Reactive mobile manipulator base sequence planning for inspection in dynamic environments

Reactive mobile manipulator base sequence planning for inspection in dynamic environments

Rosario Francesco Cavelli, Cesare Luigi Blengini, Pangcheng David Cen Cheng, Marina Indri
Department of Electronics and Telecommunications
Politecnico di Torino
Corso Duca degli Abruzzi, 24, 10129, Torino, Italy
{rosario.cavelli, cesare.blengini, pangcheng.cencheng, marina.indri}@polito.it

***Abstract*—This paper presents an optimization-based methodology for a mobile manipulator base sequence planning, with a focus on multi-target inspection tasks. The proposed method exploits an ellipsoid-based model of the robot's reachable space to maximize the number of reachable targets (defined by a set of AprilTags) from each mobile base pose, while minimizing collision risks and base repositioning operations. The proposed optimization approach incorporates dynamic obstacle perception using an RGB-D camera to determine whether an obstacle is static or dynamic, and to react accordingly. The performance of the proposed method is demonstrated through experiments in a digital twin of a laboratory scenario with realistic constraints.**

***Index Terms*—Base sequence planning, multi-target tasks, inspection tasks, mobile manipulation**

I. Introduction

Manipulators are widely used to automate different processes, reducing human effort and enhancing safety. A correct base positioning goes beyond reachability, since it also considers collision avoidance, performance optimization, and joint motion continuity [1]. Applications such as pick and place, assembly, coverage, and inspection require an effective base placement. When tasks span areas beyond the workspace of a fixed-base manipulator, mobile manipulators represent a perfect solution, thanks to the additional Degrees of Freedom (DOF) introduced by the mobile base. However, finding an effective base pose remains challenging, especially under localization uncertainty. Tasks assigned to mobile manipulators typically require completing sequences of actions at predetermined locations. Planning a sequence of base poses, each covering multiple targets, reduces execution time and energy consumption [2]. Also, it minimizes the number of base placements, which is a key objective in Robotic Task Sequencing Problems (RTSPs) due to localization uncertainties [3].

In the literature, several approaches address the base placement problem for fixed-base and mobile manipulators, but most consider static scenarios. For instance, the Hausdorff approximation planner is employed in [4] to compute clustered regions for efficient task sequences, but in a reduced search space to increase the computational speed. Relatively few works consider dynamic environments with mobile agents or humans, as in [5]. The approach in [6] proposes replanning when a dynamic obstacle limits the robot's view, whereas the one in [7] plans base-pose sequences via reachability inversion while accounting for navigation costs and human-induced changes. However, these works react to dynamic obstacles only at the motion-planning level, without integrating their handling into the multi-target base-sequence optimization. Optimizing base placement is advantageous for coverage and inspection tasks [8], [9]. In [10], a genetic algorithm is optimized for station viewpoint sequences to improve viewpoint quantity and view quality in mobile visual inspection. Sequence planning for coverage applications can also be addressed via heuristic approaches with kinematic inversion [11] or hybrid algorithms combined with reinforcement learning [12]. Yet, these inspection-oriented methods are mostly designed for static scenes and rely on offline plans, lacking the reactivity required when obstacles dynamically affect target reachability.

This study was carried out within the Space It Up project funded by the Italian Space Agency, ASI, and the Ministry of University and Research, MUR, under contract n. 2024-5-E.0 - CUP n. I53D24000060005.

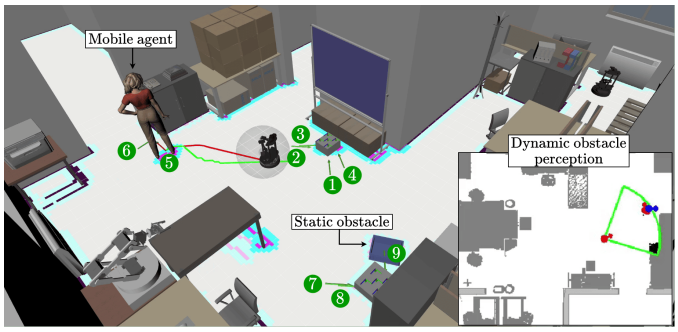

Fig. 1: Application scenario for the proposed framework. The numbers highlight the visiting order of 9 base poses to inspect all 20 targets in the scene.

This paper presents a unified framework for multi-target base-sequence planning in environments where dynamic obstacles can alter target reachability at runtime, adopting a reactive policy to handle unmapped obstacles or mobile agents. Based on [13], the proposed framework extends the optimization-based method in [14] to a multi-target base-sequence planner that considers dexterity, collision risk, and feasible navigation. Figure 1 shows the application scenario for the proposed framework. The rest of the paper is structured as follows: Section II provides the necessary preliminaries; Section III describes the proposed approach; Section IV presents its validation in a digital twin of a laboratory environment; Section V concludes the paper with remarks on future work.

