International Journal of Advanced Engineering and Technology Research
ISSN: 3106-8677 (Print), ISSN: 3106-8685 (Online) | Vol. 1, No. 2, 2026

Diagnosis-Driven Stage-Wise Trajectory Planning for a Mobile Dual-Arm Robot in Confined Derrick Pin Disassembly

Ruiling Wang [1], Hao Wang [2]

[1] School of Mechanical and Electrical Engineering, Southwest Petroleum University, Chengdu 610500, China
[2] College of Mechanical Engineering, Henan University of Science and Technology, Luoyang 471003, China

Abstract: Derrick pin disassembly in drilling structures is a typical confined-space maintenance task that requires dual-arm cooperation, multi-stage motion transition, and strict safety control. Although a fixed workstation may satisfy pose-level reachability and collision-free requirements, executable full-process motion generation remains difficult because local safety-margin degradation, inter-arm interference, and blocked stage transitions may still occur during continuous planning. To address this problem, this paper proposes a diagnosis-driven stage-wise trajectory planning method for a mobile dual-arm robotic system operating under a fixed workstation input. The method first formulates the task as an ordered sequence of stage targets under unified safety constraints. A trajectory-risk diagnosis mechanism is then introduced to identify the minimum safety margin, bottleneck location, and dominant constraint relation of each stage trajectory. Based on the diagnosis result, a cascade repa ir framework is constructed by integrating local segment repair, endpoint-neighborhood repair, and global recovery with adaptive step size. After the geometric trajectory has been repaired and verified, piecewise quintic polynomial time parameterization is applied to generate smoother executable output. The proposed method is intended to bridge the gap between pose-level feasibility and full-process executability in confined dual-arm derrick operations. It provides a practical planning framework for converting fixed-workstation feasible results into continuous and task-consistent dual-arm motions under multiple geometric safety constraints.

Keywords: Mobile dual-arm robot, confined-space operation, derrick pin disassembly, stage-wise trajectory planning, trajectory-risk diagnosis, cascade repair, quintic time parameterization.

1. Introduction

Derrick pin disassembly and assembly are typical maintenance operations in drilling equipment. These operations are usually carried out in cluttered and partially occluded areas with limited accessible space, large variations in working height, and considerable safety risk. In current field practice, the procedure still relies heavily on manual experience, which leads to high labor intensity, unstable operation quality, and limited repeatability. With the increasing demand for intelligent and less human-dependent oilfield equipment, the automation of high-risk derrick maintenance tasks has become an important practical problem.

Compared with common pick-and-place or grasping tasks, derrick pin disassembly is more difficult to formulate as a direct robot motion problem. A pin is not a robot-executable control target by itself. Instead, the task object must first be transformed into key operation points and target end-effector poses with explicit action semantics. Meanwhile, the mobile chassis position, platform heading, lifting height, and telescopic extension jointly affect the base poses of the two manipulators, the reachable workspace, and the local manipulation capability. As a result, the planning problem is strongly coupled with the overall system configuration rather than determined by arm kinematics alone.

Another difficulty lies in the gap between pose-level feasibility and full-process executability. Even when the target poses are reachable and collision-free in a discrete sense, the continuous execution process may still fail because of environmental collision, inter-arm interference, local safety-margin degradation, or blockage at stage transitions. This issue is especially prominent in confined derrick environments, where narrow passages, dense structural members, and frequent interaction between the two manipulators make the planning problem highly constrained. Therefore, it is not sufficient to determine only whether a given workstation can support several key poses. A planning method is further required to transform pose-level feasible results into executable continuous dual-arm motions for the whole task process.

Existing studies have provided important foundations in mobile manipulation, dual-arm coordination, workstation placement, and motion planning. Mobile manipulators extend task coverage through base motion, while dual-arm systems provide advantages in role allocation, collaborative manipulation, and multi-stage task execution. In parallel, workstation determination has evolved from simple reachability judgment to more comprehensive evaluation involving kinematic quality, collision avoidance, and task-related performance. In trajectory planning, sampling-based and optimization-based methods have both shown effectiveness in high-dimensional robotic systems. However, most existing work treats target poses or workstation configurations as known inputs, and pays less attention to the specific gap between fixed-workstation pose feasibility and full-process executable motion generation in confined dual-arm maintenance tasks.

For derrick pin disassembly, this gap is non-negligible. The task is characterized by staged cooperation between the two arms, frequent switching of active and supporting roles, and multiple safety constraints arising from the environment, inter-arm proximity, and self-collision. Under these

conditions, local geometric feasibility does not guarantee global execution stability. A practical planning method should therefore not only search for collision-free connections, but also identify trajectory bottlenecks, repair locally risky segments, recover blocked transitions, and finally output a smooth and executable motion sequence.

To address this problem, this paper develops a diagnosis-driven stage-wise trajectory planning method for a mobile dual-arm robotic system performing derrick pin disassembly in confined environments. The method takes a fixed workstation obtained by prior screening as input, keeps the task-pose definition and unified safety constraints consistent with the upstream planning stages, and focuses on the transformation from pose-level feasible input to executable full-process output. A cascade repair mechanism is introduced to handle local safety-margin degradation, path-connectivity failure, and stage-transition obstruction during continuous planning. In this way, the proposed method provides a practical motion-generation framework for confined derrick operations with strong geometric and cooperative constraints.

