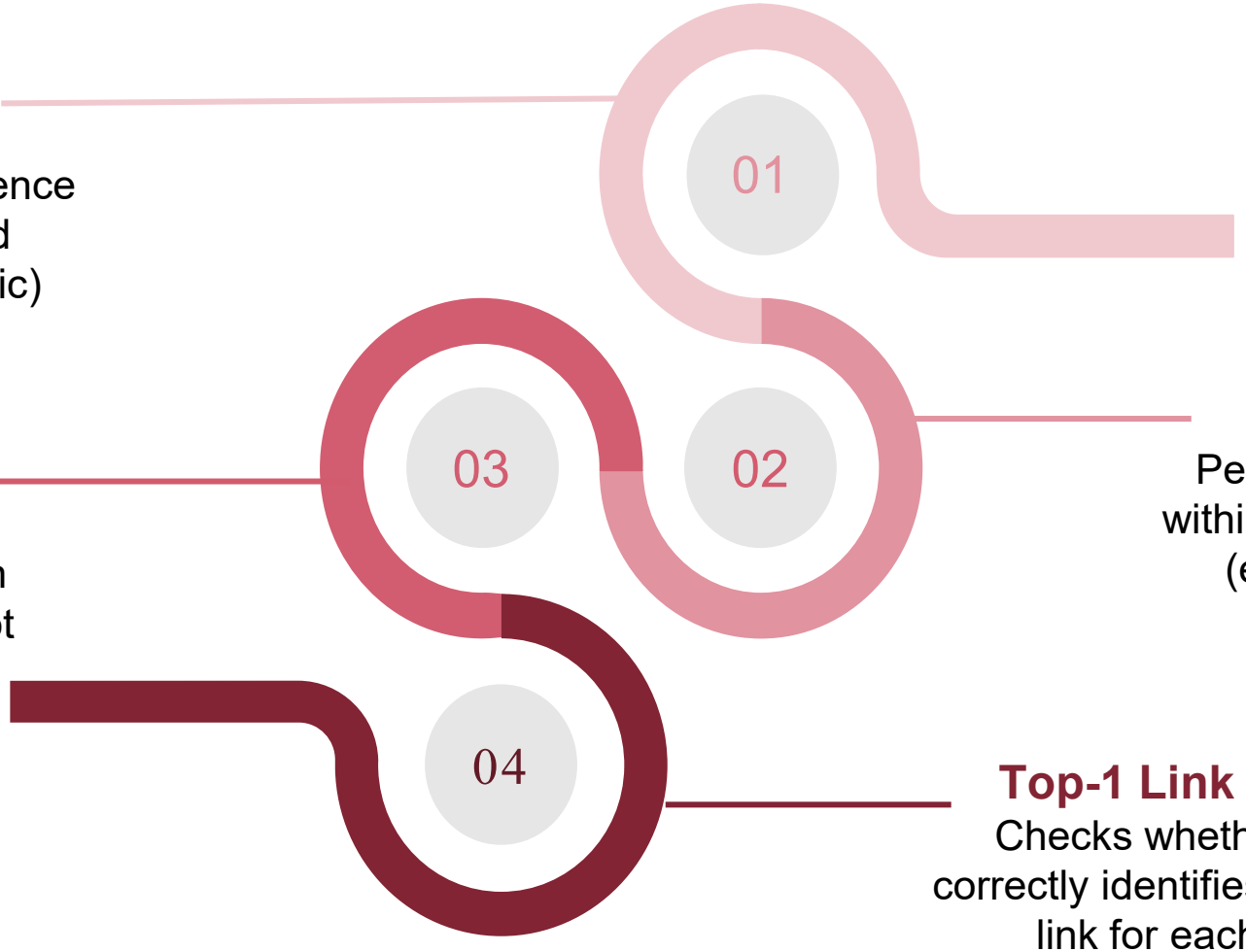
Metrics

Mean Absolute Error (MAE)

Average absolute difference between learned and ground-truth (geometric) distances

Sign Error

Rate of incorrect inside/outside classification within 10 mm from the robot surface



Precision within tolerance (P@ϵ)

Percentage of predictions within a tolerance threshold ϵ (e.g., 5 mm or 10 mm)

Top-1 Link Accuracy

Checks whether the network correctly identifies the closest robot link for each query point

Accuracy Evaluation (1/2)

Table 1. Per-link performance on the test set

Link	MAE [mm]	P@10 mm [%]
BASE_SUPER_BOX	15.7	46.4
ur_base_link_inertia	7.9	69.7
ur_shoulder_link	7.0	76.0
ur_upper_arm_link	9.8	62.0
ur_forearm_link	11.2	56.7
ur_wrist_1_link	9.9	61.2
ur_wrist_2_link	10.3	58.9
ur_wrist_3_link	10.2	59.1
ur_eef_base_link	11.1	56.4
realsense_link	14.8	46.6
ur_eef_pointer_link	10.6	58.0
Macro-mean	**10.8**	**59.5**

Accuracy Evaluation (2/2)

Table 2. Performance on $d_{\min}$ across distance regimes

Distance band	MAE [mm]
VeryNear ($\leq$ 10 mm)	7.4
Near (10–200 mm)	9.4
Mid (200–700 mm)	11.8
Far (700–1000 mm)	29.0

Table 3. Safety-related metrics in the near-surface region

Metric	Value [%]
Sign Error (@ $\lvert d_{\min}\rvert \leq 10$ mm)	20.4

Table 4. Top-1 link accuracy on the test set

Scenario	Accuracy [%]
Global (test set)	94.2
Near ($\leq$ 50 mm)	91.0

Inference Time Comparison

The inference time of the proposed neural model is compared with a traditional geometric baseline for computing the minimum signed distance between query points and the robot. Both methods use the same robot configuration and a batch of $\mathbf{B}$ query points. The Neural approach produces all distances in a single forward pass, while the Geometric approach performs nearest-point mesh queries and winding number computations

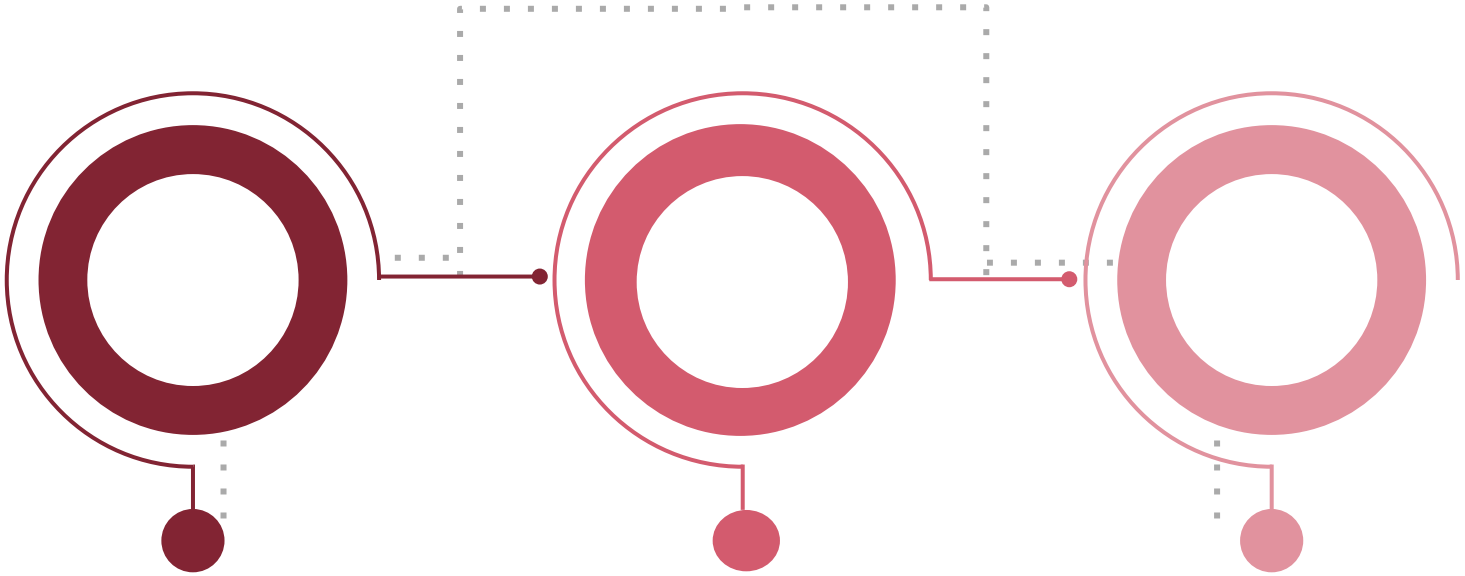
Table 5. Neural model inference times (single configuration)

B	p50 [ms]	μs/sample
1	0.560	560.1
8	0.567	70.9
32	0.570	17.8
128	0.569	4.4
512	0.571	1.1
1024	0.577	0.6

Table 6. Geometric method inference times (single configuration)

B	p50 [ms]	μs/sample
1	14.713	14713.1
8	47.141	5892.7
32	161.568	5049.0
128	602.889	4710.1
512	2741.814	5355.1
1024	6205.419	6060.0

Conclusion and Future Work



CONTRIBUTIONS

The method showed a neural pipeline for learning Signed Distance Function (SDF) of a mobile manipulator. It integrates dataset design, stratified sampling and tailored loss

RESULTS

The method showed a millimeter-level accuracy with reliable link association and good sign-consistency, in a significantly reduced computation time

FUTURE WORK

We would like to refine near-surface sampling and integrate SDF into motion planning and control pipelines