II. Preliminaries

The core of the proposed approach, which will be described in Section III, is an optimization-based procedure that

leverages a compact mathematical model of the Reachability Space (RS) (previously introduced in [14]). Such a model is a compact representation of the optimal portion of the reachable space: it encloses points reachable with a high dexterity index within two concentric ellipsoids, each one defined as $(\mathbf{p}-\mathbf{c})^\top \mathbf{E}(\mathbf{p}-\mathbf{c}) \leq 1$, where $\mathbf{p} \in \mathbb{R}^3$ is a generic point belonging to the ellipsoid, $\mathbf{c} \in \mathbb{R}^3$ is the center of the ellipsoid, and $\mathbf{E} = \mathrm{diag}\left(\frac{1}{A^2}, \frac{1}{B^2}, \frac{1}{C^2}\right)$ is the diagonal matrix computed starting from the ellipsoid's axis length $A, B, C \in \mathbb{R}^+$.

In [14], the RS model was used as a constraint of an optimization problem, which selects the mobile-base pose so that a single end-effector (EE) target lies in the region enclosed between the inner and outer ellipsoids. This ensured reachability with high dexterity, while minimizing the number of environment voxels falling inside the outer ellipsoid to reduce the collision risk. Building on this formulation, the present paper generalizes the optimization-based procedure to multi-target scenarios and base sequence planning.

Let us define the set of desired EE targets as $\mathcal{T} = \{\mathbf{t}_i\}_{i=1}^N$, where each target $\mathbf{t}_i \in \mathbb{R}^7$ can be split into $\mathbf{p}_i \in \mathbb{R}^3$ and $\mathbf{o}_i \in \mathbb{R}^4$, representing respectively its position and orientation (as a quaternion), with respect to the world reference frame. The optimization problem (see Section III) is solved iteratively: at each step, a tentative solution $\tilde{\mathbf{x}}$ yields the ellipsoids' center and the corresponding base pose, so that the RS region between the two ellipsoids covers a different portion of the environment and reaches a different subset of targets. This is exploited to divide the target set $\mathcal{T}$ into two subsets:

$$\mathcal{T}_{\text{inn}}(\tilde{\mathbf{x}}) = \left\{\mathbf{t}_i \in \mathcal{T} \mid (\mathbf{p}_i - \tilde{\mathbf{c}})^T \mathbf{E_{inn}} (\mathbf{p}_i - \tilde{\mathbf{c}}) \leq 1\right\},$$

$$\mathcal{T}_{\text{out}}(\tilde{\mathbf{x}}) = \left\{\mathbf{t}_i \in \mathcal{T} \mid (\mathbf{p}_i - \tilde{\mathbf{c}})^T \mathbf{E_{out}} (\mathbf{p}_i - \tilde{\mathbf{c}}) \leq 1\right\},$$

where $\tilde{\mathbf{c}} \in \mathbb{R}^3$ is the position of the ellipsoids' center derived from $\tilde{\mathbf{x}}$, while $\mathbf{E_{inn}}$ and $\mathbf{E_{out}}$ are the diagonal matrices containing the length of the axes of the inner and outer ellipsoids, respectively. The sets $\mathcal{T}_{\text{inn}}$ and $\mathcal{T}_{\text{out}}$ include the targets falling inside the inner and outer ellipsoid, respectively.

III. The Proposed Approach

The optimization procedure for computing feasible base poses is coupled with the observation of the environment to detect previously unknown obstacles. Indeed, if the optimal base goal becomes unreachable because of a new obstacle, the robot's behaviour is adapted depending on the nature of the obstacle itself.

A. Optimization problem

The core of the procedure for the base placement optimization can be formulated as a minimization problem:

$$\begin{aligned} \mathbf{x}^\star = \arg\min_{\mathbf{x}} \quad & \alpha \#Coll(\mathbf{x}) - \beta \#\mathcal{RT}(\mathbf{x}) + \gamma \left| \sum_{\mathbf{t}_i \in \mathcal{RT}(\mathbf{x})} \delta_{\mathbf{t}_i}(\mathbf{x}) \right| \\ \text{s.t.:} \quad & p_{i,1} \geq 0, \ \forall \mathbf{t}_i \in \mathcal{RT}(\mathbf{x}), \\ & \mathcal{RT}(\mathbf{x}) \neq \emptyset, \\ & \mathcal{T}_{\text{inn}}(\mathbf{x}) = \emptyset, \\ & [x_{\text{b}}, y_{\text{b}}] \in \mathcal{C}_{\text{free}}. \end{aligned} \tag{1}$$

The decision variable $\mathbf{x}$ contains the parameters describing the pose of the reference frame located at the ellipsoids' center ($\mathrm{R_{ell}}$ in Figure 2a) that are free to vary thanks to the DOF of the mobile base. Specifically, $\mathbf{x} = [x, y, \theta]$, where x and y are the unconstrained coordinates of the origin of $\mathrm{R_{ell}}$, and θ represents the rotation of this frame about the z-axis of the fixed reference frame ($\mathrm{R_0}$ in Figure 2a). The corresponding mobile-base pose $\mathbf{x}_{\text{b}} = [x_{\text{b}}, y_{\text{b}}, \theta_{\text{b}}]$, computed with respect to $\mathrm{R_0}$, is obtained from $\mathbf{x}$ through a fixed rigid transformation $\mathbf{T}^{\text{b}}_{\text{ell}}$, encoding the known offset between ellipsoid centers and the base frame $\mathrm{R_b}$.

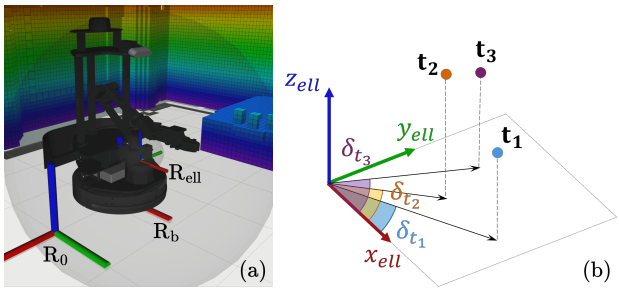

Fig. 2: (a) Visualization of reference frames $\mathrm{R_0}$, $\mathrm{R_b}$ and $\mathrm{R_{ell}}$. (b) Projection of $\mathbf{t_1}$, $\mathbf{t_2}$ and $\mathbf{t_3}$ on the XY plane of $\mathrm{R_{ell}}$ and definition of the angles $\delta_{\mathbf{t_1}}$, $\delta_{\mathbf{t_2}}$ and $\delta_{\mathbf{t_3}}$.

Objective function The scalars $\alpha, \beta, \gamma \in \mathbb{R}^+$ are user-defined weights that balance the contributions of the three terms in the objective, and unify their heterogeneous units. The operator $\#$ denotes the cardinality of a set. As defined in [14], the set $Coll(\mathbf{x})$ collects obstacle parts that fall inside the optimal reachable space induced by $\mathbf{x}$. Minimizing $\alpha \#Coll(\mathbf{x})$, hence, discourages base placements with a high risk of collisions. The set $\mathcal{RT}(\mathbf{x}) \subseteq \mathcal{T}$ denotes the subset of targets lying in the portion of the RS characterized by a high dexterity index, i.e., those targets that fall between the two ellipsoids:

$$\mathcal{RT}(\mathbf{x}) = \mathcal{T}_{\text{out}}(\mathbf{x}) \setminus \mathcal{T}_{\text{inn}}(\mathbf{x}). \tag{2}$$

The term $-\beta \#\mathcal{RT}(\mathbf{x})$ rewards solutions that include more targets in $\mathcal{RT}(\mathbf{x})$, thus favoring base poses that enable handling a larger number of targets from a single base placement. Finally, the term:

$$\gamma \left| \sum_{\mathbf{t}_i \in \mathcal{RT}(\mathbf{x})} \delta_{\mathbf{t}_i}(\mathbf{x}) \right|$$

acts as a regularizer. For each target $\mathbf{t}_i \in \mathcal{RT}(\mathbf{x})$, $\delta_{\mathbf{t}_i}(\mathbf{x}) \in [-\pi, \pi)$ denotes the signed angle between the x-axis of $\mathrm{R_{ell}}$ and the projection onto its XY plane of the vector pointing from the origin to the target, as shown in Figure 2b. Since $\delta_{\mathbf{t}_i}(\mathbf{x})$ is a signed angle, targets located on opposite sides of the base x-axis contribute with opposite signs and tend to cancel out in the sum. Minimizing the absolute value, therefore, encourages base placement yielding to a symmetric arrangement of the targets in $\mathcal{RT}(\mathbf{x})$.

Constraints The constraint $p_{i,1} \geq 0, \ \forall \mathbf{t}_i \in \mathcal{RT}(\mathbf{x})$ enforces the x-coordinate of each reachable target to be non-

negative. This feasibility condition biases the optimization toward base poses for which the desired reachable targets lie in front of the robot, i.e., in the region typically covered by perception sensors, thus improving target detectability. Other perceptual requirements could be enforced by adding field-of-view constraints to explicitly keep targets within the sensor visibility region.

The constraint $\mathcal{RT}(\mathbf{x}) \neq \emptyset$ removes degenerate solutions where the base placement does not guarantee reaching any of the desired EE targets. In addition, enforcing $\mathcal{T}_{\text{inn}}(\mathbf{x}) = \emptyset$ prevents selecting targets that, although formally reachable, lie in kinematically unfavorable regions of the workspace near the manipulator itself. This choice improves execution robustness by avoiding configurations associated with low dexterity or near-singular behavior, which may require excessive joint velocities and degrade tracking accuracy.

Finally, $[x_{\text{b}}, y_{\text{b}}] \in \mathcal{C}_{\text{free}}$ restricts the base pose to the collision-free navigable space; $\mathcal{C}_{\text{free}}$ is defined as the free-space component that is path-connected to the current base pose, so that for any $[x_{\text{b}}, y_{\text{b}}] \in \mathcal{C}_{\text{free}}$ at least one collision-free navigation trajectory exists from the current robot pose.

Overall, the optimization formulation in (1) allows a feasible robot base placement to reach several targets while penalizing potential collisions. At the same time, the angular regularizer and the frontal-feasibility constraint bias the solution toward perceptually convenient and well-balanced target layouts. Enforcing $[x_{\text{b}}, y_{\text{b}}] \in \mathcal{C}_{\text{free}}$ guarantees that the optimal base pose is reachable from the current robot position through at least one feasible navigation trajectory within the free space.

B. Iterative base-sequence generation

The optimization problem in (1) is embedded in an iterative scheme to generate a sequence of robot base poses that collectively cover the full target set $\mathcal{T}$. Such set is firstly partitioned into K disjoint clusters $\{\mathcal{T}_k\}_{k=1}^{K}$, with $\mathcal{T} = \bigcup_{k=1}^{K} \mathcal{T}_k$ and $\mathcal{T}_k \cap \mathcal{T}_\ell = \emptyset$ for $k \neq \ell$, so that nearby targets that can be potentially reached from the same base pose are grouped. The clusters are then assigned to the optimization problem in (1) according to an order obtained by solving a Traveling Salesman Problem (TSP) over cluster representatives, e.g., their centroids. The aim is to minimize the total navigation length from the robot's initial position to the final destination while visiting all targets.

For a given cluster $\mathcal{T}_k$, the residual set, i.e., the set of targets for which a base pose still needs to be computed, is initialized as $\mathcal{T}_k^{(0)} = \mathcal{T}_k$ and iterates as follows: at iteration i, the optimization problem in (1) is solved over $\mathcal{T}_k^{(i)}$, obtaining the optimal ellipsoid pose $\mathbf{x}_{(i)}^{\star}$ and the associated base pose $\mathbf{x}_{\text{b},(i)}^{\star}$. All targets reachable from $\mathbf{x}_{\text{b},(i)}^{\star}$ are removed from the residual set to form $\mathcal{T}_k^{(i+1)}$, and $\mathbf{x}_{\text{b},(i)}^{\star}$ is appended to the output base-pose sequence. The loop terminates when $\mathcal{T}_k^{(i)} = \emptyset$ or when no feasible solution is found for the current residual set.

The iterative optimization process successfully terminates if the improvement of the objective value is lower than a predefined tolerance ε_{tol}, corresponding to convergence to a locally optimal solution for the current residual set. Conversely, termination due to exceeding either the maximum number of iterations $N_{\max}$ or the maximum available computation time $T_{\max}$ indicates that no solution satisfying all constraints could be found for the residual set. In this case, the iterative procedure is interrupted, and only the poses computed in the previous iterations are assigned as targets to the mobile base.

C. Computation of directly navigable free space

The navigability constraint in (1) is enforced by explicitly constructing the set $\mathcal{C}_{\text{free}}$ as the subset of planar locations that are both collision-free for the mobile base and path-connected to the current robot pose. To this end, a 2D occupancy map is obtained as the planar projection of the 3D environment representation, and processed to produce a binary free-space mask.

First, the occupancy grid is converted into a 2D image representation, as shown in Figure 3a. Then, obstacles are inflated by a radius accounting for the robot footprint r_{robot} and a safety margin r_{safe}, yielding a conservative map where any remaining free pixel can accommodate the base without intersecting known obstacles (Figure 3b). Finally, to exclude free regions that are not directly reachable by the mobile base, a flood-fill expansion is initialized from the current robot position and used to extract only the connected component of the inflated free space (Figure 3c). The resulting mask shown in Figure 3d defines $\mathcal{C}_{\text{free}}$ and is used as a hard feasibility test during optimization, so that candidate base poses are accepted only if they lie within the free-space region.

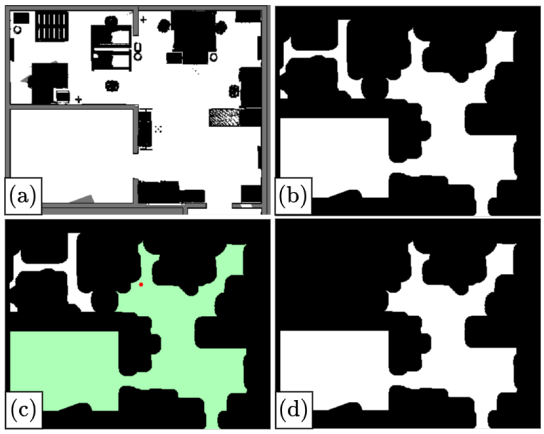

Fig. 3: Pipeline to compute the $\mathcal{C}_{\text{free}}$ mask starting from the 2D projection of the occupancy map.

D. Dynamic obstacle perception

During execution, some mobile base poses of the sequence computed by the optimization can become invalid due to previously unknown changes in the scene. The map-based representation of the environment, on which the optimization in (1) relies to enforce reachability of the computed base poses, may become outdated when new objects appear after planning. To detect and handle these changes, a perception layer is introduced to build a dynamic-obstacle map.

The perception layer relies on an onboard RGB-D camera to provide both appearance and depth cues of the environment. Depth measurements are integrated to update the 3D

environment model online. The updated 3D representation is projected onto the ground plane to refresh the 2D occupancy map, thus accounting for newly observed obstacles. In parallel, semantic segmentation is performed to understand whether the newly perceived obstacle is a *mobile agent*, e.g., a person or another mobile robot. The segmentation is performed using GroundedSAM [15]. It enables identifying the classes of interest without retraining the model for the specific environment. This property also allows updating the set of classes at runtime to extend the robot's reactions to new classes of obstacles. The pixel-level masks obtained from the RGB images are lifted to 3D through depth back-projection, and then projected onto the 2D map to obtain a mask that marks the projection of newly detected obstacles as mobile agents.

Since dynamic obstacles are perceived through the onboard camera, their state can only be asserted within the portion of space that is currently observable. For this reason, dynamic-obstacle information is tracked only inside the camera visibility frustum, defined by its field-of-view and effective sensing range. When a dynamic obstacle leaves the camera frustum, its state is no longer maintained. This choice follows from relying exclusively on onboard sensing and from the need to limit the computational load of online perception during navigation.

E. Obstruction check and reactive policy

The dynamic-obstacle map is exploited online to assess whether the next navigation goal remains feasible while the robot is moving. In particular, when the goal lies inside the current camera visibility frustum, its location is tested for validity by jointly considering (i) the projected 2D occupancy map, and (ii) the projected dynamic-obstacle mask. This frustum-based check ensures that the robot's reactions are triggered only for new obstacles that are in regions currently observable by the onboard RGB-D camera.

If the target base pose is detected as obstructed, the navigation toward it is interrupted, and the obstruction is re-evaluated after a short waiting time. This is done to reduce sensitivity to transient obstacles and sensor noise. If the obstruction persists, the obstacle is classified based on the dynamic-obstacle mask. If the obstruction is due to the presence of a mobile agent, the goal is deferred by re-queuing it for later execution, under the assumption that the obstruction may disappear. It is worth noting that re-queuing is performed over the entire base-pose sequence, which may include base poses from which it is possible to reach EE targets belonging to other clusters. If no mobile agents are perceived, the obstruction is treated as due to a static obstacle. In this latter case, re-optimization is triggered immediately, and the optimization problem in (1) is solved using as $\mathcal{T}_k$ the set of EE targets that would have been reachable from the obstructed base pose. Moreover, $\mathcal{C}_{\text{free}}$ is computed again considering the updated 2D occupancy grid.

IV. Implementation and Validation Tests

Challenging scenarios for mobile manipulators are given by indoor spaces such as laboratories or offices, where the furniture and equipment limit the robot's navigation and base placement. The proposed optimization framework is then implemented and validated in the digital twin of a laboratory (Figure 4), where it is difficult to find optimal base poses to reach the desired EE targets. In this way, realistic constraints could be tested in a controlled and repeatable setting, without possible contingent problems, like localization noise.

A LoCoBot WX250 [16] was used as a test platform; it is a mobile manipulator equipped with a 6-DOF arm, an Intel RealSense D435 RGB-D camera, and a 2D LiDAR. Most of the software was developed and executed on Ubuntu 20.04 using Python 3.8.10 and ROS1 Noetic [17]. The simulation environment was implemented in Gazebo 11, while RViz was used for visualization. The construction of the obstacle map relied on OctoMap, while indoor localization was performed using RTAB-Map.

The semantic segmentation module based on Grounded-SAM was executed on a separate workstation equipped with an NVIDIA RTX 5070 Ti and running Ubuntu 24.04. Compatibility issues across operating systems and ROS versions were addressed by adopting an MQTT-based bridge to exchange RGB-D images and the resulting segmentation masks between the two machines. The code used in the experimental campaign is available at [18].

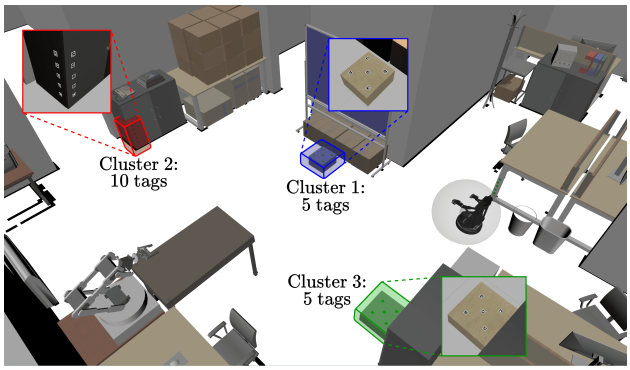

Fig. 4: Top view of the simulation environment representing a digital twin of a real laboratory. The distribution of the tags in the environment and their clustering are shown, too.

A. Problem scenario

The carried out tests are focused on inspection tasks. A set of AprilTags is distributed in various positions in the environment, with different orientations. The robot is asked to navigate to the inspection locations and to scan each tag by bringing the EE to the associated target pose. To this aim, the standard two-finger parallel gripper is replaced by a secondary RGB-D camera mounted at the EE, enabling close-range acquisition of each tag.

During execution, previously unknown obstacles may appear and obstruct planned base goals; the framework is expected to detect these changes and to adapt depending on whether the obstruction is caused by a mobile agent or by a static object.

B. AprilTag detection poses

When an AprilTag is detected using the `apriltag_ros` package [19], a tag-fixed reference frame is created, with x and y lying on the tag plane and z orthogonal to it and exiting

the tag plane. An EE inspection target pose is then defined from this frame by placing the EE origin at $5\,cm$ along the tag z axis and setting the EE orientation with $x_{EE} \parallel -z_{tag}$ and $z_{EE} \parallel y_{tag}$ (the remaining axis completes a right-handed frame), as illustrated in Figure 5.

In the inspection task, a target is marked as successfully inspected upon detection of the corresponding tag ID by the RGB-D camera mounted as EE of the mobile manipulator.

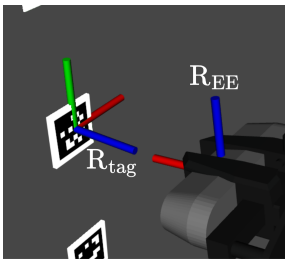

Fig. 5: The tag reference frame $\mathrm{R_{tag}}$ and the EE target pose $\mathrm{R_{EE}}$ used for its detection. The x, y, and z axes of each reference frame are reported in red, green, and blue, respectively.

C. Test 1: Unobstructed execution

The aim of the first test was the validation of the effectiveness of the base sequence obtained by the iterative optimization method presented in Section III in nominal conditions. In this test, all the computed base poses were reachable, and no previously unknown obstacles appeared during execution.

A total of 20 target EE poses were used, each associated with an AprilTag placed at different positions and orientations across the laboratory environment. Tags were distributed across 3 distinct areas of the environment, on both vertical and horizontal surfaces, to stress the reachability model.

The 20 targets were partitioned into 3 clusters, whose visiting order was defined by solving the TSP associated with the clusters' centroids. The visiting order obtained aims to minimize the estimated navigation path needed to reach all EE targets, while navigating from its starting position towards the final one. Figure 4 shows the distribution of the targets in the environment, the resulting clusters, and their visiting order.

The base pose sequence obtained to reach all the EE targets is shown in Figure 6. A total of 9 base poses have been computed to inspect the 20 tags over the environment. The robot successfully navigated to all 9 base poses in the planned order and completed the inspection of all 20 targets. Figure 7 shows representative frames from the video of the complete experiment that is available at [20].

D. Test 2: Reactive execution in case of obstructions

A second test was conducted using the same environment layout, i.e., the same target set, and cluster partition, achieving an almost identical base sequence, having the same number of base poses with only minor changes in the position and orientation of each pose. In this second test, however, mobile agents and a static obstacle appeared during execution, triggering the reactive components of the framework.

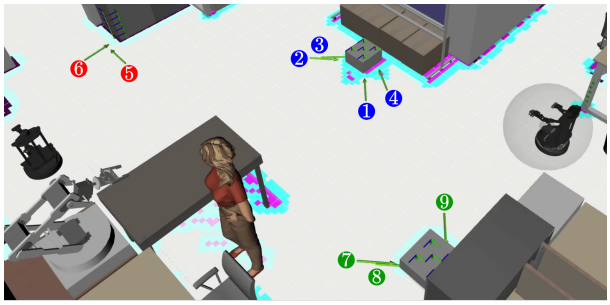

Fig. 6: Sequence of base poses computed to inspect all tags in the environment.