The remainder of this paper is organized as follows. Section 2 describes the robotic system, the task representation, and the planning problem. Section 3 briefly introduces the fixed workstation input used for continuous planning. Section 4 presents the proposed diagnosis-driven stage-wise trajectory planning method. The subsequent sections provide implementation details, case-based validation, and concluding remarks.

2. System Description and Problem Formulation

2.1. Robotic System and Task Scenario

The robotic platform considered in this study is a mobile dual-arm operation system designed for derrick pin disassembly and assembly in confined spaces. The system consists of a mobile chassis, a lifting mechanism, a telescopic mechanism, and two manipulators mounted on the upper platform. The mobile chassis provides large-range access to different operation regions. The lifting and telescopic mechanisms adjust the vertical and longitudinal positions of the working platform. The two manipulators then perform coordinated local operations around the target pin.



Figure 1. Mobile dual-arm robotic system and confined derrick pin disassembly scenario

This configuration combines mobility and manipulation capability. The mobile base is responsible for coarse positioning with respect to the derrick structure, while the upper mechanisms and the two arms handle fine approach, support-operation coordination, and stage transitions. Such a configuration is suitable for derrick tasks because the operation points are spatially distributed, partially occluded, and often surrounded by dense structural members. A fixed-base or single-arm solution generally cannot simultaneously satisfy workspace coverage, pose adjustment flexibility, and cooperative execution requirements.

In the target task, the two manipulators do not simply move independently toward a common goal. Instead, they act in a stage-wise cooperative manner. During pin disassembly, one arm first approaches the working region and performs the pushing or loosening action while maintaining local stability of the pin, and the other arm subsequently approaches the pin for grasping and extraction. The assembly process follows the reverse logic. Therefore, the task is intrinsically sequential and cooperative, rather than a single-shot end-point reaching problem. This characteristic directly motivates a stage-wise motion-planning formulation.

2.2. Operation-pose Representation and Task Stages

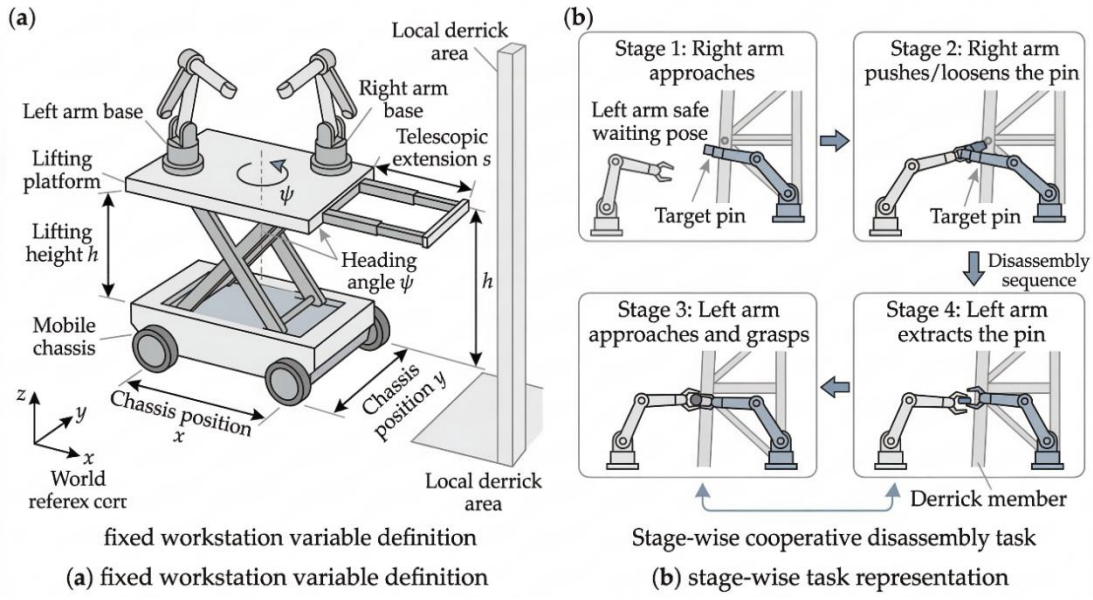


Figure 2. Fixed workstation variable and stage-wise task representation

To describe the global operating condition of the robot, a unified operation-pose variable is introduced as

$$\boldsymbol{\xi} = [x, y, s, h, \psi]^{\mathrm{T}} \tag{1}$$

Where x and y denote the planar position of the mobile chassis, s denotes the telescopic extension, h denotes the lifting height, and ψ denotes the platform heading. This variable provides a compact description of the robot workstation configuration. Once $\boldsymbol{\xi}$ is given, the base poses of the left and right manipulators can be determined accordingly, and the subsequent task poses are evaluated under the same geometric frame.

The derrick pin itself is not directly treated as a robot control input. Instead, task-oriented target poses are generated from the pin geometry, its installation location, and the action semantics of disassembly or assembly. In other words, the task object is transformed into robot-executable end-effector poses through an intermediate representation of key operation points. This transformation makes it possible to connect object-level task semantics with manipulator-level motion planning.

Because the disassembly and assembly processes are stage-dependent, the full task is decomposed into several ordered planning stages. