ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL, ELECTRONICS AND INFORMATION ENGINEERING "GUGLIELMO MARCONI" (DEI)

SECOND CYCLE DEGREE

Integrating Whole-Body Occupancy and Manipulability Map into Inverse Reachability Maps for Quality-Driven Base Positioning

Autonomous and Mobile Robotics

Supervisor
Prof. Gianluca Palli

Co-Supervisor
Prof. Andrea Govoni

Defended by
Mohammed Ali Abdullah Jabur

Graduation Session / MARCH / 2026

Academic Year 2025/2026

Abstract

Mobile manipulators—industrial robots like the UR5 mounted on mobile platforms—promise transformative flexibility for manufacturing and logistics. However, mobility introduces a critical challenge: determining where to position the base such that the manipulator executes multi-target trajectories with collision-free, kinematically optimal configurations. Existing inverse reachability methods treat collision detection as post-processing validation, requiring expensive iterative checking that becomes prohibitive in cluttered environments.

This thesis introduces the Inverse Hybrid Dynamic Reachability Map (iHDRM) framework, fundamentally integrating whole-body collision awareness through a precomputed Occupation Map. For every valid joint configuration, the framework maps which workspace voxels the robot occupies, enabling collision detection through set intersection rather than geometric computation. This reduces collision checking from $O(N_{\text{configs}})$ to $O(|V_{\text{obstacle}}|)$.

For trajectories, the framework computes the intersection of candidate base poses across targets ($B_{\text{common}} = B_1 \cap \cdots \cap B_N$), filters by collision constraints, and selects optimal locations through manipulability-based quality aggregation, providing completeness, correctness, and optimality guarantees.

Experimental validation demonstrates successful base placement for diverse trajectory shapes in cluttered environments. GPU-accelerated parallel computation enables practical offline map construction at high spatial resolution. The framework extends industrial manipulators with mobile flexibility while maintaining collision safety and kinematic quality, providing a practical path toward autonomous mobile manipulation.

Contents

List of Figures

Introduction

Industrial robotic manipulators—fixed-base systems like the Universal Robots UR5—form the backbone of modern automated manufacturing, performing tasks from assembly and welding to pick-and-place and inspection. These systems excel in structured environments where targets lie within their kinematic workspace and the robot base is permanently mounted at a predetermined location. However, **a fundamental limitation** constrains their deployment: the fixed base position defines a finite workspace, and any task requiring manipulation beyond this workspace becomes infeasible without *manually repositioning the entire robot*—a costly, time-consuming process requiring disassembly, recalibration, and production downtime.

Mounting fixed-base manipulators on mobile platforms—creating **mobile manipulators**—offers a transformative solution. By adding mobility, the same UR5 arm that could only reach an arm's length radius from a fixed location can now access entire warehouses, manufacturing floors, or service environments. The mobile platform becomes an additional degree of freedom, dynamically positioning the manipulator's fixed workspace wherever manipulation is needed. This flexibility enables diverse applications: mobile manufacturing cells serving multiple workstations, warehouse robots handling items across vast facilities, assistive systems operating throughout homes, and inspection robots navigating complex industrial sites.

Yet this mobility introduces a critical planning challenge: **determining where to position the mobile base** such that the mounted manipulator can successfully execute tasks while avoiding collisions. Unlike the fixed-base scenario where workspace analysis is performed once during installation, mobile systems must solve this placement problem *for every new task*. The manipulator's kinematic reach, geometric collision constraints, and configuration quality all depend on base location—and for tasks involving *trajectories* (such as wiping surfaces, opening drawers, or following welding seams), the base must enable collision-free execution across *all* trajectory points simultaneously.

This base placement problem is deceptively difficult. A naïve approach might position the mobile base close to a target object to maximize the manipulator's reach—yet this risks collision between the robot body (both the mobile platform and the arm links) and nearby obstacles. Conversely, maintaining safe distance from obstacles may place targets at the edge of the manipulator's workspace or force the arm into singular configurations where control authority degrades. The situation worsens in *cluttered environments*—precisely where mobile manipulation offers the greatest value—where navigation space is constrained and obstacles surround potential base locations from multiple directions.

Existing approaches to mobile manipulator planning typically address base placement and motion planning as **separate, sequential problems**. Reachability Map (RM) methods precompute the manipulator's workspace, enabling rapid queries about which poses the arm's end-effector can reach from a given base location. Inverse Reachability Map (IRM) methods invert this representation, identifying potential base positions for a desired end-effector pose. While computationally efficient, these methods suffer from a critical limitation: they treat

collision detection as a *post-processing validation step* rather than an integrated component of the representation. When a selected base location proves to be in collision, the system must iteratively sample alternative positions and re-validate—a process that becomes prohibitively expensive in cluttered environments where many candidate locations fail.

This limitation stems from a deeper issue: standard reachability representations lack **whole-body spatial awareness**. An inverse kinematics (IK) solution specifies joint angles that position the end-effector at a target pose, but the UR5—like most 6-DOF manipulators with spherical wrists—can achieve the *same* end-effector pose through **up to 8 distinct joint configurations**. These alternative solutions differ in their shoulder orientation (left vs. right), elbow configuration (up vs. down), and wrist orientation (flip vs. no-flip)—and consequently occupy dramatically different volumes in 3D space. One configuration might sweep the arm through a table surface while another clears it safely. Existing methods store these alternative configurations but fail to precompute and index *which voxels each configuration occupies*—information essential for rapid collision filtering. Consequently, online planning must perform expensive geometric collision checks for every candidate base location across every trajectory point, creating a computational bottleneck that limits real-time responsiveness.

0.1 The iHDRM Framework: Integrated Collision-Aware Base Placement

This thesis introduces the **Inverse Hybrid Dynamic Reachability Map** (iHDRM) framework, a comprehensive solution that fundamentally integrates whole-body collision awareness into mobile manipulator base placement. The framework advances the state-of-the-art through three key innovations:

First, the iHDRM incorporates an **Occupation Map** that explicitly tracks which workspace voxels are occupied by the robot's body for every stored joint configuration. During offline construction, the framework computes forward kinematics for all valid configurations (all 8 possible IK solutions for the UR5), representing each robot link as a point cloud and mapping these points to voxel indices. Each voxel stores a list of configuration indices that occupy it—a reverse lookup structure enabling instant identification of which configurations would collide with obstacles mapped to that voxel. This precomputation transforms online collision checking from an expensive per-configuration geometric operation into a rapid set intersection between obstacle voxels and occupation lists.

Second, the framework provides **explicit multi-target trajectory optimization**. Rather than aggregating base poses through simple minimum or maximum operations, the iHDRM computes the intersection of valid poses across all trajectory targets—identifying base locations from which every point is reachable. This intersection is computed efficiently by transforming the IRM to each target pose (rather than transforming the massive voxel grid), finding collision-free configurations at each target, and progressively filtering the common base pose set. The framework then validates these candidates by ensuring at least one collision-free configuration exists for each target, eliminating positions where geometric feasibility across the full trajectory cannot be guaranteed.

Third, among validated base locations, the framework selects the optimal pose through **manipulability-based quality assessment**. For each candidate base, the system identifies the maximum Yoshikawa manipulability index achievable at each trajectory target, then computes the mean of these maxima. This quality measure—$\text{Quality}(b) = \frac{1}{N}\sum_t w_{\max}(b,t)$—ensures the selected base provides consistently high kinematic flexibility across the entire trajectory, avoiding singular configurations and enhancing task execution robustness.

0.2 Significance and Impact

The iHDRM framework addresses a critical gap in mobile manipulation: the ability to **efficiently and reliably** position mobile bases carrying fixed manipulators in cluttered, real-world environments. By integrating collision awareness directly into the map structure rather than treating it as post-processing, the framework enables:

- **Real-time responsiveness** through precomputed occupation data that eliminates iterative collision validation
- **Guaranteed feasibility** for trajectory tasks by verifying collision-free configurations exist for all targets before execution
- **Quality optimization** through manipulability-based selection, ensuring robust task execution even in confined spaces
- **Configurational diversity exploitation** by storing and evaluating all 8 IK solutions per pose for the UR5, finding collision-free alternatives when primary solutions fail

These capabilities transform the deployment paradigm for industrial manipulators. Rather than dedicating a UR5 to a single fixed location, manufacturers can deploy the same robot on a mobile platform to serve multiple workstations, adapt to changing production layouts, or handle exceptional tasks requiring manipulation in diverse locations. Applications include: mobile manufacturing cells alternating between assembly stations without reconfiguration; warehouse automation where UR5-equipped platforms manipulate items across vast facilities; flexible production systems adapting to product variations by repositioning manipulators; and inspection tasks requiring the robot to navigate to various equipment locations. In each case, the ability to rapidly identify safe, high-quality base placements directly determines system effectiveness and return on investment.

0.3 Thesis Organization

This thesis is organized as follows. **Chapter 1** reviews related work on reachability representations, inverse reachability methods, and collision-aware planning, identifying the specific limitations addressed by the iHDRM framework. **Chapter 2** details the Reachability Map construction methodology, establishing the foundational workspace representation for fixed-base manipulators. **Chapter 3** introduces the complete iHDRM framework, describing offline construction of the Occupation Map and online querying for collision-aware, quality-optimized base placement, including the enhanced multi-target trajectory handling methodology with set-theoretic formulations and manipulability-based optimization. **Chapter 4** presents the theoretical foundations and computational tools—inverse kinematics for the UR5 (explaining the 8-configuration multiplicity), manipulability measures, signed distance functions, and GPU acceleration—that enable efficient offline map construction. **Chapter 5** presents experimental validation demonstrating the framework's effectiveness in identifying collision-free, high-quality base placements for diverse trajectory shapes (linear, diagonal, sinusoidal, U-shaped, L-shaped) in cluttered environments, while analyzing methodological considerations including resolution-dependent discretization, conservative collision approximation, and memory scaling challenges. **Chapter 6** concludes with a synthesis of contributions, discussion of practical deployment implications, acknowledgment of current limitations, and identification of future research directions including dynamic environment adaptation, multi-robot coordination, task-specific quality metrics, and integration with high-level task planning frameworks.

The central thesis of this work is that explicit integration of whole-body collision awareness into inverse reachability representations, combined with multi-target trajectory optimization

and quality-based selection, provides the comprehensive foundation necessary for reliable mobile manipulator deployment. By precomputing occupation data for all kinematically distinct configurations (all 8 IK solutions for robots like the UR5), the iHDRM framework demonstrates that comprehensive collision-aware base placement is both theoretically sound and computationally tractable, offering a practical path toward extending the capabilities of industrial fixed-base manipulators through strategic mobility. [1]

[1] In accordance with university regulations, AI tools such as ChatGPT, Google Gemini, and Claude AI were used to assist with paraphrasing and improving writing clarity. All core ideas and content remain the sole work of the author. Use of AI complies with UNIBO guidelines on transparency and academic integrity.

Chapter 1

related works

Mobile manipulator base placement represents a critical challenge in robotics, requiring the integration of workspace analysis, kinematic feasibility, and collision avoidance. This chapter reviews existing approaches to reachability analysis and base placement optimization, organizing the literature into three main categories: foundational reachability representations, inverse reachability methods for base placement, and recent advances addressing computational efficiency and collision awareness.

1.1 Reachability Map Representations

1.1.1 Capability Maps and Workspace Characterization

Zacharias et al. [1] introduced the **Capability Map** as a foundational framework for representing a robot arm's kinematic reachability and directional structure. Unlike randomized sampling methods that provide only binary reachability information, the capability map captures directional characteristics through geometric primitives (cones, rings, cylinders). The workspace is discretized into voxels, with spheres inscribed in each voxel to sample orientational reachability. A *reachability index* R/N quantifies the percentage of successfully reachable poses on each sphere, where R represents reachable points and N total sampled points. Geometric structures observed in IK solution distributions are approximated using shape primitives, enabling rapid direction testing compared to evaluating individual IK solutions.

Zacharias et al. [2] extended this work to **constrained linear trajectories**, addressing tasks such as drawer opening. Their two-stage approach first intersects the reachability sphere map with a task plane, then applies pattern recognition and morphological operations to identify feasible trajectory starting points. The second stage computes and validates mobile base placements, checking both kinematic consistency and collision-free execution. This work established the paradigm of using reachability representations for mobile manipulator placement.

1.1.2 Quality Metrics and Manipulability

Burget and Bennewitz [4] emphasized that reachability alone is insufficient **configuration quality** critically impacts task execution. They introduced manipulability-based quality assessment using the Yoshikawa measure $w = \sqrt{\det(J(q)J^T(q))}$, where $J(q)$ is the Jacobian matrix. For humanoid robots, they constructed reachability maps covering workspace volumes reachable from statically stable double-support configurations, storing both joint configurations and manipulability indices per voxel. The *Inverse Reachability Map* (IRM) inverts this representation, enabling rapid stance pose selection by maximizing manipulability among collision-free configurations.

1.2 Inverse Reachability for Base Placement

1.2.1 Foundational Inversion Methods

Vahrenkamp et al. [3] pioneered the systematic application of **reachability inversion** for mobile manipulator base placement. Their approach inverts all base-to-TCP transformations from a Reachability Distribution (RD), creating an *Inverse Reachability Distribution* (IRD) centered on target poses. By applying kinematic constraints (e.g., upright standing on flat ground), the 6D IRD is reduced to a 3D *Oriented Reachability Map* (ORM) in $SE(2)$. For multi-target trajectories, they compute an aggregated $\text{ORM}_{\text{tr}} = \min\{\text{ORM}_i(x, y, \gamma)\}$, selecting base locations reaching all trajectory points. A *Lazy ORM* variant computes entries on-demand, accelerating single IK queries.

1.2.2 Tool Frame Extensions and Orientation-Based Representations

Dong and Trinkle [5] identified a critical limitation: position-based reachability maps (PBR maps) cannot accommodate **online end-effector frame extensions**, requiring complete map recomputation for each new tool—impractical for unknown or variable tools. Their *Orientation-Based Reachability Map* (OBR map) exploits the observation that extended orientation terms involve only rotation, not translation. By indexing first by orientation, then position (x,y,z), the OBR map enables tool frame extensions in seconds rather than hours. The same pattern-matching algorithm applies to both fully-specified and partially-constrained paths, with the OBR map providing crucial flexibility for real-world manipulation.

1.2.3 Dimensionality Reduction Techniques

Rudorfer [13] introduced **RM4D**, reducing reachability map dimensionality from 6D to 4D for common 6/7-axis arms. Exploiting two key assumptions—360° base rotation and 360° wrist rotation about the approach vector—the method maps 6D poses to 4D structures using (p_z, θ, x^*, y^*), where θ is the approach vector angle and (x^*, y^*) represents canonical base positions. This compact representation requires an order of magnitude less memory and build time than traditional maps while maintaining state-of-the-art accuracy, using a single 4D structure for both forward reachability and inverse base placement queries.

1.3 Collision-Aware Planning and Dynamic Environments

1.3.1 Whole-Body Collision Detection

Yang et al. [6] addressed a fundamental limitation: standard IRM methods ignore collisions, requiring expensive post-query validation. Their **Inverse Dynamic Reachability Map** (iDRM) integrates collision awareness directly into the map structure through two critical lists per voxel: a *Reach List* (samples with stance frames in the voxel) and an *Occupation List* (samples intersecting the voxel with any body part). During online queries, collision detection runs once between environment obstacles and iDRM voxels—if a voxel is occupied, all samples in its occupation list are immediately invalidated. This enables real-time collision filtering for millions of samples without individual checks, with feasible configurations further filtered by ground contact and balance constraints.

1.3.2 Neural Network-Based Map Updates

Sandakalum et al. [11] proposed **Inv-Reach Net**, a neural architecture that explicitly accounts for obstacles when updating IRMs. The network comprises two sub-models: a 3D convolutional classifier predicting which voxels suffer reachability reduction, and an autoencoder regressing the

specific reduction magnitude. Given a probabilistic occupancy grid, the network outputs ΔIRM, with the updated map computed as $\text{IRM}_{\text{updated}} = \text{IRM}_{\text{master}} - \Delta\text{IRM}$. Training leverages Master IK data files to avoid redundant IK queries. The approach significantly outperforms traditional methods in speed and success rates, enabling rapid adaptation to changing environments.

1.3.3 Surface-Based Reachability for Task Planning

Hertle and Nebel [7] extended reachability inversion from point-based to **surface-based representations**, addressing Combined Task and Motion Planning (CTAMP) challenges. Their surface inverse reachability map provides a surface reachability index $R_S = \frac{1}{n}\sum_i R_{P_i}$, representing the percentage of surface area reachable from a given pose. Points are sampled on a grid inside surface polygons, with validation ensuring collision-free IK solutions. Integration with symbolic planning via semantic attachments enables gradual pose sampling as needed, eliminating the need for exhaustive a priori pose generation.