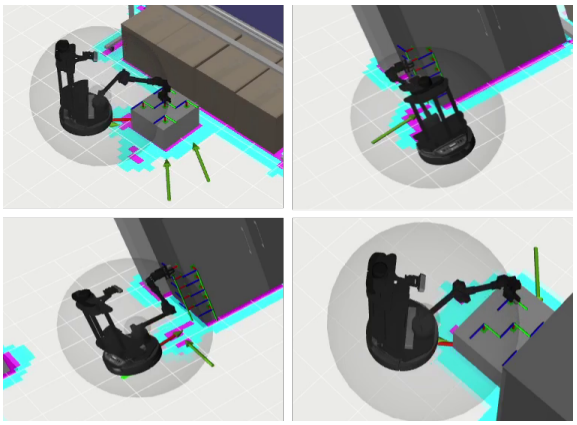

Fig. 7: Frames of the video of the first experiment [20].

While navigating towards the first base pose, the frustum-based check detected an obstruction within the camera field of view (Figure 8a); the segmentation pipeline detected a mobile agent (Figure 8b), in particular a mobile robot, triggering a re-queue of the goal. The remaining 3 base poses allowing to reach the AprilTags in cluster 1 were unobstructed, hence executed as planned, completing the inspection of the associated targets.

A similar event occurred for the fifth target base pose, where a human agent was detected (Figures 8c and 8d). Also in this case, the goal was re-queued, and the associated tags inspection deferred. The remaining base pose was reached instead, and the tag associated with it was inspected successfully.

Similarly, while navigating toward the ninth and last planned base pose, the frustum-based check detected an obstruction; since the segmentation pipeline did not return a mobile agent (Figures 8e and 8f), the obstruction was classified as corresponding to a new static obstacle, triggering immediate re-optimization. A new instance of the optimization problem (1) was created using the associated target subset T_9, and the updated $\mathcal{C}_{\mathrm{free}}$. However, in this case, the time limit was reached, yielding to the termination without finding a suitable base pose. This led to the impossibility of reaching one target of the last cluster. This failure was due to the position of the newly detected obstacle, which did not allow to find a collision-free pose of the mobile base suitable to reach the tag. Finally, the robot reached the two re-queued base poses.

Neither the mobile robot nor the human agent was present, allowing the mobile manipulator to reach the base poses as originally planned, and to inspect the remaining targets. The complete video of the second experiment can be found at [20].

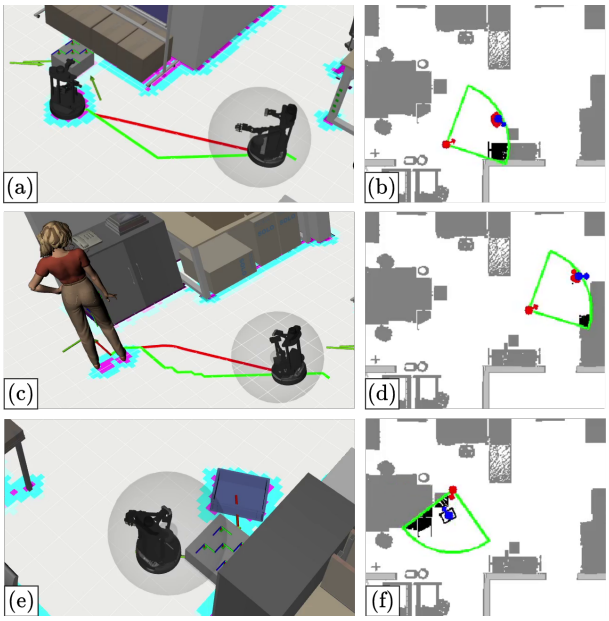

Fig. 8: Detection and classification of the obstacles obstructing the base poses from the video in [20].

E. Experimental results

During both tests, a mean time of $4.9\,s$ was required to solve an instance of the optimization problem in (1), hence, to find each base pose. This value is largely dominated by the computational overhead of the simulation environment. Indeed, preliminary tests carried out on the physical LoCoBot WX250 yielded a mean optimization time of about $1.1\,s$ per base pose, i.e., roughly 22% of the time measured in simulation, confirming that the framework is feasible for real-time replanning on the real platform. To assess the effectiveness of the computed base poses for reaching the desired EE targets, the difference between the reached EE pose and the desired one was measured. A mean error of about $5\,mm$ on the position, and a mean error of $0.34\,rad$ for the orientation were achieved. It is worth noting that the code used during the experimental campaign is not optimized: further refinement could lower the computation time, leading to better performances.