1.4 Application-Specific Methods and Practical Implementations

1.4.1 Multi-Elevation and Unstructured Environments

Birr et al. [10] introduced the **Oriented Surface Reachability Map** (OSRM) for humanoid base placement on arbitrarily oriented 3D planes, not just flat $SE(2)$ surfaces. Combining an IRM with planar region segmentation of height maps, the method ensures steady ground contact, target reachability, and collision-free placement. Planar regions $r = (n, d, W)$ are represented in Hessian normal form ($n^T p + d = 0$), with the OSRM mapping 3D tuples (x, y, γ) to reachability values. The approach models planar regions as a graph, detecting traversable edges and adding non-traversable edges to the obstacle space, enabling safe 2D path planning in 3D environments with multiple elevation levels.

1.4.2 Domain-Specific Optimizations

Cho et al. [14] applied reachability inversion to **agricultural harvesting**, optimizing Kanpei citrus picking. Their manipulability score $\mu_i = \sqrt{\lambda_{\min}/\lambda_{\max}}$ assesses distance from singularities using Jacobian eigenvalue ratios. For multi-fruit harvesting from a single position, IRMs are overlaid with aggregated scores—crucially, positions failing to reach any fruit receive negative values, preventing selection of high-scoring but incomplete solutions.

Zhong et al. [12] developed an OGM-IRM fusion method for **wheelchair-mounted robotic arms** (WMRAs). The system merges Occupied Grid Maps from SLAM (Cartographer algorithm) with IRMs showing grasping ability from various base placements. Optimal placements identified in the IRM are validated against OGM obstacle data, with the WMRA automatically moving to collision-free positions. While improving efficiency and coordination, the method's computational intensity currently prevents real-time operation.

1.4.3 Open-Source Tools and Coverage Planning

Makhal and Goins [9] released **Reuleaux**, an open-source library automating reachability analysis and base placement from URDF models. Workspace voxelization uses octrees for efficient spatial partitioning, with self-collision pre-checking to exclude invalid regions. A Directional Reachability Measure $\text{DRM} = (R/N) \times 100$ quantifies each voxel's reachability. For mobile robots on flat ground, the 6D union map reduces to $SO(2)$ through horizontal slicing. A PlaceBase index D_{PB} identifies high-probability base locations, with the final selection considering reachability scores across all task poses. Primary limitations include manual task pose specification and lack of collision checking during base placement.

Osswald et al. [8] applied IRM-based methods to **viewpoint planning** for humanoid 3D environment coverage. Rather than dense volumetric structures, their IRM is stored as a k-d tree or octree database indexed by base frame roll, pitch, and z components, enabling fast nearest-neighbor searches. View poses are sampled by ray-casting from surfaces, with valid configurations evaluated using a cost function prioritizing stability (foot weight distribution c_w) and energy (power P, time Δt) over maneuverability:

$$c = k_w \cdot c_w + k_t \cdot \Delta t + k_p \cdot P$$

The optimal path is solved as a Traveling Salesman Problem using Lin–Kernighan heuristics.

1.5 Industrial Applications and 3D Base Positioning

Ince et al. [15] developed the **ARBP-IR method** (Automated Robot Base Positioning based on Inverse Reachability) addressing industrial forging cell challenges where heavy immobile components constrain robot placement. Their approach extends inverse reachability to full 3D positioning (x,y,z), unlike prior 2D-limited methods. The reachability map assigns R = P/N values to voxels, where P is the number of reachable poses among N total poses sampled on spherical surfaces. For multiple Points of Interest (POIs), ORMs are overlapped by multiplying reachability values rather than selecting minimums, prioritizing positions with high reachability across all POIs. Validation checks collision, mechanical limits, and singularity avoidance. The method enables arbitrary orientations (including ceiling mounting) with explicit singularity checking, addressing critical gaps in prior industrial applications.

1.6 Summary and Research Gaps

The literature establishes a progression from basic reachability representations to collision-aware, application-specific methods. Key advances include: (1) *quality-based selection* through manipulability metrics, distinguishing feasible from optimal configurations; (2) *efficient inversion techniques* enabling rapid base placement queries; (3) *dimensionality reduction* strategies (RM4D, OBR maps) improving memory and computation efficiency; and (4) *collision integration* through occupation lists and neural network updates.

However, significant gaps remain. Most methods separate collision checking from reachability analysis, performing validation as a post-processing step. The iDRM [6] pioneered integrated collision awareness but lacks **whole-body occupation tracking** across robot configurations—critical for cluttered environments where different IK solutions for the same end-effector pose may have drastically different collision profiles. Furthermore, existing methods typically optimize for *single targets* or use simple aggregation (minimum/maximum) for trajectories, lacking sophisticated multi-target optimization considering both collision and quality metrics simultaneously. The computational burden of online collision checking for multiple configurations remains prohibitive for real-time applications.

The **iHDRM** framework presented in this thesis addresses these limitations by: (1) integrating whole-body occupation tracking directly into the inverted map structure, enabling rapid collision filtering across all robot configurations; (2) providing explicit multi-target trajectory optimization through common-base-pose intersection combined with manipulability-based selection; and (3) maintaining computational efficiency through offline precomputation of both kinematic solutions and geometric occupation data. This comprehensive approach bridges the gap between reachability-based planning and collision-aware execution, enabling practical deployment of mobile manipulators in complex, real-world environments.

Chapter 2

Reachability Map (RM)

The Reachability Map (RM) serves as the foundational data structure representing the kinematic capabilities of the robot arm within its operational space. It provides a comprehensive offline analysis of the robot's workspace, enabling efficient online motion planning by pre-computing and storing inverse kinematics solutions, manipulability metrics, and collision information. The generation of this map involves several systematic phases, from initialization to the computation of specialized sub-maps.

2.1 System Initialization

The process begins by establishing the digital twin of the robot and its environment, configuring both the kinematic model and the operational workspace.

2.1.1 Kinematic Setup

The robot kinematics are initialized by loading a configuration file containing the Denavit-Hartenberg (DH) parameters and the physical joint limits. These parameters define the geometric relationships between consecutive links and establish the kinematic chain that characterizes the robot's structure and motion capabilities.

2.1.2 Environment and Geometry Setup

The workspace is configured using a setup file that includes:

- The robot's Unified Robot Description Format (URDF)
- Paths to the robot's structural mesh files for accurate geometric representation
- The selected manipulability evaluation method (e.g., Yoshikawa manipulability measure)
- Collision detection algorithms for both environmental obstacles and self-collision
- Environment collision objects defining the workspace constraints
- Collision map for comprehensive self-collision detection

This initialization stage establishes the complete geometric and physical model necessary for accurate reachability analysis and collision checking.

2.2 Workspace Discretization

To systematically evaluate reachability across the continuous physical space, the workspace is converted into a discrete grid structure through voxelization.

2.2.1 Bounding Volume Definition

The theoretically possible workspace of the robot arm is encapsulated by a large bounding cube centered at the robot's base. The dimensions of this cube are derived from the total arm reach, creating a cubic volume with side length equal to the maximum arm extension.

2.2.2 Voxelization Process

The enveloping bounding cube is uniformly subdivided into smaller, equally sized subcubes called voxels. The resolution (voxel size) is determined by balancing two competing factors:

- **Task accuracy requirements:** Finer resolution provides more precise spatial information
- **Computational efficiency:** Coarser resolution reduces memory usage and computation time

This discretization approach serves multiple purposes: it enables intuitive 3D visualization of the workspace, facilitates efficient spatial queries during task planning, and allows the system to analyze specific operational regions with targeted precision. **Figure** 2.1 illustrates the discretized workspace structure, showing the cubic voxel grid extending throughout the volume reachable by the robot arm. The grid structure provides the foundation for all subsequent reachability analysis.

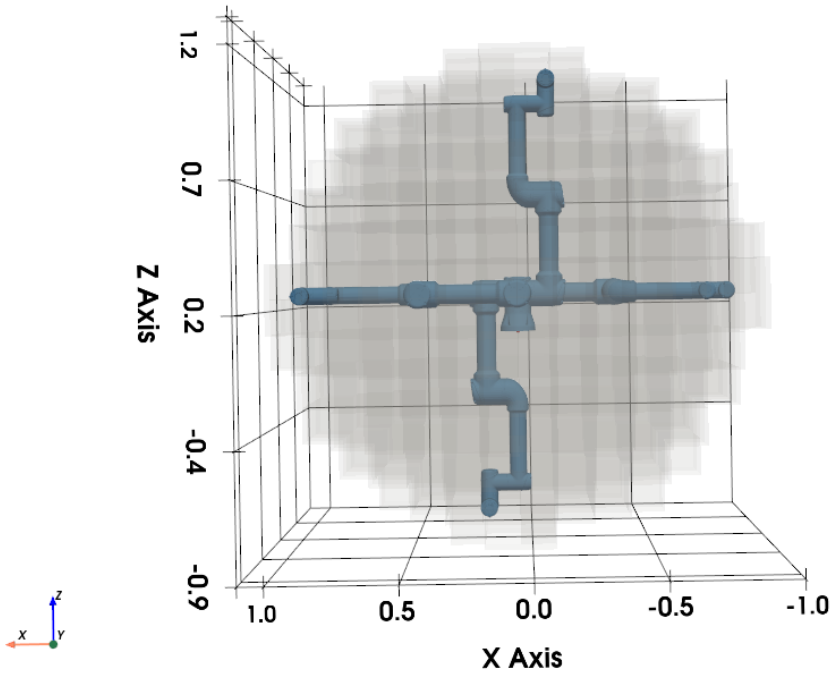

Figure 2.1: *Discretized workspace showing the volume covered by the robot arm*

2.3 Pose Generation and Spherical Sampling

Within each discrete voxel, the system evaluates the robot's ability to approach target positions from multiple orientations, providing a comprehensive assessment of directional reachability.

2.3.1 Sphere Inscription and Point Distribution

A sphere is inscribed within each voxel, serving as the locus for sampling target poses. Based on the required task accuracy and the desired granularity of orientation sampling, a set of target points is uniformly distributed across the surface of this sphere. This spherical sampling strategy ensures comprehensive coverage of all possible approach directions to a point within the voxel.

2.3.2 Orientation Assignment

For each point generated on the sphere surface, a specific target orientation is computed using spherical coordinates converted to Euler angles. The orientation is defined by three angles calculated as follows:

$$\phi = \arctan\left(\frac{y}{x}\right)$$

$$\theta = \arccos\left(\frac{z}{\text{radius}}\right)$$

$$\psi = \pi$$

where ϕ represents the azimuthal angle, θ the polar angle, and ψ a fixed roll angle. These angles define the end-effector orientation at each sampled point on the sphere. **Figure** 2.2 demonstrates the uniform distribution of sampled poses on the inscribed sphere within a single voxel. Each point on the sphere represents a potential end-effector position and orientation, with the density of points determining the orientational resolution of the reachability analysis.

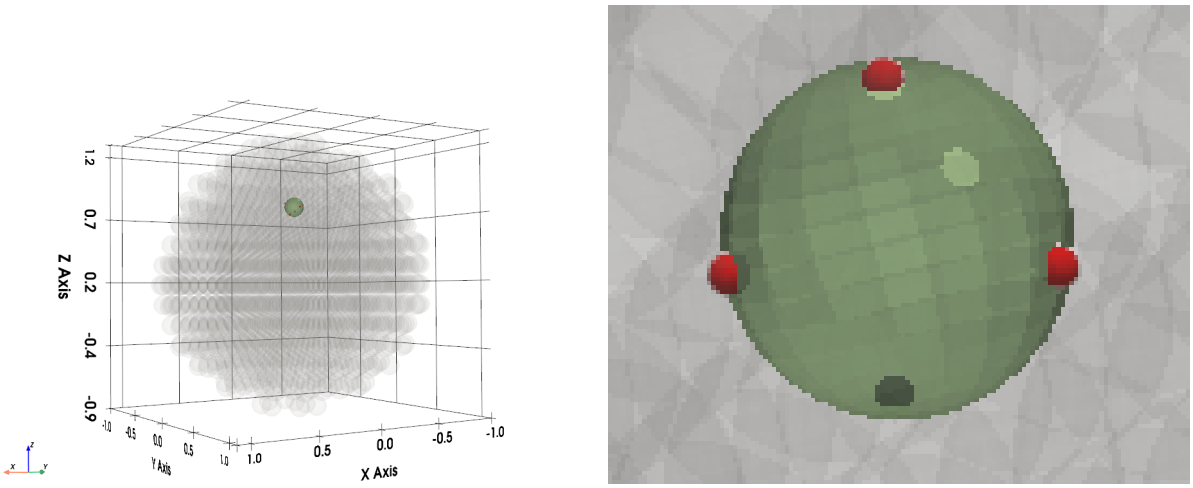

Figure 2.2: *Distribution of sampled poses on the inscribed sphere within a voxel*

2.3.3 Capability Map Initialization and IK Validation

The Capability Map is initialized with the workspace resolution, sphere discretization parameters, and robot-specific coefficients including the maximum number of inverse kinematics (IK) configuration solutions for each pose and the robot's degrees of freedom (DOF). An IK solver then attempts to find valid joint configurations for each generated pose on the sphere. Points are marked as reachable only if at least one valid IK solution exists that satisfies the joint limits and does not result in kinematic singularities.

2.4 Computation of Specialized Sub-Maps

Once the IK solutions are generated for all sampled poses, the data is organized into four specialized maps. Each map serves a distinct purpose in facilitating efficient online motion planning and task execution.

2.4.1 Reachability Map

The Reachability Map stores the local reachability status for each voxel, characterizing the robot's kinematic flexibility within different regions of the workspace. For each voxel, reachability is assessed based on the existence of valid joint solutions across all sampled poses on the inscribed sphere.

A **Reachability Index** is computed for each voxel to quantify its overall accessibility. This index represents the percentage of poses on the sphere for which at least one valid IK solution exists, considering the maximum allowable configuration solutions per pose:

$$\text{Reachability Index} = (\text{Number of reachable points}) \,/\, (\text{Total number of sampled points})$$

This metric provides a normalized measure ranging from 0 (no reachable poses) to 1 (all poses reachable), enabling direct comparison of kinematic flexibility across different workspace regions. **Figure** 2.3 visualizes the workspace voxels colored according to their reachability index. Bright regions (yellow) indicate high reachability where the robot can reach most sampled orientations, while darker regions (blue) show limited reachability or workspace boundaries. This visualization reveals the spatial distribution of kinematic capability, with central regions typically exhibiting higher reachability than peripheral zones.

2.4.2 Joint Solution Map

Rather than computing inverse kinematics solutions online during task execution, the Joint Solution Map pre-computes and caches all valid joint configurations for every successfully evaluated pose within each voxel. This offline computation eliminates the computational overhead of real-time IK solving, significantly accelerating motion planning queries.

For each reachable point on the sphere, all valid joint configurations (up to the maximum number of solutions) are stored with their associated voxel index. This enables instant retrieval of feasible joint-space trajectories during online planning, reducing the planning problem to a lookup and selection process rather than an optimization problem.

2.4.3 Manipulability Map

The Manipulability Map assesses the kinematic quality of every stored joint solution, providing a quantitative measure of how well the robot can move in arbitrary directions from each

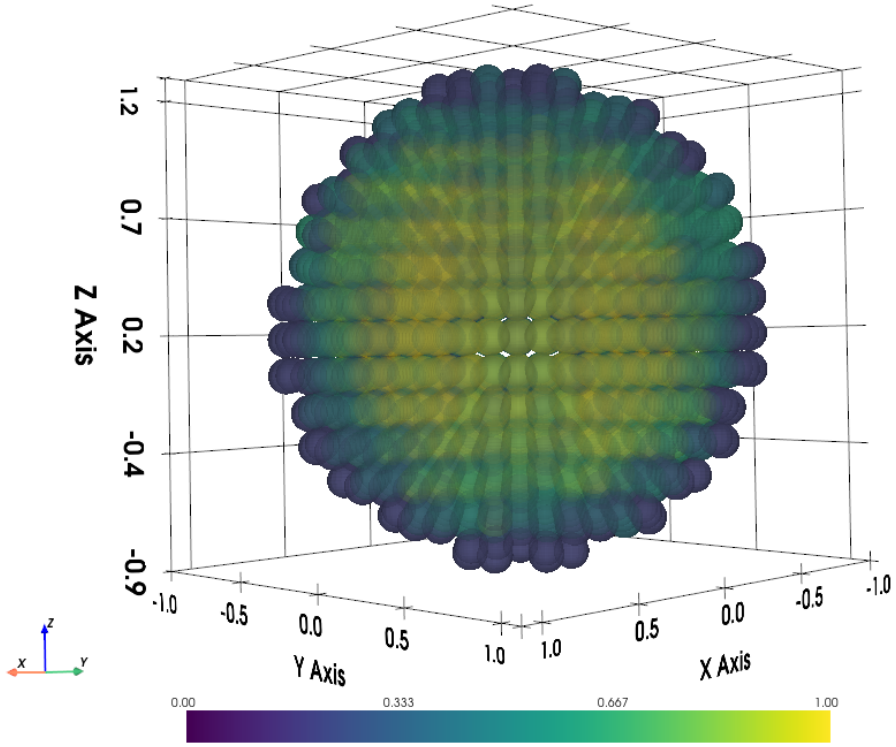

Figure 2.3: *Voxels colored by reachability index, showing regions of high (bright) and low (dark) reachability*

configuration. For each valid joint solution, a manipulability index is computed using the Yoshikawa manipulability measure, which is defined as:

$$w(q) = \sqrt{\det(J(q)J(q)^T)}$$

where $J(q)$ is the Jacobian matrix at joint configuration q. This metric quantifies the distance from kinematic singularities—higher values indicate better manipulability and greater freedom of motion, while lower values suggest proximity to singular configurations where the robot loses one or more degrees of freedom.

By storing these manipulability indices, the planner can intelligently compare multiple valid solutions for reaching the same pose and actively filter out poor or near-singular configurations, favoring solutions that provide better motion capability and control authority. **Figure** 2.4 demonstrates two joint configurations that can reach the same target point (sphere). The varying manipulability values for these configurations enable quality-based selection, ensuring the chosen solution provides optimal kinematic performance rather than simply any feasible solution.

2.4.4 Collision Map

The Collision Map provides comprehensive spatial awareness by storing collision information for each valid joint configuration. Rather than performing expensive collision checks during online planning, this map pre-computes and stores collision metrics for all robot links across all configurations.

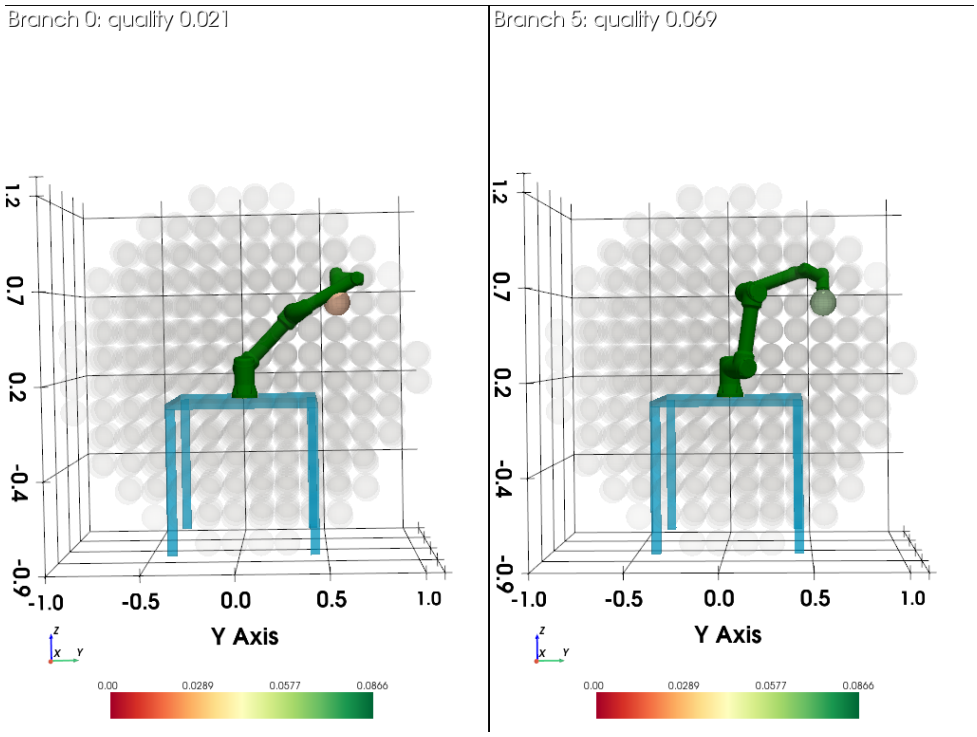

Figure 2.4: *Manipulability values for different joint configurations reaching the same target point*

For each joint configuration required to reach a point in the voxel, the map stores a Signed Distance Field (SDF) value for every robot link. The SDF represents the minimum distance between the link and the nearest obstacle:

- **Positive SDF values:** Indicate the link is in free space, with the value representing the clearance distance to the nearest obstacle
- **Negative SDF values:** Indicate the link is penetrating an obstacle, with the magnitude representing the penetration depth
- **Zero SDF value:** Indicates the link is in contact with an obstacle surface

This representation enables rapid collision filtering during online planning and provides gradient information that can be used for collision avoidance and trajectory optimization. Configurations with any link showing negative SDF values are immediately identified as colliding and can be excluded from the solution space. **Figure** 2.5 illustrates a robot configuration attempting to reach a target point (sphere) where collision occurs. The colliding robot links are highlighted in red, demonstrating how the SDF-based collision detection identifies geometric interference. Such configurations would be filtered out during planning, ensuring only collision-free solutions are considered.

2.5 Summary and Advantages

The Reachability Map framework provides a comprehensive offline characterization of the robot's kinematic capabilities within its workspace. By pre-computing and organizing inverse kinematics solutions, manipulability metrics, and collision information into specialized sub-maps, the framework enables several key advantages for online motion planning:

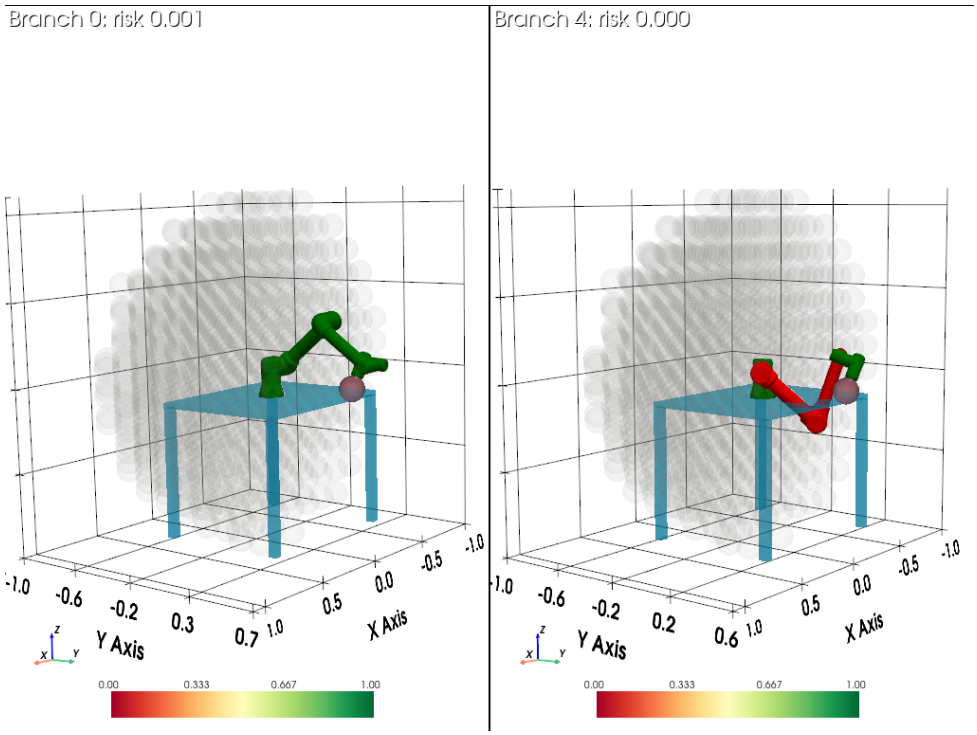

Figure 2.5: *Robot reaching a target point with colliding links highlighted in red*

- **Computational Efficiency:** Eliminates expensive online IK computation and collision checking, reducing planning time from seconds to milliseconds
- **Quality-Aware Planning:** Enables selection of high-quality configurations based on manipulability, avoiding singular or poorly conditioned poses
- **Spatial Understanding:** Provides intuitive visualization and analysis of workspace regions with varying kinematic capabilities
- **Collision Awareness:** Pre-computed SDF values enable rapid collision filtering and provide gradient information for optimization
- **Scalable Representation:** Voxelized structure allows efficient memory usage and spatial queries through standard data structures

These characteristics make the Reachability Map particularly valuable for applications requiring real-time or near-real-time motion planning in complex, constrained environments, forming the foundation for the more advanced Inverse Hybrid Dynamic Reachability Map (iHDRM) framework for mobile manipulator base placement.

Chapter 3

The iHDRM Framework: Construction and Query

The iHDRM (Inverse Hybrid Dynamic Reachability Map) framework is designed to optimize mobile base placement by integrating reachability analysis with whole-body collision awareness. The process is divided into an offline construction phase and an online query phase for real-time decision-making.

3.1 Phase A: Offline Construction of the iHDRM

3.1.1 Initialization of the Map

The iHDRM is initialized by inheriting resolution, workspace details, and kinematic coefficients (e.g., maximum configuration solutions) from the core Reachability Map (RM).

3.1.2 Computation of the Maps

The maps are computed by iterating through all reachable points in the RM map. The construction involves computing several specialized maps through kinematic inversion and spatial indexing.

3.1.2.1 Kinematic Inversion

For every reachable point in the RM, the position and orientation vectors are retrieved to construct a homogeneous transformation matrix ($T_{\text{base}\to\text{ee}}$). This matrix is inverted to obtain the base pose relative to the end-effector:

$$T_{\text{ee}\to\text{base}} = (T_{\text{base}\to\text{ee}})^{-1}$$

3.1.2.2 Map Generation

Based on this inversion, the following maps are populated:

A. Inverse Reachability Map
This map stores the density of valid base poses within each voxel. For each reachable point, the XYZ coordinates are used to find the corresponding voxel index, and the value of this voxel in the inverse reachability map is incremented by one. The inverse reachability index is calculated as:

$$\text{Inverse Reachability Index} = \frac{\text{Number of base poses in voxel}}{\text{Maximum number of base poses in any voxel}}$$

Figure 3.1 visualizes the workspace voxels colored according to their inverse reachability index, with warmer colors indicating regions where more base poses can reach the end-effector targets. High-density regions (green) represent optimal zones for base placement.

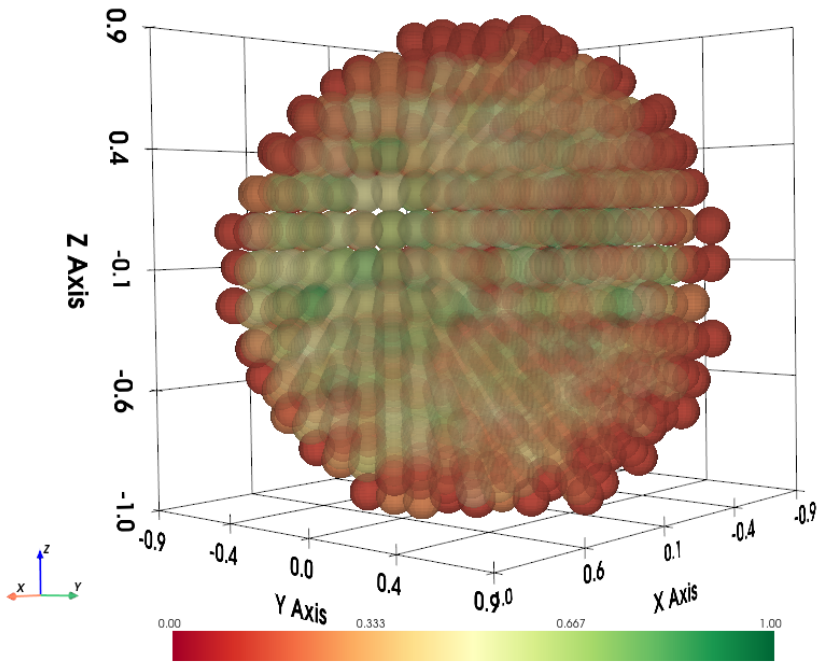

Figure 3.1: *Voxels colored by inverse reachability index*

B. Poses Map
This map stores the specific base pose vectors within their corresponding voxels. Using the XYZ coordinates of the inverted vector, the corresponding voxel index is found, and the position vector is stored in the poses map using this voxel index. **Figure** 3.2 shows the spatial distribution of these base poses throughout the workspace, illustrating the density and spread of feasible base locations.

C. Manipulability and Joint Solution Maps
Since the inversion only changes the reference frame (relative to the end-effector), the manipulability values and joint configurations remain unchanged. These values are simply mapped to the corresponding voxels in the new grid. The new IRM inverts the position and orientation vector to get the base pose relative to the end-effector, but the manipulability values and joint configurations themselves do not change—they are just mapped to the corresponding voxel in the target-centric coordinate system.

3.1.3 The Occupation Map (Whole-Body Awareness)

A critical contribution of the iHDRM is the Occupation Map. The framework iterates through every valid joint configuration and computes the robot's forward kinematics with the base placed

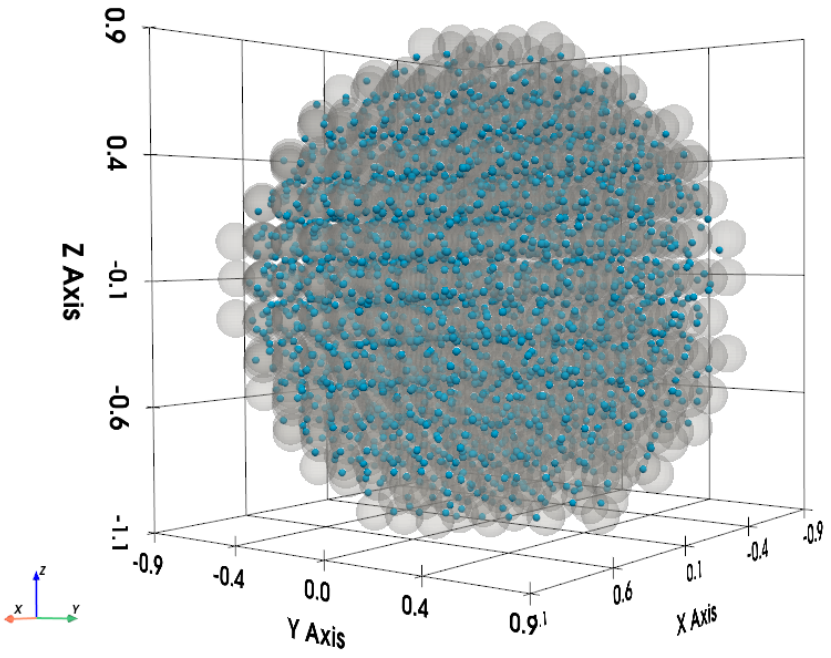

Figure 3.2: *Distribution of base poses in the workspace*

at each inverted position. This produces a point cloud representing the robot's entire body in the world frame.

3.1.3.1 Voxelization

By iterating through each joint configuration of the robot and applying forward kinematics with the robot base at the inverted position vector, the robot's occupied points in the world frame are computed based on the kinematics. The robot's body is represented as a point cloud in the world frame. The indices of all voxels occupied by this point cloud are identified.

Figure 3.3 demonstrates a specific robot configuration with its occupied voxels highlighted. The blue robot links are shown alongside the voxel grid cells that intersect with the robot's swept volume, illustrating how the robot's physical presence is mapped to discrete workspace regions

3.1.3.2 Storage

The joint configuration index is stored in every voxel it occupies. This allows for rapid reverse-lookup of which configurations collide with a specific region of space.

3.2 Phase B: Online Query and Collision Avoidance

During the query phase, the framework determines valid base locations by checking for collisions against the environment and verifying kinematic feasibility.

3.2.1 Collision Identification

Collision voxels are identified by mapping the point clouds of environmental obstacles to the grid's voxel indices. By iterating through the point clouds of obstacles in the environment and

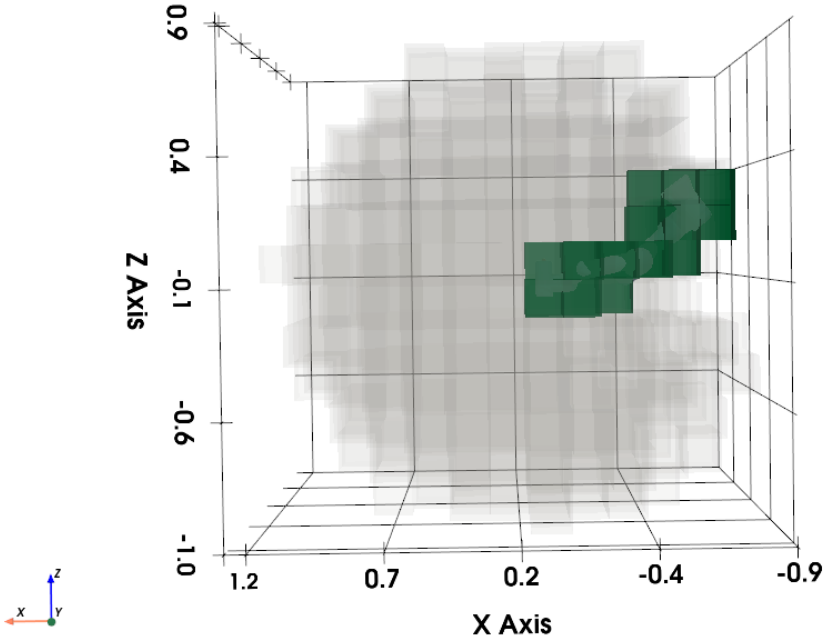

Figure 3.3: *Example robot configuration showing occupied voxels*

finding the mapping of their coordinates to voxel indices, these voxels are considered collision voxels.

Figure 3.4 illustrates this process: the highlighted green voxel represents a workspace cell identified as occupied by an obstacle. These collision voxel indices form the basis for querying the Occupation Map to identify which robot configurations would intersect with the environment.

3.2.1.1 Filtering Configurations and Safety Check

The framework queries the Occupation Map using collision voxel indices to retrieve all robot configurations that intersect with obstacles. These 'colliding configurations' are excluded from the potential solution set. By excluding these configurations from the joint solution maps, we obtain the free configurations that are safe from colliding with the environment.

Figure 3.5 shows an example of an invalid configuration where the robot's links (blue) penetrate the environmental obstacle (brown box), demonstrating a collision scenario that would be detected and rejected by the Occupation Map filtering mechanism. In contrast, **Figure** 3.6 illustrates a collision-free configuration where the robot successfully clears the obstacle while reaching the target pose, representing a valid solution that passes the safety check.

3.2.2 Handling Trajectories: Multi-Target Search and Optimization

Multi-target trajectory tasks—where the end-effector must visit a sequence of poses $T = \{T_1, T_2, \ldots, T_N\}$—represent a critical challenge for mobile manipulator base placement. Unlike single-pose queries, trajectory execution requires identifying base locations that enable collision-free manipulation across **all** trajectory points simultaneously. The iHDRM framework addresses

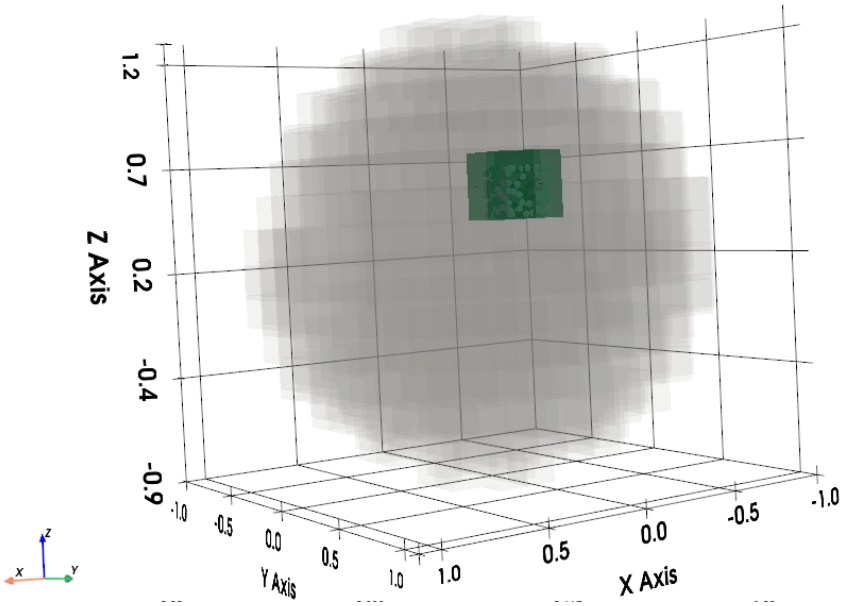

Figure 3.4: *Visualization of collision and occupied voxels*

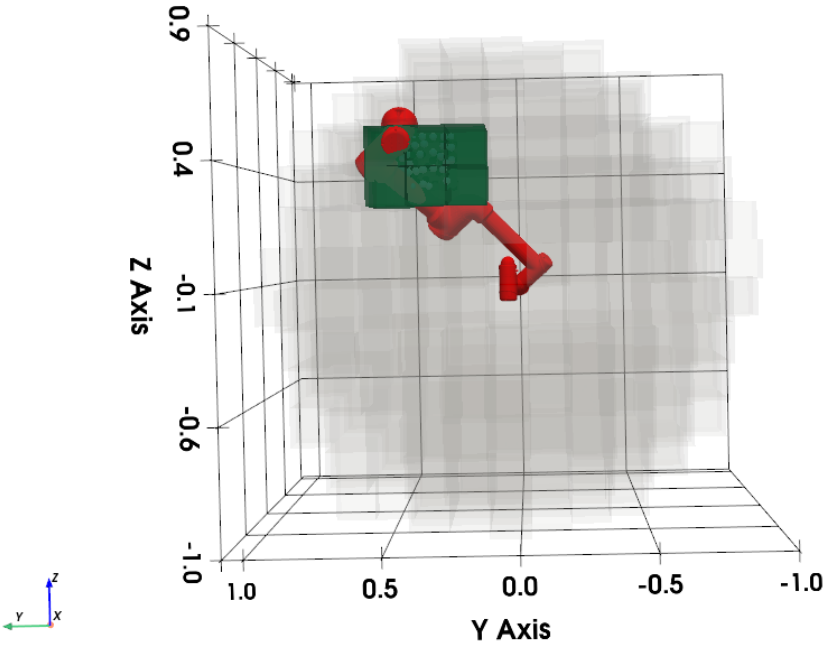

Figure 3.5: *Example of robot configuration in collision with environment*

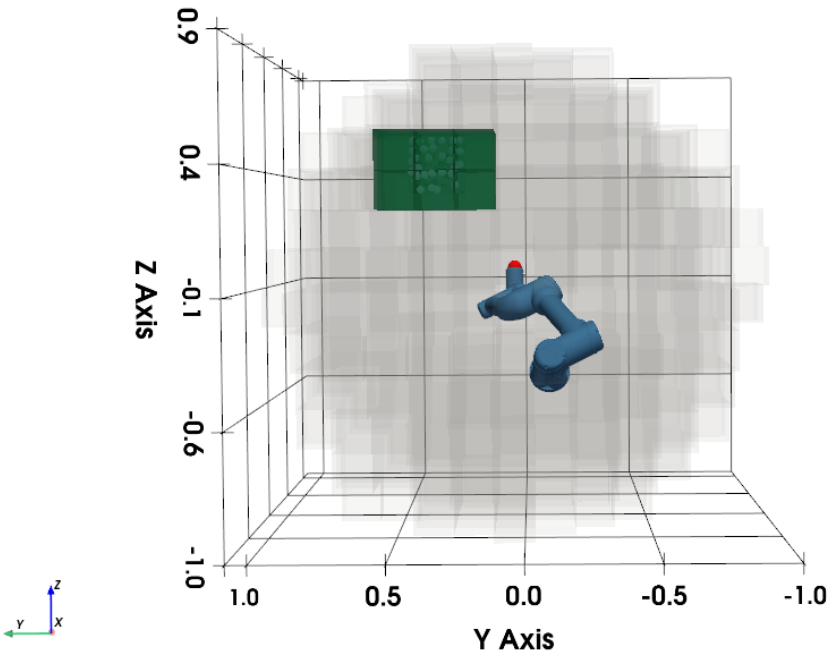

Figure 3.6: *Example of collision-free robot configuration*

this through a systematic set-based approach: computing candidate base pose sets for each target, intersecting these sets to find common feasible locations, filtering by collision constraints, and selecting the optimal pose through quality aggregation.

3.2.2.1 Set-Based Formulation of Multi-Target Reachability

Given a trajectory $T = \{T_1, T_2, \ldots, T_N\}$ of N target poses in $SE(3)$, the multi-target base placement problem seeks a base pose $b \in SE(3)$ such that every target is reachable with high kinematic quality and collision-free configurations exist for the entire trajectory.

A. Per-Target Base Pose Sets

For each target pose $T_i \in T$, the IRM is transformed to align its origin with T_i, the new base poses are calculated:

$$b = T_{\text{world}\to\text{base\_new}} = T_{\text{world}\to\text{target}} \times T_{\text{target}\to\text{base}}$$

enabling extraction of the **candidate base pose set** B_i. This set contains all base locations from which the end-effector can reach T_i:

$$B_i = \{b \in SE(3) \mid \exists q \in \mathbb{R}^n : FK(b, q) = T_i\}$$

where $FK(b, q)$ denotes forward kinematics mapping base pose b and joint configuration q to end-effector pose, and n is the number of robot joints ($n = 6$ for the UR5).

B. Feasible Common Base Pose Set

The set of base poses from which **all** targets are reachable is obtained through **set intersection**:

$$B_{\text{common}} = \bigcap_{i=1}^{N} B_i = B_1 \cap B_2 \cap B_3 \cap \cdots \cap B_N$$

This intersection is computed iteratively:

$$B_{\text{common}}^{(0)} = B_1$$
$$B_{\text{common}}^{(i)} = B_{\text{common}}^{(i-1)} \cap B_{i+1}, \quad \text{for } i = 1, 2, \ldots, N-1$$

After $N-1$ intersection operations, $B_{\text{common}}^{(N-1)} = B_{\text{common}}$ contains only base poses enabling reach to every target.

3.2.2.2 Configuration-Level Collision Filtering

Membership in B_{common} guarantees kinematic reachability but not collision-free execution. For each base pose $b \in B_{\text{common}}$ and each target T_i, the framework must verify that at least one of the robot's multiple IK solutions (up to 8 for the UR5) is collision-free.

A. Configuration Sets for Each Target
All relevant obstacle points are transformed into the new Target Frame (instead of transforming the iHDRM) to keep the same voxel indexing strategy. The transformation is calculated using the following formula:

$$P_{\text{target}} = (T_{\text{world}\to\text{target}})^{-1} \times P_{\text{world}}$$

These transformed P_{target} points are converted into voxel indices (i, j, k) in the Target-Centric Grid, and these voxels are considered collision voxels. **Figure** 3.7 demonstrates the result of this transformation, showing the collision voxels (representing obstacles in the target-centric frame), the original IRM (black) centered at the origin and the shifted IRM (red) centered at a specific target pose in the workspace. The base pose distributions shift spatially while maintaining their relative geometric relationships, enabling efficient collision checking.

For a given base pose b and target T_i, let $Q_i(b)$ denote the set of valid IK solutions:

$$Q_i(b) = \{q_1, q_2, \ldots, q_k\} \quad \text{where } k \leq 8$$

The collision-free configuration subset $Q_i^{\text{free}}(b) \subseteq Q_i(b)$ is obtained by evaluating each configuration against the Occupation Map:

$$Q_i^{\text{free}}(b) = \{q \in Q_i(b) \mid \text{Occupied}(q) \cap \text{Obstacles} = \emptyset\}$$

where $\text{Occupied}(q)$ denotes the set of voxels occupied by the robot in configuration q, and Obstacles represents environment obstacle voxels.

B. Valid Base Pose Set After Collision Filtering
A base pose b is **trajectory-valid** if and only if collision-free configurations exist for every target:

$$B_{\text{valid}} = \{b \in B_{\text{common}} \mid \forall i \in \{1, \ldots, N\} : Q_i^{\text{free}}(b) \neq \emptyset\}$$

This ensures that the selected base pose enables collision-free trajectory execution. If $Q_i^{\text{free}}(b) = \emptyset$ for any target T_i, then b is eliminated from consideration, as no collision-free path through that target exists.

3.2.2.3 Quality-Based Optimal Base Selection

Among the trajectory-valid base poses in B_{valid}, the framework selects the optimal pose through manipulability-based quality aggregation.

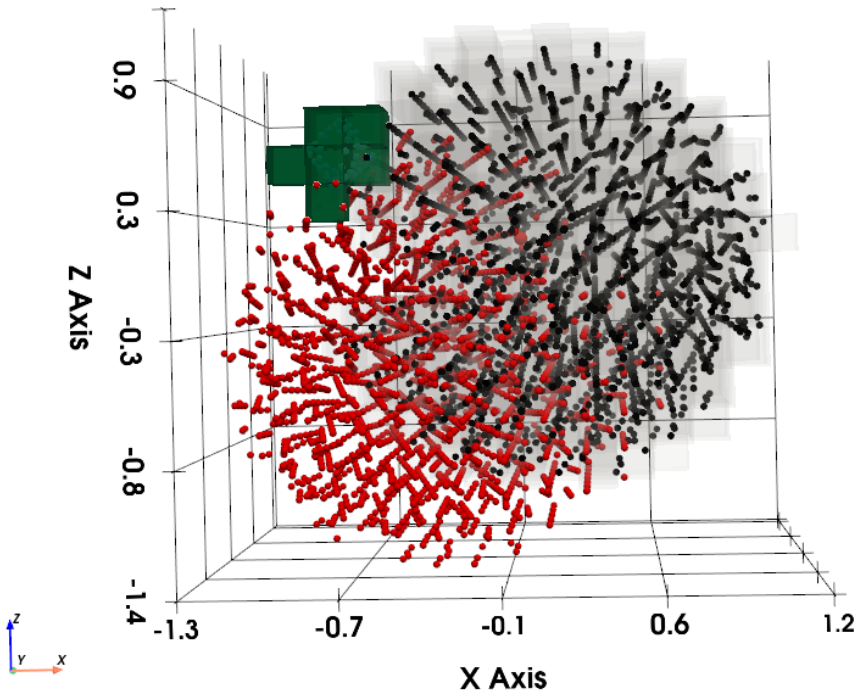

Figure 3.7: *Transformed collision voxels with occupied voxels and IRM poses centered at new target (red) and IRM poses centered at origin (black)*

A. Target-Wise Maximum Manipulability
For each base pose $b \in B_{\text{valid}}$ and each target T_i, the maximum manipulability among all collision-free configurations is computed:

$$w_{\max}(b, T_i) = \max_{q \in Q_i^{\text{free}}(b)} w(q)$$

where $w(q) = \sqrt{\det(J(q)J^T(q))}$ is the Yoshikawa manipulability index for configuration q. This measure identifies the best configuration for each target from a kinematic quality perspective.

B. Trajectory-Wise Mean Quality
To assess a base pose's suitability across the entire trajectory, the framework computes the mean of maximum manipulabilities:

$$\text{Quality}(b) = \frac{1}{N} \sum_{i=1}^{N} w_{\max}(b, T_i)$$

This aggregation strategy ensures that the selected base provides **consistently high** kinematic flexibility across all trajectory points, rather than favoring poses that excel at some targets while performing poorly at others.

C. Optimal Base Pose
The optimal base pose $b_{\text{optimal}} \in B_{\text{valid}}$ is the location maximizing trajectory-wise quality:

$$b_{\text{optimal}} = \arg \max_{b \in B_{\text{valid}}} \text{Quality}(b)$$

This selection guarantees:

1. **Complete trajectory reachability:** All targets $T_i \in T$ are kinematically reachable.

2. **Collision-free execution:** At least one collision-free configuration exists for every target.

3. **Kinematic quality optimization:** The base location provides maximum average manipulability across the trajectory.

4. **Singularity avoidance:** High manipulability values inherently steer away from singular configurations.

3.2.2.4 Computational Implementation

A. Transformation Back to Initial Frame
For computational efficiency, candidate base poses from B_{common} must be transformed back to the initial iHDRM frame (centered at the origin) to query stored manipulability values and joint configurations. For each target T_i, base poses are transformed via:

$$T_{\text{target}\to\text{base}} = (T_{\text{world}\to\text{target}})^{-1} \times T_{\text{world}\to\text{base\_new}}$$

This inverse transformation maps base poses expressed in the world frame back to the target-centered frame, enabling index-based lookup in the precomputed iHDRM data structures. **Critically, joint configurations and manipulability values remain invariant** under these coordinate transformations, as they represent intrinsic geometric properties of robot postures independent of reference frame choice.

B. Progressive Set Intersection Algorithm
The iterative intersection algorithm progressively filters the candidate set:

Algorithm 1 Multi-Target Base Pose Selection

Input: Trajectory $T = \{T_1, T_2, \ldots, T_N\}$, iHDRM, Obstacles
Output: b_{optimal}

Initialize $B_{\text{common}} \leftarrow \emptyset$
for $i = 1$ **to** N **do**
    Transform iHDRM to target pose T_i
    Extract candidate base pose set B_i
    if $i = 1$ **then**
        $B_{\text{common}} \leftarrow B_1$
    else
        $B_{\text{common}} \leftarrow B_{\text{common}} \cap B_i$
    end if
    if $B_{\text{common}} = \emptyset$ **then**
        return FAILURE (no common base exists)
    end if
end for
$B_{\text{valid}} \leftarrow \emptyset$
for each $b \in B_{\text{common}}$ **do**
    collision_free $\leftarrow$ true
    for $i = 1$ **to** N **do**
        Transform b back to initial frame for T_i
        Retrieve $Q_i(b)$ from iHDRM
        Compute $Q_i^{\text{free}}(b)$ via Occupation Map
        if $Q_i^{\text{free}}(b) = \emptyset$ **then**
            collision_free $\leftarrow$ false
            break
        end if
    end for
    if collision_free **then**
        $B_{\text{valid}} \leftarrow B_{\text{valid}} \cup \{b\}$
    end if
end for
if $B_{\text{valid}} = \emptyset$ **then**
    return FAILURE (no collision-free base exists)
end if
for each $b \in B_{\text{valid}}$ **do**
    quality_sum $\leftarrow 0$
    for $i = 1$ **to** N **do**
        Compute $w_{\max}(b, T_i)$ from $Q_i^{\text{free}}(b)$
        quality_sum $\leftarrow$ quality_sum $+ w_{\max}(b, T_i)$
    end for
    Quality$(b) \leftarrow$ quality_sum$/N$
end for
$b_{\text{optimal}} \leftarrow \arg\max_{b \in B_{\text{valid}}} \text{Quality}(b)$
return b_{optimal}

3.2.2.5 Theoretical Properties and Guarantees

The set-based multi-target approach provides several theoretical guarantees:

- **Completeness:** If a feasible collision-free base pose exists for the trajectory, the algorithm will find it (provided sufficient discretization resolution).

- **Correctness:** The returned base pose b_{optimal} is guaranteed to enable collision-free reach to all trajectory targets.
- **Optimality:** Among all valid base poses in the discrete representation, b_{optimal} maximizes average manipulability across the trajectory.
- **Efficiency:** Precomputed occupation data enables collision filtering in $O(|\text{Obstacles}| \times |Q|)$ time per base pose, where $|Q| \leq 8$ for the UR5, avoiding expensive geometric collision checks.

3.3 Summary

The iHDRM framework provides a comprehensive solution for mobile manipulator base placement by combining offline construction of detailed inverse reachability and occupation maps with efficient online querying for collision-free configurations. The framework's ability to handle multi-target trajectories while maintaining whole-body collision awareness and optimizing for manipulability makes it particularly suitable for complex manipulation tasks in cluttered environments.

Chapter 4

Tools for RM and iHDRM Construction

4.1 Hardware Acceleration Through GPU Computing

The construction of high-resolution Reachability Maps and Inverse Hybrid Dynamic Reachability Maps involves evaluating billions of independent mathematical operations across discretized workspace voxels, sampled orientations, and robot configurations. On conventional CPU-based architectures, this computational burden can extend to days or weeks, rendering iterative development and high-resolution analysis impractical. The framework addresses this fundamental computational bottleneck through hardware acceleration using Graphics Processing Units (GPUs), exploiting the inherently parallel structure of reachability computation to achieve speedups of two to three orders of magnitude.

4.1.1 The Computational Challenge: Dimensionality and Scale

Reachability map generation constitutes a **brute-force spatial discretization problem** where computational complexity scales rapidly with workspace resolution, orientation sampling density, and geometric fidelity.

4.1.1.1 Workspace Discretization Layer

The continuous 3D workspace is subdivided into discrete voxels. For a cubic workspace of dimension L with voxel resolution r, the number of voxels scales as:

$$N_{\text{voxels}} = \left(\frac{L}{r}\right)^3$$

For example, a $2\,\text{m} \times 2\,\text{m} \times 2\,\text{m}$ workspace with $2\,\text{cm}$ resolution yields 1,000,000 voxels.

4.1.1.2 Orientation Sampling Layer

Within each voxel, 50–200 uniformly distributed poses are sampled on an inscribed sphere. With 100 sphere points per voxel: $1{,}000{,}000 \times 100 = 100{,}000{,}000$ target poses requiring IK evaluation.

4.1.1.3 Inverse Kinematics Solution Layer

The UR5 produces up to 8 distinct solutions per pose. Total configurations: $100\text{M} \times 8 = 800\text{M}$ configurations, each requiring manipulability computation, forward kinematics, and SDF queries.

4.1.1.4 Per-Configuration Operations

Each configuration requires:

1. **Manipulability Computation:** Jacobian matrix (6×6), matrix multiplication, determinant, square root—hundreds of floating-point operations
2. **Forward Kinematics:** Multiple 4×4 matrix multiplications with trigonometric evaluations
3. **SDF Queries:** With 15 basis functions per dimension ($15^3 = 3{,}375$ coefficients), checking 6 links against 100 obstacle points yields $\sim$2 million operations per configuration
4. **Self-Collision Checking:** Additional pairwise SDF queries

4.1.1.5 Overall Computational Burden

Total: 800M configurations $\times$ $\sim$2M operations = 1.6 **trillion floating-point operations.**

4.1.1.6 CPU Performance Limitations

A 16-core CPU achieving 30 GFLOPS effective throughput requires:

$$\frac{1.6 \times 10^{12}}{30 \times 10^{9}} \approx 53{,}000 \text{ seconds} \approx \mathbf{14.7 \ \ hours}$$

Realistic implementations often extend this to **several days**, making iterative development impractical.

4.1.2 GPU Architecture: Parallel Computing at Scale

4.1.2.1 SIMD Architecture

GPUs implement SIMD (Single Instruction, Multiple Data): a single instruction broadcasts to thousands of cores executing simultaneously on different data. Modern GPUs have 80–100 Streaming Multiprocessors with 64–128 cores each, yielding 5,000–10,000 cores operating in parallel.

4.1.3 Reachability as an Embarrassingly Parallel Problem

Reachability map generation exhibits **embarrassing parallelism**—operations are independent and require minimal inter-process communication, ideally suited for GPU acceleration.

4.1.3.1 Data Independence

Computing IK, manipulability, or collision for one pose is completely independent of any other. This eliminates synchronization overhead—each GPU thread operates autonomously.

4.1.3.2 Uniform Computation Pattern

All poses undergo identical steps: IK $\rightarrow$ validation $\rightarrow$ Jacobian $\rightarrow$ manipulability $\rightarrow$ forward kinematics $\rightarrow$ SDF $\rightarrow$ storage. This uniformity aligns perfectly with GPU SIMD architecture.

4.1.4 GPU Implementation Strategy

4.1.4.1 Batched Processing

1. Batch Construction: Organize millions of poses into large arrays
2. Memory Transfer: Bulk transfer to GPU VRAM
3. Parallel Execution: Process 10,000–50,000 poses concurrently
4. Result Retrieval: Return only valid, filtered results

4.1.4.2 GPU Kernel Functions

- Kernel 1 – IK: Each thread computes up to 8 configurations for one pose
- Kernel 2 – Manipulability: Each thread processes one configuration
- Kernel 3 – Collision: Each thread handles configuration × link × obstacle point

4.1.4.3 Static Data Management

Load once into GPU VRAM:

- Robot kinematic parameters
- Link geometries
- SDF Bernstein coefficients (15^3 per link)
- Obstacle point clouds

Eliminates repeated CPU-GPU transfers.

4.1.5 Achieved Performance Gains

4.1.5.1 Computational Speedup

Modern high-end GPU (RTX 4090 / A100):

- Peak: 40–80 TFLOPS
- Effective sustained: 10–20 TFLOPS
- Speedup vs CPU: 300×–600×

GPU Time:

$$\frac{1.6 \times 10^{12}}{15 \times 10^{12}} \approx \mathbf{1.8 \ \ minutes}$$

Reduction from 14.7 hours (CPU) to 1.8 minutes (GPU) — speedup of 490×.

4.1.5.2 Practical Impact

- Rapid Prototyping: Test multiple parameters in minutes
- High-Resolution Maps: 5 mm voxels remain feasible (minutes-hours vs weeks)
- Interactive Debugging: Regenerate maps without multi-day delays
- Multi-Robot Support: Efficient comparative studies

4.1.6 Summary

GPU-based hardware acceleration addresses the fundamental computational bottleneck by exploiting the embarrassingly parallel structure of reachability computation. Specialized kernels for IK, manipulability, and collision detection achieve 2-3 orders of magnitude speedup, transforming multi-day batch processing into near-interactive computation.

4.2 Inverse Kinematics (IK)

Inverse Kinematics (IK) serves as a fundamental computational tool in the construction of both the Reachability Map (RM) and the Inverse Hybrid Dynamic Reachability Map (iHDRM). It provides the essential bridge between task-space specifications and joint-space control, enabling the determination of whether spatial locations are kinematically feasible and, if so, which robot configurations can achieve them.

4.2.1 The Necessity of IK: Bridging Cartesian and Joint Space

In robotic manipulation and motion planning, a fundamental disconnect exists between how tasks are specified and how robots are controlled. Tasks are naturally defined in **Cartesian space**—a 6-dimensional representation comprising position coordinates (X, Y, Z) and orientation angles (Roll, Pitch, Yaw). However, robot control occurs in **joint space**, where motion is commanded through specific angular values for each actuated joint, collectively represented as the joint configuration vector q.

Inverse Kinematics is the mathematical transformation that maps from Cartesian space to joint space, solving for the joint angles required to position the end-effector at a desired pose:

$$q = \text{IK}(x, y, z, \text{roll}, \text{pitch}, \text{yaw})$$

4.2.1.1 Role in Reachability Map Generation

Within the context of RM construction, IK serves as the primary feasibility test for each sampled pose on the voxel's inscribed sphere. The process follows this logic:

1. A candidate pose is generated through spherical sampling within a voxel.
2. The IK solver attempts to compute valid joint configurations for this pose.
3. The pose is marked as **reachable** if and only if at least one valid IK solution exists.
4. The pose is marked as **unreachable** if no valid solution can be found (due to joint limits, singularities, or geometric constraints).

Without IK, it would be impossible to determine whether a spatial voxel is actually kinematically accessible by the robot's end-effector. Furthermore, IK enables the evaluation of reach quality through manipulability metrics, as these depend on the specific joint configurations used to achieve each pose. The offline computation of IK solutions for millions of sampled poses forms the foundation of the reachability analysis.

4.2.2 Solution Approaches: Analytical vs. Numerical Methods

IK solvers are broadly classified into two categories, each with distinct characteristics and computational properties:

4.2.2.1 Numerical Solvers

Numerical methods, such as Jacobian pseudo-inverse techniques and gradient-based optimization, employ iterative approximation to converge toward a solution. Starting from an initial joint configuration, these methods compute incremental updates using the relationship:

$$\Delta q = J^{\dagger}(q)\Delta x$$

where $J^{\dagger}$ represents the Moore-Penrose pseudo-inverse of the Jacobian matrix.

Characteristics:

- **Advantages:** Highly adaptable to complex kinematic structures, including redundant manipulators with more degrees of freedom than required; can incorporate constraints and optimization objectives.
- **Disadvantages:** Computationally expensive due to iterative nature; convergence not guaranteed; typically finds only a single solution per initial guess; sensitive to initial configuration and may fail to find solutions even when they exist.

4.2.2.2 Analytical (Closed-Form) Solvers

Analytical methods derive exact algebraic and geometric equations that directly compute joint angles from the desired Cartesian pose. These solutions exploit the specific geometric structure of the robot's kinematic chain to decompose the problem into manageable subproblems.

Characteristics:

- **Advantages:** Orders of magnitude faster than numerical methods (typically microseconds vs. milliseconds); guaranteed to find all possible solutions simultaneously; deterministic behavior with no dependency on initial conditions.
- **Disadvantages:** Requires specific kinematic structures (e.g., spherical wrist, intersecting joint axes); not applicable to all robot designs; derivation can be mathematically complex.

4.2.2.3 Selection for RM and iHDRM Construction

The offline generation of RM and iHDRM requires evaluating IK solutions for millions of sampled poses across the discretized workspace. Given this computational scale:

$$\text{Number of IK calls} \approx \text{Number of voxels} \times \text{Points per sphere} \times \text{Configuration attempts}$$

For a typical workspace with 10,000 voxels, 100 points per sphere, and checking for up to 8 configurations, this results in approximately 8 million IK evaluations. With numerical methods requiring 1–10 milliseconds per solution, total computation would span 2–22 hours. In contrast, analytical methods completing in 10–100 microseconds reduce this to 1–2 minutes.

Therefore, an **analytical solver is strictly required** to maintain feasible computation times for RM and iHDRM construction. This necessity influences robot selection for the framework, favoring designs with kinematic structures amenable to closed-form solutions.

4.2.3 The UR5 Analytical Solution: Multiple Configuration Theorem

The Universal Robots UR5 is a 6-degree-of-freedom (6-DOF) serial manipulator with a kinematic structure specifically designed to permit complete closed-form IK solutions. The robot possesses a **spherical wrist** configuration, where the axes of the final three joints (wrist joints 4, 5, and 6) intersect at a common point. This geometric property enables analytical decomposition of the IK problem into position and orientation subproblems.

4.2.3.1 Emergence of Multiple Solutions

When the analytical IK solver computes joint angles for a single 6D target pose, the trigonometric equations arising from the UR5's kinematic structure yield **up to 8 distinct joint configurations** capable of positioning the end-effector at the exact same spatial pose. This multiplicity is not a mathematical artifact or solver ambiguity—it represents genuine kinematic redundancy in the robot's configuration space.

These 8 solutions arise from three independent geometric binary choices in the robot's posture, each corresponding to a fundamental branching in the kinematic chain:

1. **Shoulder Configuration (Left vs. Right):** The base revolute joint (Joint 1) can rotate to face the target directly, or rotate approximately 180° away, causing the arm to reach "backward" over the base. Both configurations can position the wrist at the same location but with the shoulder on opposite sides of the workspace.
2. **Elbow Configuration (Up vs. Down):** For a given distance between shoulder and wrist, the elbow joint (Joint 3) can bend upward (forming a convex arc resembling a mountain peak) or downward (forming a concave arc resembling a valley). Both configurations achieve the same wrist position but with dramatically different link orientations.
3. **Wrist Configuration (Flip vs. No-Flip):** The final three wrist joints can achieve the same end-effector orientation through two distinct combinations of joint angles, typically separated by approximately 180° in one or more wrist joints. This arises from the multiple ways to represent the same 3D orientation using three revolute joints.

Mathematically, these three independent binary choices produce:

$$2\ \textbf{(shoulder)} \times 2\ \textbf{(elbow)} \times 2\ \textbf{(wrist)} = 8\ \textbf{maximum theoretical solutions}$$

In practice, not all 8 solutions may be valid for every reachable pose, as some configurations may violate joint limits or result in kinematic singularities. However, the analytical solver efficiently evaluates all branches and returns only the feasible solutions.

4.2.3.2 Integration with the Capability Map

To accommodate this multiplicity of solutions, the Capability Map is initialized with robot-specific coefficients, including the maximum number of configuration solutions per pose (8 for the UR5) and the number of degrees of freedom. During RM construction, the Joint Solution Map stores all valid configurations for each reachable pose, maintaining separate entries for each of the up to 8 solutions.

4.2.4 Critical Role in the iHDRM Framework

The existence of multiple IK solutions is not merely a mathematical curiosity—it constitutes a **critical enabler** for collision-free motion planning within the iHDRM framework. While all 8 configurations place the end-effector at an identical target pose, the **physical volume occupied by the robot's body** differs dramatically among them.

4.2.4.1 Collision Diversity Across Configurations

Consider a scenario where the robot must reach a target point near a table obstacle:

- An "Elbow Down" configuration might cause the upper arm or forearm to sweep through the table surface, resulting in collision.

- An ”Elbow Up” configuration achieves the same end-effector pose but lifts the arm above the table, maintaining safe clearance.
- A ”Left Shoulder” configuration might position the base links closer to a wall, causing collision, while a ”Right Shoulder” configuration provides adequate separation.

The Occupation Map in the iHDRM framework explicitly captures these differences by storing which voxels are occupied by the robot's body for each of the 8 configurations independently. During online query, when collision voxels are identified from environmental obstacles, the framework can selectively exclude only those configurations that intersect with the obstacles, while preserving collision-free alternatives.

4.2.4.2 Enhanced Planning Flexibility

By storing up to 8 joint solutions per pose in the Joint Solution Map, the framework provides the online planner with multiple fallback options:

1. **Collision Avoidance:** If one configuration collides, alternative configurations reaching the same pose may remain collision-free.
2. **Quality Optimization:** Among collision-free configurations, the planner can select the one with highest manipulability or other quality metrics.

4.2.5 Summary

Inverse Kinematics serves as an indispensable computational tool in the construction of Reachability Maps and the iHDRM framework. The analytical IK solver for the UR5 robot provides both the computational efficiency required for offline map generation (through closed-form solutions) and the kinematic flexibility required for online collision-free planning (through multiple configuration solutions). The systematic storage of all valid IK solutions in the Joint Solution Map, coupled with whole-body occupation tracking in the Occupation Map, enables the framework to exploit this configurational diversity for robust, collision-aware mobile manipulator planning.

4.3 Kinematic Manipulability (The Yoshikawa Index)

Kinematic manipulability provides a quantitative measure of the robot's motion capability at a given configuration, complementing the binary reachability information provided by Inverse Kinematics. While IK determines whether a pose is kinematically feasible, manipulability assesses the quality of that feasibility, enabling the selection of superior configurations from among multiple valid solutions. This metric plays a crucial role in both offline RM construction and online base placement optimization within the iHDRM framework.

4.3.1 The Necessity of Quality Metrics: Beyond Binary Reachability

Inverse Kinematics provides a fundamental answer to the question: ”Can the robot reach this target pose?” However, this binary determination provides no insight into **how well** the robot can perform tasks at that pose. Consider two valid configurations that both place the end-effector at an identical target:

Configuration A: The robot is fully extended with the arm nearly straight, joints approaching their limits, positioned close to a kinematic singularity. Minimal ability to apply forces, poor motion control, and limited freedom for adjustments.

Configuration B: The robot maintains moderate joint angles with the elbow comfortably bent, well-separated from singularities. Excellent force application in all directions, smooth motion control, and substantial freedom for local adjustments.

Both configurations satisfy the IK constraint, yet Configuration A represents a **poor solution** while Configuration B represents an **optimal solution**. A kinematic quality metric enables systematic differentiation, supporting solution ranking, base pose optimization, singularity avoidance, and task performance prediction.

4.3.2 The Yoshikawa Manipulability Measure

The framework employs the **Yoshikawa manipulability index** [16] as its kinematic quality metric, a standard tool in robotics for quantifying configuration quality.

4.3.2.1 Mathematical Definition

The Yoshikawa index w [16] is derived from the robot's Jacobian matrix $J(q)$, which relates joint velocities to end-effector velocities: $\dot{x} = J(q)\dot{q}$. The index is defined as:

$$w = \sqrt{\det(J(q)J^T(q))}$$

4.3.2.2 Geometric Interpretation

The Yoshikawa index equals the volume of the **manipulability ellipsoid**—the set of end-effector velocities achievable with unit joint velocities. Physical significance:

- **Large, spherical ellipsoid:** Uniform rapid motion in all Cartesian directions (isotropic capabilities)
- **Small ellipsoid:** Reduced motion capability overall
- **Elongated ellipsoid:** Anisotropic—moves well in some directions but poorly in others, signaling proximity to singularities

4.3.2.3 Singularity Avoidance

A kinematic singularity occurs when the robot loses one or more degrees of freedom. As the robot approaches a singularity, the ellipsoid flattens, and $w \to 0$. By maximizing w, the framework inherently steers away from dangerous singular configurations, simultaneously improving motion capability and robustness.

4.3.3 Integration with the Manipulability Map

4.3.3.1 Computation Workflow

For each voxel during RM construction:

1. Pose sampling on inscribed sphere
2. IK computation for all valid configurations (up to 8 for UR5)
3. Jacobian evaluation: compute $J(q)$ for each configuration
4. Manipulability computation: $w = \sqrt{\det(J(q)J^T(q))}$
5. Storage in Manipulability Map indexed by `[voxel][pose][configuration]`

4.3.3.2 Map Structure

The Manipulability Map maintains parallel organization to the Joint Solution Map:

- **Indexing:** `[voxel_index][pose_index][configuration_index]`
- **Content:** Single scalar w value per entry
- **Correspondence:** Direct mapping to Joint Solution Map entries
- **Completeness:** Every valid IK solution has an associated manipulability value

This pre-computation eliminates real-time Jacobian computation during online planning.

4.3.4 Critical Role in the iHDRM Framework

4.3.4.1 Invariance During Kinematic Inversion

During iHDRM construction, position and orientation vectors are inverted to obtain base poses relative to the end-effector. However, **joint configurations themselves do not change**—they represent the same physical robot posture. Since w depends only on joint configuration q, **manipulability values remain identical** and are simply remapped to corresponding voxels in the base-centric coordinate system. This invariance allows the framework to inherit all quality information without recomputation.

4.3.4.2 Multi-Target Trajectory Evaluation

For trajectories with multiple sequential targets, the framework identifies base locations providing high-quality reach to **all** targets through systematic aggregation:

For each candidate base pose b:

1. **Target-wise maximum:** For each target t, find maximum manipulability among all collision-free configurations: $w_{\max}(b, t)$
2. **Trajectory-wise mean:** Compute mean of maximum values across all targets:

$$\text{Quality}(b) = \frac{1}{N} \sum_{t} w_{\max}(b, t)$$

4.3.4.3 Optimal Base Pose Selection

The base pose with highest quality measure is selected:

$$b_{\text{optimal}} = \arg\max_{b} \text{Quality}(b)$$

This ensures the mobile manipulator is positioned for maximum kinematic flexibility, minimum proximity to singularities, superior force application and motion control, and enhanced robustness.

4.3.5 Summary

The Yoshikawa manipulability index transforms kinematic reachability analysis from binary feasibility checking into nuanced optimization. By quantifying configuration quality through the manipulability ellipsoid volume, the framework systematically prefers well-conditioned poses over poorly-conditioned ones, inherently avoiding singularities and enhancing task performance. Offline computation and storage of manipulability values enables efficient online base placement optimization through quality-based aggregation across trajectory targets. Combined with collision-awareness, manipulability-based selection ensures chosen base locations provide not merely feasible but optimal kinematic conditions for task execution.

4.4 Collision Detection and Signed Distance Functions (SDF)

Collision detection constitutes a fundamental requirement for safe motion planning. Within the RM and iHDRM frameworks, collision awareness must be integrated at two levels: self-collision detection to ensure the robot's links do not intersect each other, and environment collision detection to prevent contact with external obstacles. The framework requires a collision representation that balances geometric accuracy, computational efficiency, and mathematical properties conducive to optimization-based planning.

4.4.1 The Necessity of Spatial Safety Evaluation

For the RM and iHDRM to be practically viable, the framework must provide guarantees that identified configurations are feasible. This requirement manifests in two forms:

4.4.1.1 Self-Collision and Environment Collision

Self-collision occurs when the robot's own links intersect (e.g., forearm sweeping back into upper arm). Environment collision involves interference between robot links and external obstacles. The Occupation Map tracks which voxels are occupied by the robot's body for each configuration, enabling rapid identification of configurations that would collide with obstacles mapped to those same voxels.

4.4.1.2 Computational Requirements

The collision detection method must satisfy:

1. Computational efficiency: Millions of configurations evaluated during offline construction require microsecond-level collision tests
2. Geometric accuracy: Must capture 3D geometry accurately, not just simplified bounding volumes
3. Mathematical continuity: Should provide continuous gradients for optimization-based planning
4. Distance quantification: Beyond binary collision/no-collision, provide quantitative distance information

4.4.2 The Signed Distance Function (SDF)

The framework employs **Signed Distance Functions** (SDFs) [17] as the mathematical representation of spatial boundaries. The SDF provides a continuous scalar field encoding both distance and inside/outside information.

4.4.2.1 Mathematical Definition

For a closed surface S, the SDF assigns to any point $p \in \mathbb{R}^3$ the shortest Euclidean distance to that surface. The signed distance function $f(p)$ uses a sign convention:

$$f(p) = \begin{cases} +d(p, S) & \text{if } p \text{ is outside } S \\ 0 & \text{if } p \text{ is on } S \\ -d(p, S) & \text{if } p \text{ is inside } S \end{cases}$$

4.4.2.2 Collision Detection Logic

This scalar representation enables instant collision status determination:

- $f(p) > 0$: Point is safely outside (no collision). Magnitude indicates clearance distance.
- $f(p) = 0$: Point lies on surface boundary.
- $f(p) < 0$: Point is inside (collision detected). Magnitude indicates penetration depth.

For practical collision checking with safety margins: collision is declared if $f(p) \leq \varepsilon$, where ε is a small safety margin (e.g., 1–5 mm).

4.4.3 Application to Multi-Link Robot Kinematics

4.4.3.1 Link-Local SDF and Frame Transformation

Each robot link's geometry is represented by its own SDF $f_l(\cdot)$, pre-computed and stored in the link's local coordinate frame. Let $T_l^w(q) \in SE(3)$ denote the transformation matrix of link l's frame relative to the world frame, given configuration q.

To query the signed distance of world-frame point p^w relative to link l:

$$f_l(p^w, q) = f_l\left((T_l^w(q))^{-1} p^w\right)$$

4.4.3.2 Collision Map Storage

The Collision Map stores, for each joint configuration, a vector of SDF values—one minimum SDF per robot link. This enables:

- Quick identification of which links are in collision (negative SDF)
- Clearance metrics computation
- Visualization support

4.4.4 Summary

Collision detection through Signed Distance Functions provides the RM and iHDRM frameworks with a unified mathematical representation for spatial safety evaluation. The coordinate frame transformation approach enables whole-body collision checking with minimal overhead. This combination of geometric accuracy, computational efficiency, and mathematical smoothness positions SDF-based collision detection as an indispensable tool for comprehensive reachability and base placement frameworks.

Chapter 5

Experimental Validation and Results

5.1 Experimental Setup and Map Generation

This chapter presents the experimental validation of the Inverse Hybrid Dynamic Reachability Map (iHDRM) framework for mobile manipulator base placement. The experiments evaluate the framework's ability to identify collision-free, kinematically optimal base locations for a UR5 manipulator mounted on a mobile platform, executing multi-target trajectories in cluttered environments. The validation encompasses three critical aspects: offline iHDRM construction and spatial properties, online base placement for diverse trajectory shapes, and systematic analysis of collision avoidance behavior. The chapter concludes with methodological considerations addressing computational efficiency, geometric accuracy, and directions for future enhancement.

5.1.1 Hardware and Software Configuration

The experimental framework was implemented for a Universal Robots UR5 6-DOF manipulator with the following specifications:

- **Robot Model:** UR5 collaborative robot arm (850 mm reach)
- **Workspace Discretization:** 0.16 m voxel resolution (160 mm)
- **Configuration Space:** 6 revolute joints, up to 8 IK solutions per pose
- **Mobile Platform:** $SE(3)$ base mobility (translation + rotation)
- **Computation:** GPU-accelerated offline map construction

The 0.16 m resolution represents a practical balance between computational tractability and spatial precision, enabling comprehensive workspace coverage while maintaining manageable memory footprints for the occupation data structures.

5.1.2 Offline Reachability Map Construction

The foundational Reachability Map (RM) was generated offline through systematic workspace discretization. The continuous $SE(3)$ workspace was partitioned into a uniform voxel grid with 0.16 m edge length. For each voxel, a sphere was inscribed and sampled with uniformly distributed orientation frames. For each sampled pose, all analytical IK solutions (up to 8 for the UR5) were computed and validated against joint limits.
This offline phase produced four complementary data structures:

1. **Reachability Map:** Binary indicators of pose reachability per voxel.
2. **Joint Solution Map:** All valid joint configurations achieving each reachable pose.

3. **Manipulability Map:** Yoshikawa indices $w(q)$ for kinematic quality assessment.
4. **Collision Map:** SDF-based self-collision detection for each configuration.

5.1.3 Support Surface Modeling and Self-Collision Filtering

A critical challenge in mobile manipulation is distinguishing between legitimate collisions (robot links intersecting obstacles) and false-positive collisions (robot intersecting its own support table or floor). To address this without imposing online computational overhead, the support table was modeled as a **permanent, static kinematic link** attached to the robot base during offline generation.
During RM construction, any joint configuration causing intersection between robot links and this virtual table was eliminated by the offline collision checker. This preprocessing ensures that all stored configurations in the final database are:

- **Inherently free of self-collision**
- **Guaranteed to clear the support surface**
- **Valid for placement on any unconstrained $SE(3)$ position**

This approach transforms a potentially expensive online geometric query ("Does this configuration collide with the floor?") into a trivial binary lookup ("Is this configuration in the database?"), maintaining real-time responsiveness during online base placement.

5.1.4 Occupation Map Generation

For each validated, collision-free joint configuration $q \in Q$, forward kinematics was computed to determine the spatial positions of all robot links. Each link was represented as a dense point cloud, and these points were mapped to the 0.16 m voxel grid. For voxel v_i, the **Occupation List** O_i stores the indices of all configurations whose point clouds intersect that voxel:

$$O_i = \{q_j \mid \text{PointCloud}(q_j) \cap v_i \neq \emptyset\}$$

This reverse-lookup structure—mapping from voxels to configurations rather than configurations to voxels—is the cornerstone of the iHDRM's collision-checking efficiency. During online queries, identifying which configurations collide with environment obstacles reduces to set intersection between obstacle voxel indices and occupation lists, avoiding expensive per-configuration geometric computations.

5.2 Multi-Target Trajectory Experiments

5.2.1 Experimental Protocol and Trajectory Design

The online validation phase tested the framework's ability to find optimal base poses for multi-target trajectories in the presence of environmental obstacles. Five trajectory shapes were evaluated:

1. **Linear trajectory:** Straight-line path testing basic intersection logic.
2. **Diagonal trajectory:** Oblique path validating non-axis-aligned motion.
3. **Sinusoidal curve:** Smooth oscillating path simulating contour following.
4. **U-shaped trajectory:** 180° reversal testing workspace envelope limits.

5. **L-shaped (L2) trajectory:** 90° corner path representing pick-and-place tasks.

All trajectories were discretized based on the 0.16 m map resolution—continuous paths were sampled at 0.16 m spatial intervals, with target poses (red spheres in visualizations). To isolate the spatial intersection logic from orientational complexity, end-effector orientation was held **rigidly constant** throughout each trajectory, matching one of the pre-sampled orientations from the RM generation phase.

5.2.2 Environmental Obstacle Configuration

A single environmental obstacle—a rectangular box (brown in visualizations)—was positioned to create challenging collision scenarios. The obstacle was intentionally placed to:

- Intersect potential base locations for kinematically feasible but collision-prone placements
- Force the framework to select non-obvious base poses requiring careful collision filtering
- Validate the occupation list filtering by creating scenarios where naive base selection would fail

During online queries, the obstacle's voxel occupancy was computed through point cloud voxelization, providing the set of occupied voxels V_{obstacle} used for collision checking.

5.2.3 Results: Linear and Diagonal Trajectories

Initial experiments focused on straight-line trajectories to establish baseline performance. **Figure** 5.1 and **Figure** 5.2 demonstrate successful execution of linear and diagonal paths, respectively. In both cases:

- The algorithm successfully computed the intersection $B_{\text{common}} = B_1 \cap B_2 \cap \cdots \cap B_N$ across all N targets.
- The collision filtering eliminated base poses where any configuration $q \in Q_i(b)$ would intersect the brown box obstacle.
- The optimal base location b_{optimal} (represented by the spawned support table) was selected through manipulability-based quality maximization.
- The robot's kinematic chain (blue links) successfully navigates beneath the obstacle while maintaining reach to all targets.

The discrete target spheres align precisely with the 0.16 m grid, confirming proper trajectory discretization. The visualization shows the complete robot configuration at one trajectory point, with the continuous spline connecting targets indicating the planned end-effector path.

5.2.4 Results: Complex Articulated Trajectories

Following linear baselines, the framework was evaluated on complex, non-linear trajectories simulating realistic manipulation tasks. **Figure** 5.3 illustrates execution of a spatial sine wave, while **Figure** 5.4 and **Figure** 5.5 demonstrates an L-shaped and U-shaped paths.
Despite the substantial spatial translations required to follow these curves—with targets spanning multiple voxels in multiple directions—the framework successfully:

1. **Identified the intersection space:** For the sinusoidal trajectory with $N = 7$ targets discretized at 0.16 m intervals, the algorithm computed B_{common} by progressively filtering 7 candidate sets.

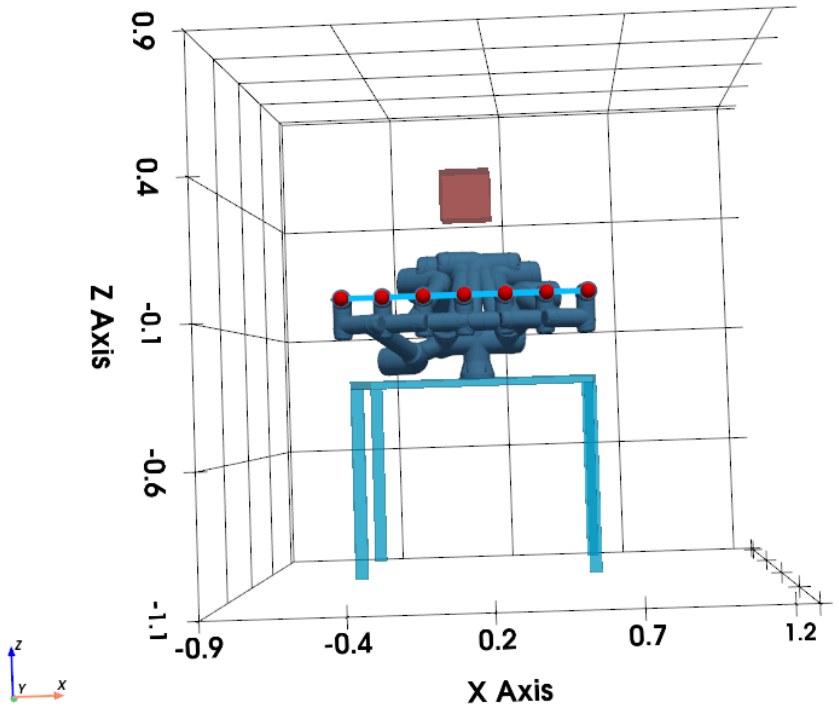

Figure 5.1: Linear trajectory execution with collision avoidance.

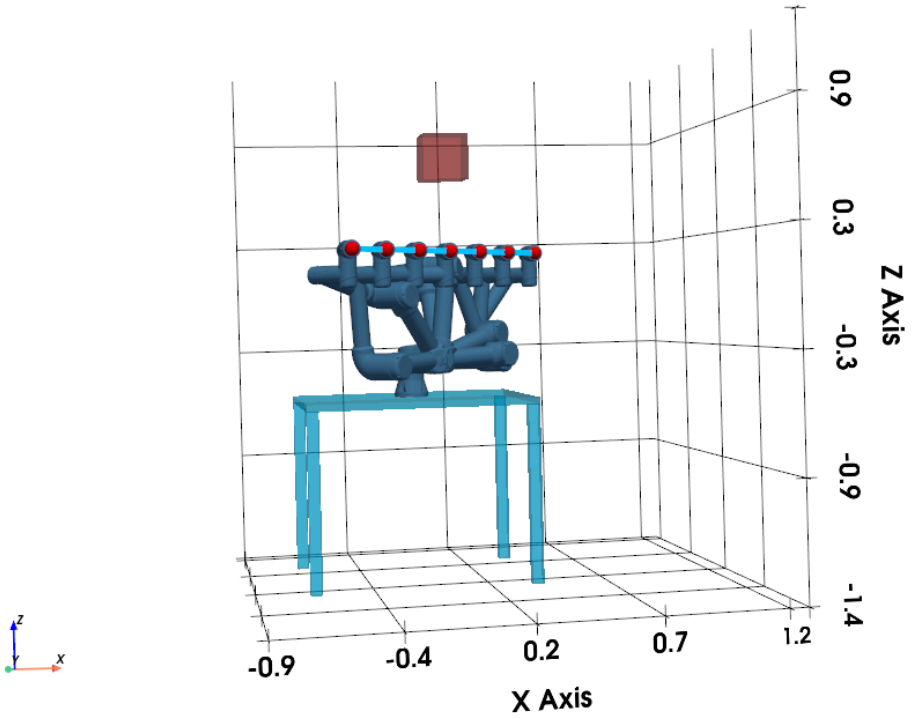

Figure 5.2: Diagonal trajectory execution with collision avoidance.

2. **Filtered collision-prone configurations:** The Occupation Map enabled rapid identification of configurations intersecting the obstacle, eliminating entire base pose candidates in $O(|V_{\text{obstacle}}|)$ time per pose.

3. **Optimized manipulability:** Among collision-free candidates, b_{optimal} was selected by maximizing $\text{Quality}(b) = \frac{1}{N}\sum_{i=1}^{N} w_{\max}(b, T_i)$, ensuring consistently high kinematic quality across all trajectory points.

The continuous splines connecting discrete targets highlight the intended end-effector path. The successful execution of these complex geometries validates the framework's capability to handle multi-dimensional spatial constraints while maintaining collision awareness and kinematic optimization.

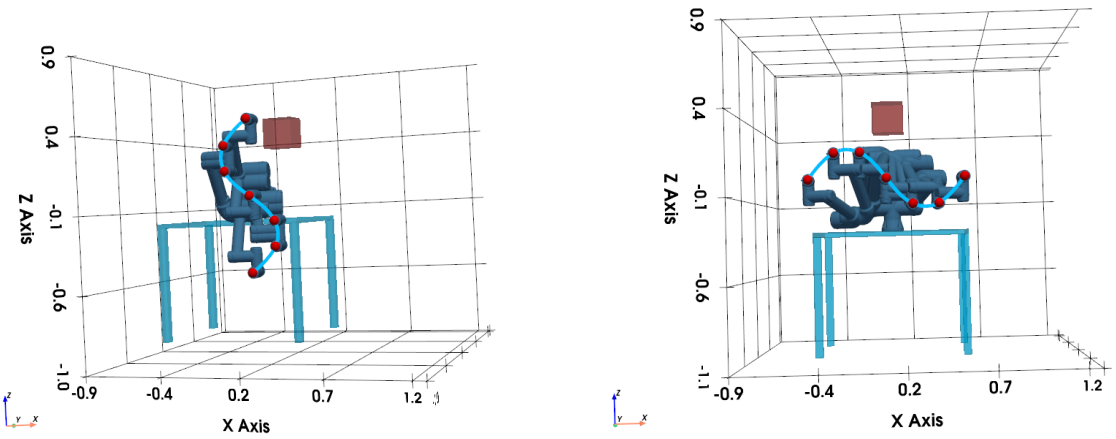

Figure 5.3: Sinusoidal trajectory execution.

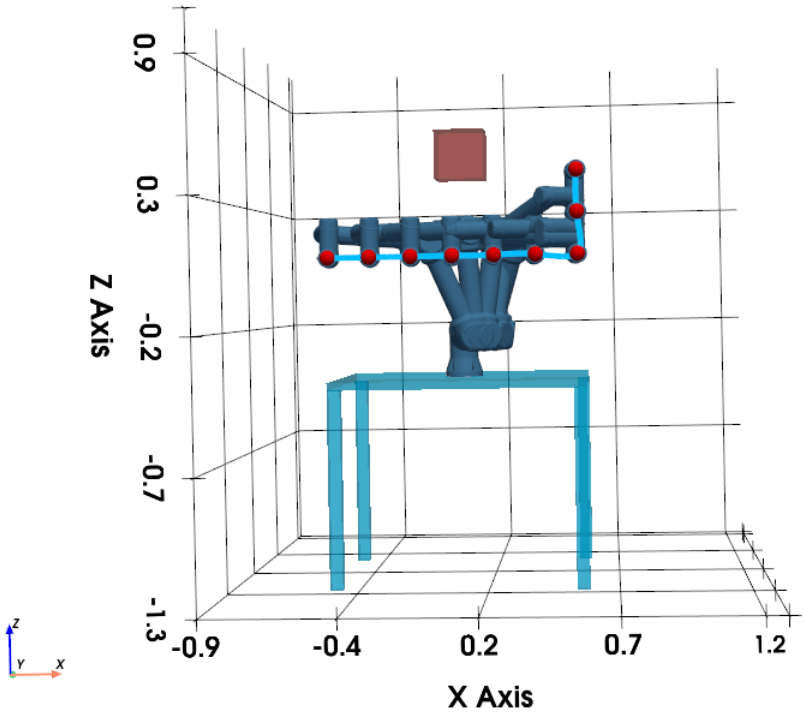

Figure 5.4: L-shaped trajectory execution.

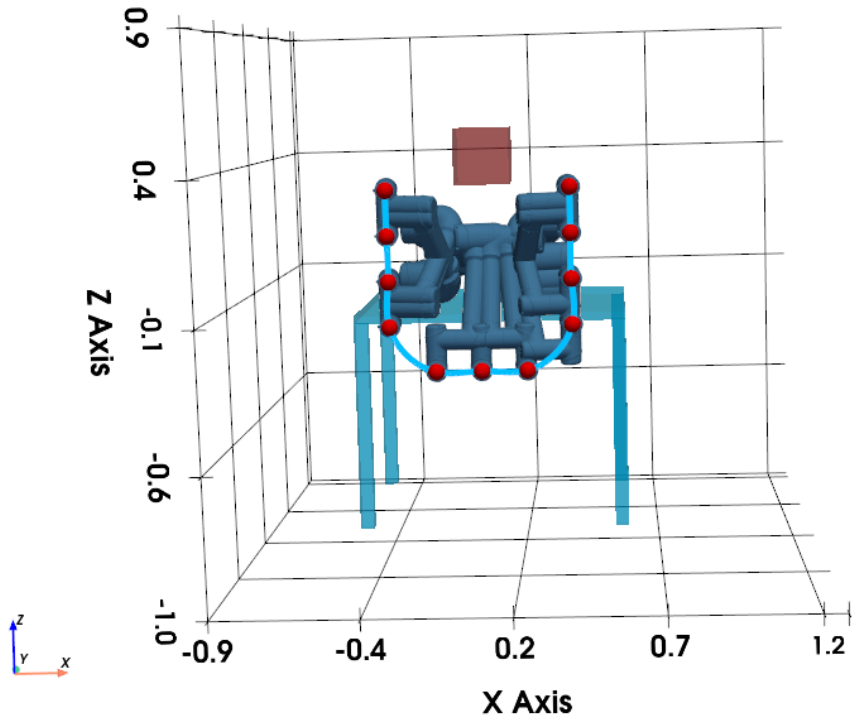

Figure 5.5: U-shaped trajectory execution.

5.2.5 Collision Avoidance Validation: Negative Cases

To validate the robustness of the collision-checking pipeline, the framework's output was compared against purely kinematic base placements—base poses selected solely for reachability without considering environmental obstacles. **Figure** 5.6, **Figure** 5.7 and **Figure** 5.8 illustrate these "invalid" scenarios for linear, L2, sinusoidal, diagonal and U trajectories.

In each invalid case, the visualization clearly shows:

- **Severe geometric intersections:** Robot links (blue) penetrating deeply into the obstacle box (brown).
- **Kinematic feasibility without collision awareness:** These base poses enable reach to all trajectory targets, demonstrating membership in B_{common}.
- **Occupation list detection:** The framework correctly identifies voxel overlap: $\text{Occupied}(q) \cap V_{\text{obstacle}} \neq \emptyset$ for multiple configurations.
- **Rejection mechanism:** These poses are excluded from B_{valid}, as $\exists i : Q_i^{\text{free}}(b) = \emptyset$.

This systematic comparison between "success" (**Figures** 5.1–5.4) and "invalid" (**Figure** 5.6, **Figure** 5.7 and **Figure** 5.8) cases demonstrates that the iHDRM framework is not simply finding any reachable base pose, but actively filtering to ensure collision-free execution. The occupation list mechanism successfully prevents selection of geometrically infeasible configurations that would fail during actual robot motion.

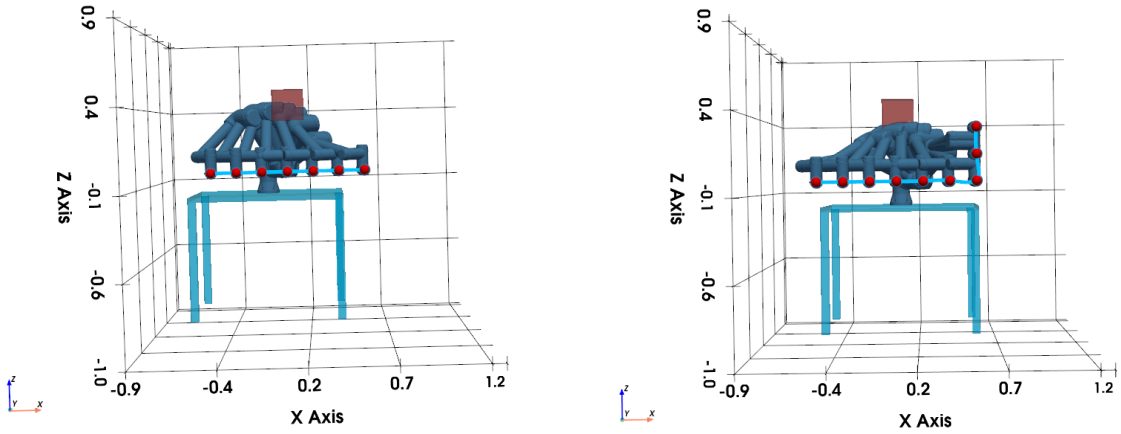

Figure 5.6: Invalid base placements showing collisions for linear and L-shaped trajectories.

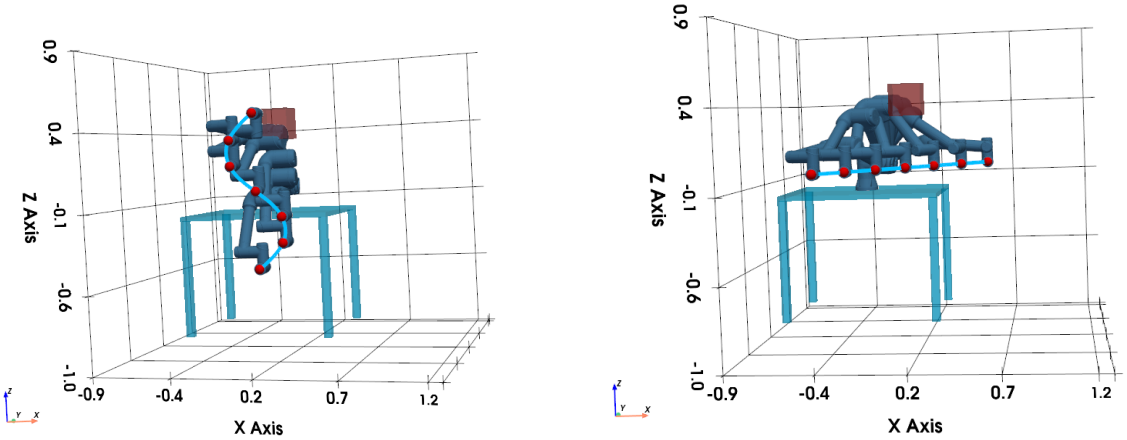

Figure 5.7: Additional invalid placements for sinusoidal and diagonal trajectories.

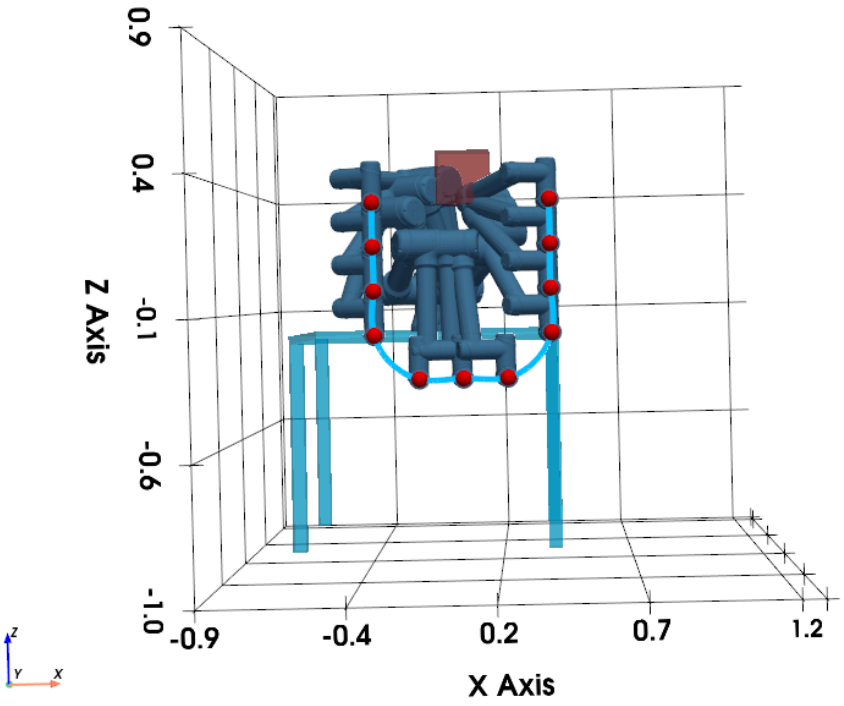

Figure 5.8: Additional invalid placements for U-shaped trajectory.

5.3 Analysis and Methodological Considerations

5.3.1 Resolution-Dependent Intersection Guarantees

The experimental results reveal a critical relationship between trajectory discretization strategy and solution existence. When trajectories are discretized **strictly aligned** with the map resolution (0.16 m steps), the candidate base pose sets B_i exhibit spatial overlap by construction—targets occupy the same voxel centers across the trajectory, maximizing intersection probability.
This **grid-snapping strategy** provides guaranteed solution existence (assuming kinematic reach is not exceeded) for coarser resolutions, where voxel spacing is large. However, for finer resolutions, the voxel dimensions shrink, and spatial proximity increases. In such cases, valid overlapping solutions may exist even if trajectory discretization is **decoupled from exact grid alignment**, as adjacent voxels provide sufficient spatial overlap.
Future work should investigate adaptive discretization strategies that balance computational efficiency (fewer targets to intersect) with solution robustness (sufficient spatial overlap for non-trivial B_{common}).

5.3.2 Conservative Collision Filtering: AABB Rotation Error

The iHDRM framework employs a highly conservative collision-checking mechanism that occasionally produces false-positive rejections. This conservatism arises from the coordinate transformation required during online queries: environmental obstacles, initially defined in the world frame, are transformed into the end-effector-centered frame used by the iHDRM.

- **End-Effector Frame:** After spatial rotation, the obstacle's point cloud is re-voxelized using Axis-Aligned Bounding Boxes (AABBs). Because AABBs cannot rotate, they artificially inflate around the rotated points, occupying additional voxels that the actual obstacle does not physically intersect.
- **World Frame Reality:** In the true world frame, the voxels tightly bound the obstacle geometry, and the robot safely clears it without collision.

This AABB inflation results in false-positive collision detections—the framework rejects base poses that are physically safe but appear dangerous due to the conservative voxel approximation. While this ensures absolute safety (no false negatives), it occasionally discards valid solutions, potentially reducing B_{valid} unnecessarily.

The computational advantage of this approach is real-time evaluation speed—voxel-based set intersection operates in $O(|V_{\text{obstacle}}|)$ time, orders of magnitude faster than explicit geometric collision detection. Future enhancements could mitigate AABB inflation through:

1. Oriented Bounding Boxes (OBBs) that rotate with the point cloud.
2. Multi-resolution voxel grids with finer local refinement.
3. Hybrid approaches using coarse voxel filtering followed by precise geometric validation for borderline cases.

5.3.3 Memory Complexity and Scalability Analysis

A significant computational challenge is the memory footprint required for the Occupation Map. The current implementation stores occupation lists O_i as unoptimized arrays of 32-bit integer configuration indices. For a workspace discretized at resolution r with N_{voxels} voxels and an average of k configurations per voxel, total memory scales as:

$$\text{Memory} \approx N_{\text{voxels}} \times k \times 4 \text{ bytes}$$

Since $N_{\text{voxels}} \propto (L/r)^3$ where L is the workspace dimension, memory consumption grows cubically as resolution becomes finer. For the experimental 0.16 m resolution covering a $\sim$2 m^3 workspace, this remains tractable. However, halving the resolution to 0.08 m would increase memory requirements by approximately 8×, quickly approaching practical limits.
Future optimizations should implement:

- **Sparse data structures:** Octrees store only occupied voxels, dramatically reducing memory for sparsely-occupied workspaces.
- **Bit-array encoding:** Representing configuration indices as bit flags (1 bit per possible configuration) instead of 32-bit integers reduces storage by 32×.
- **Run-length encoding:** Compressing contiguous sequences of configuration indices.
- **Hierarchical resolution:** Coarse voxels for rapid initial filtering, fine voxels for local refinement.

These techniques could reduce memory consumption by 1–2 orders of magnitude while preserving the $O(1)$ lookup speed essential for real-time performance.

5.3.4 Limitations: Fixed Orientation Discretization

The current experimental validation constrains end-effector orientation to remain **rigidly fixed** throughout trajectory execution. While this simplification enabled isolated validation of the spatial intersection logic (B_{common} computation) and collision filtering (Occupation Map effectiveness), it limits applicability to real-world tasks requiring dynamic tool rotations—such as contour welding, surface finishing, or complex assembly operations.
Extending the framework to support full $SE(3)$ trajectory discretization requires:

1. **Orientation sampling:** Discretizing not just spatial translations (0.16 m steps) but also orientational changes (e.g., 15° rotational increments).
2. **Denser orientation database:** Increasing the number of orientation samples per voxel during RM generation to capture richer rotational reach.
3. **Coupled discretization:** Ensuring trajectory discretization respects both translational and orientational resolution to maintain intersection guarantees.
4. **Computational scaling:** Managing increased intersection complexity as the number of orientation degrees of freedom expands.

This enhancement would transform the framework from validating translation-only paths to supporting fully articulated 6-DOF trajectories, substantially expanding applicability to realistic manufacturing scenarios.

5.4 Summary of Experimental Findings

The experimental validation demonstrates that the iHDRM framework successfully achieves its core objectives:

- **Spatial Intersection Logic:** The set-based formulation ($B_{\text{common}} = \bigcap B_i$) correctly identifies base locations enabling reach to all trajectory targets, validated across linear, diagonal, sinusoidal, U-shaped, and L-shaped paths.

- **Collision-Aware Filtering:** The Occupation Map mechanism efficiently eliminates collision-prone configurations, transforming geometric collision detection into rapid set intersection. Systematic comparison of "success" versus "invalid" cases confirms the framework actively prevents geometric infeasibility.
- **Quality Optimization:** Manipulability-based selection ensures chosen base poses maximize kinematic quality across trajectories, avoiding singular configurations and enhancing robustness.
- **Computational Efficiency:** Precomputed occupation data enables real-time online queries, with collision checking operating in $O(|V_{\text{obstacle}}|)$ time independent of configuration count.

The identified limitations—resolution-dependent discretization requirements, conservative AABB-based collision approximation, memory scaling challenges, and fixed-orientation constraints—provide clear directions for future enhancement. Nevertheless, the framework's ability to reliably identify collision-free, high-quality base placements for complex multi-target trajectories validates its practical viability for mobile manipulator deployment in structured industrial environments.

Chapter 6

Conclusion

This thesis introduced the Inverse Hybrid Dynamic Reachability Map (iHDRM) framework, a comprehensive solution for collision-aware, quality-optimized base placement of mobile manipulators executing multi-target trajectories. By explicitly integrating whole-body spatial awareness into inverse reachability representations, the framework transforms mobile manipulator planning from iterative trial-and-error into a principled optimization problem with provable correctness guarantees.

6.1 Principal Contributions

The iHDRM framework advances mobile manipulator base placement through three key innovations:

- **Integrated Collision-Aware Representation:** The Occupation Map precomputes which workspace voxels are occupied by the robot body for every joint configuration, enabling collision detection through rapid set intersection: Collision-Free$(q) \iff$ Occupied$(q) \cap V_{\text{obstacle}} = \emptyset$. This integration transforms geometric collision checking from $O(N_{\text{configs}})$ per-configuration operations into $O(|V_{\text{obstacle}}|)$ per-base-pose set intersections—a computational reduction of 1–2 orders of magnitude.

- **Set-Theoretic Multi-Target Optimization:** For trajectories $T = \{T_1, \ldots, T_N\}$, the framework computes common base poses through progressive intersection $B_{\text{common}} = B_1 \cap \cdots \cap B_N$, followed by configuration-level collision filtering to obtain $B_{\text{valid}} = \{b \mid \forall i : Q_i^{\text{free}}(b) \neq \emptyset\}$. This formulation provides completeness, correctness, and optimality guarantees.

- **Manipulability-Based Quality Selection:** The optimal base $b_{\textbf{optimal}}$ maximizes mean manipulability across all trajectory targets, ensuring consistently high kinematic quality, inherent singularity avoidance, and robust task execution.

  $b_{\text{optimal}} = \arg\max_b \left[\frac{1}{N} \sum_{i=1}^{N} w_{\max}(b, T_i) \right]$

6.2 Practical Impact and Limitations

6.2.1 Deployment Implications

The framework addresses critical barriers for industrial mobile manipulator deployment. Traditional fixed-base robot installation requires days to weeks of expert manual analysis per workstation—time-consuming, error-prone, and inflexible. Mounting manipulators like the UR5 on mobile platforms promises transformative flexibility but reintroduces the base placement

problem for every new task. The iHDRM enables **automated, rapid, provably-correct placement**, transforming weeks of analysis into seconds of query. This automation dramatically reduces deployment barriers and enables dynamic reconfiguration in response to production demands.
The framework provides particular value in cluttered environments—warehouses, manufacturing floors, assistive settings—through exhaustive collision checking (all 8 UR5 configurations evaluated), conservative safety guarantees, real-time responsiveness, and automatic support surface modeling.

6.2.2 Current Limitations

Experimental validation identified four principal limitations requiring future investigation:

- **Conservative Collision Approximation:** AABB-based voxelization of rotated obstacles produces artificially inflated volumes, triggering false-positive rejections of physically safe base poses. While guaranteeing safety, this conservatism unnecessarily constrains B_{valid}. Hybrid approaches—coarse voxel filtering followed by precise geometric validation—could improve this trade-off.
- **Memory Scaling:** Occupation Map memory consumption scales cubically with resolution (halving voxel size increases memory 8×). Sparse data structures (Octrees), bit-array encoding (32× reduction), and hierarchical resolution could reduce consumption by 1–2 orders of magnitude while preserving lookup efficiency.
- **Fixed Orientation Constraint:** Experimental validation held end-effector orientation rigid, isolating spatial intersection logic. Real-world tasks (welding, finishing, assembly) require dynamic tool rotations. Extending to full $SE(3)$ trajectory support demands coupled position-orientation discretization, richer orientation databases.
- **Grid-Aligned Discretization:** Guaranteed solution existence requires trajectory discretization aligned with map resolution. Adaptive strategies—dense sampling at inflection points, sparse sampling along straight segments—could balance computational efficiency against solution robustness.

6.3 Future Research Directions

The framework establishes foundations for four research directions:

- **Dynamic Environments:** Current assumptions of static obstacles limit applicability in settings with moving machinery, human coworkers, or temporary materials. Extensions for incremental map updates, temporal reasoning, reactive replanning, and sensor fusion (LiDAR, cameras) would enable adaptive behavior. Machine learning approaches—particularly neural networks predicting occupation list modifications from obstacle motion—could enable real-time responsiveness in human-shared workspaces.
- **Multi-Robot Coordination:** Multiple mobile manipulators in shared workspaces require collaborative occupation maps (unions of occupation lists), centralized optimization assigning base poses to minimize conflicts, or decentralized negotiation protocols. Temporal reservation systems using time-indexed occupation data could enable sequential workspace access.
- **Task-Specific Quality Metrics:** The Yoshikawa manipulability index provides universal kinematic assessment, but task-specific metrics could improve performance: directional manipulability for force-intensive tasks, trajectory smoothness for high-speed operations,

energy efficiency minimizing joint motion, or learned preferences from neural networks trained on successful executions. Reinforcement learning could discover optimal metrics through simulation or real-world trials.

- **High-Level Task Planning Integration:** The framework addresses base placement for given trajectories but leaves higher-level questions open: which tasks from which bases, how to sequence tasks minimizing repositioning, when to move versus adapt in place. Integration with task-and-motion planning (TAMP) frameworks—where symbolic planning reasons about task sequences while motion planning queries the iHDRM for feasibility—would enable end-to-end autonomous goal-directed behavior.

6.4 Concluding Perspective

The transition from fixed-base to mobile manipulation represents a fundamental shift in robotic automation—from dedicated single-purpose workstations to flexible multi-purpose platforms. However, this mobility introduces the challenging problem of determining where to position the base for safe, effective task execution. The iHDRM framework addresses this challenge through principled integration of kinematic reachability, whole-body collision awareness, and quality-based optimization.

The experimental validation demonstrates practical viability: the framework successfully identifies collision-free, high-quality base placements for diverse trajectory shapes (linear, diagonal, sinusoidal, U-shaped, L-shaped) in cluttered environments, validated through systematic comparison of valid versus invalid configurations. The precomputed occupation data transforms collision detection from an online computational bottleneck into an offline preprocessing step, enabling real-time queries. The set-theoretic formulation provides mathematical rigor—guaranteeing completeness, correctness, and optimality within the discrete representation.

Looking forward, the iHDRM framework establishes building blocks for increasingly autonomous mobile manipulation. As industrial automation confronts demands for greater flexibility—customized production, small-batch manufacturing, rapidly-changing product lines—mobile manipulators with automated, intelligent base placement become not merely advantageous but essential. The framework demonstrates that collision-aware, quality-optimized base placement is both theoretically sound and computationally tractable, providing a practical path toward deploying industrial manipulators with the flexibility of mobility and the reliability of proven fixed-base systems. This synthesis of capability and confidence represents a meaningful step toward realizing the promise of mobile manipulation in real-world industrial environments.

Bibliography

1. F. Zacharias, C. Borst, and G. Hirzinger, “Capturing robot workspace structure: representing robot capabilities,” in 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3229–3236, Oct 2007.

2. F. Zacharias, C. Borst, M. Beetz, and G. Hirzinger, “Positioning mobile manipulators to perform constrained linear trajectories,” in 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2578–2584, Sept 2008.

3. N. Vahrenkamp, T. Asfour, and y. p. Rudiger Dillmann, journal=2013 IEEE International Conference on Robotics and Automation, “Robot placement based on reachability inversion,”

4. F. Burget and M. Bennewitz, “Stance Selection for Humanoid Grasping Tasks by Inverse Reachability Maps,” in IEEE International Conference on Robotics and Automation, 2015, p. 5669–5674.

5. J. Dong and J. C. Trinkle, “Orientation-based reachability map for robot base placement,” in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1488–1493, Sept 2015.

6. Y. Yang, V. Ivan, Z. Li, M. Fallon, and S. Vijayakumar, “iDRM: Humanoid Motion planning with Real-Time End-Pose Selection inComplex Environments,” in IEEE-RAS International Conference on Humanoid Robots, 2016, pp. 271–278.

7. A. Hertle and B. Nebel, ”Identifying good poses when doing your household chores: Creation and exploitation of inverse surface reachability maps,” 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 2017, pp. 6053-6058

8. S. Osswald, P. Karkowski, and M. Bennewitz, “Efficient Coverage of 3D Environments with Humanoid Robots using Inverse Reachability Maps,” in IEEE-RAS International Conference on Humanoid Robots, 2017, pp. 151–157

9. Abhijit Makhal and Alex K. Goins, ”Reuleaux: Robot Base Placement by Reachability Analysis”, in 2018 Second IEEE International Conference on Robotic Computing

10. T. Birr, C. Pohl and T. Asfour, ”Oriented Surface Reachability Maps for Robot Placement,” 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022.

11. T. Sandakalum, N. X. Yao, and M. H. Ang, “Inv-reach net: Deciding mobile platform placement for a given task,” in 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids). IEEE, Nov. 2022.

12. Zhong, M., He, Y., Liu, Y., Han, R., Liu, Y. (2023). Optimization of Wheelchair-Mounted Robotic Arms' Base Placement by Fusing Occupied Grid Map and Inverse Reachability Map. Applied Sciences, 13(14), 8510.

13. Rudorfer, Martin. "Rm4d: A combined reachability and inverse reachability map for common 6-/7-axis robot arms by dimensionality reduction to 4d." 2025 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2025.

14. Cho, Y.; Choi, D.; Ko, K.; Park, J.H. Position Selection and Collision-Free Path Planning for Fruit Picking Robots. Appl. Sci. 2025, 15, 4419.

15. Ince, Caner-Veli Niestroj, Jan Raatz, Annika. (2025). 6-DoF Robot Base Positioning Methodology in Manufacturing Cells Using Inverse Reachability (ARBP-IR). Procedia CIRP. 136. 302-307. 10.1016/j.procir.2025.08.053.

16. Yoshikawa, Foundations of Robotics: Analysis and Control. Cambridge, Massachusetts: MIT Press, 1990.

17. Y. Li, Y. Zhang, A. Razmjoo, and S. Calinon, "Representing Robot Geometry as Distance Fields: Applications to Whole-body Manipulation," in Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA), 2024, pp. 15351–15357.