V. Conclusion and Future Works

This paper presented an optimization-based framework for mobile manipulator base sequence planning. The base placements aimed to reach a set of EE poses in an inspection task, while minimizing the risk of collisions. Experiments in a simulated laboratory validated both the nominal planning pipeline and the reactive behavior under static and dynamic obstructions. However, during the experiments, some limitations arose. The obtained intra-cluster visiting order could be improved through a weighted-target formulation to enforce a more structured ordering. Re-queuing deferred goals to the end of the full sequence may also disrupt the spatially optimized inter-cluster order; a proximity-aware rescheduler could address such an issue. Future work will focus on deploying the framework on the physical robot in more dynamic environments, defining dedicated quantitative metrics for a thorough performance evaluation, and comparing it against state-of-the-art methods.

References

[1] Z. Zhao, L. Cui, S. Xie, S. Zhang, Z. Han, L. Ruan, and Y. Zhu, "B*: Efficient and optimal base placement for fixed-base manipulators," *IEEE Robotics and Automation Letters*, 2025.

[2] D. Spensieri, R. Salman, J. S. Carlson *et al.*, "A unified sampling method for optimal feature coverage and robot placement," *Robotics and Computer-Integrated Manufacturing*, vol. 93, p. 102932, 2025.

[3] Q.-N. Nguyen, N. Adrian, and Q.-C. Pham, "Task-space clustering for mobile manipulator task sequencing," in *2023 IEEE Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3693–3699.

[4] F. Sukkar, J. Wakulicz, K. M. B. Lee, W. Zhi, and R. Fitch, "Multi-query robotic manipulator task sequencing with gromov-hausdorff approximations," *IEEE Transactions on Robotics*, 2025.

[5] B. Burgess-Limerick, J. Haviland, C. Lehnert, and P. Corke, "Reactive base control for on-the-move mobile manipulation in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2048–2055, 2024.

[6] Z. Li, Y. Niu, Y. Su, H. Liu, and Z. Jiao, "Dynamic planning for sequential whole-body mobile manipulation," in *2024 IEEE 19th Conf. on Industrial Electronics and Applications (ICIEA)*. IEEE, 2024, pp. 1–7.

[7] F. Reister, M. Grotz, and T. Asfour, "Combining navigation and manipulation costs for time-efficient robot placement in mobile manipulation tasks," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9913–9920, 2022.

[8] H. Yang, Z. Jiao, S. Wang, Y. Niu, S. Liu, and H. Liu, "Integrated exploration and sequential manipulation on scene graph with llm-based situated replanning," *arXiv preprint arXiv:2602.04419*, 2026.

[9] J. Wu, J. Li, S. Zhang, Z. He, Z. Wang, X. Leng, H. Liu, J. Zhang, J. Wang, S.-C. Zhu *et al.*, "Path and motion optimization for efficient multi-location inspection with humanoid robots," *arXiv preprint arXiv:2510.11401*, 2025.

[10] F. Kong, F. Du, and D. Zhao, "Station-viewpoint joint coverage path planning towards mobile visual inspection," *Robotics and Computer-Integrated Manufacturing*, vol. 91, p. 102821, 2025.

[11] R. Malhan and S. K. Gupta, "Finding optimal sequence of mobile manipulator placements for automated coverage planning of large complex parts," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 86212. American Society of Mechanical Engineers, 2022, p. V002T02A006.

[12] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, vol. 9, pp. 119 310–119 342, 2021.

[13] R. F. Cavelli, P. D. Cen Cheng, and M. Indri, "Modeling the reachability space of robotic manipulators through ellipsoid equations," *Journal of Intelligent & Robotic Systems*, vol. 111, no. 3, p. 90, 2025.

[14] ——, "Mobile manipulator base placement optimization using an ellipsoidal reachability model," in *2025 IEEE 30th Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2025, pp. 1–8.

[15] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan *et al.*, "Grounded sam: Assembling open-world models for diverse visual tasks," *arXiv preprint arXiv:2401.14159*, 2024.

[16] Trossen Robotics, *LoCoBot WX250*, Trossen Robotics, 2020, accessed: February 2026. [Online]. Available: https://docs.trossenrobotics.com/interbotix_xslocobots_docs/specifications/locobot_wx250s.html

[17] ROS, *ROS Noetic*, accessed: February 2026. [Online]. Available: https://wiki.ros.org/noetic

[18] "Github repository for experiments," [Accessed: February 2026]. [Online]. Available: https://github.com/Saro0800/base_seq_opt_multi_dynamic_env_SIM.git

[19] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4193–4198.

[20] "Simulation/experiments demo video," https://youtu.be/cm5PyPlF3X4, [Online; accessed March 2026].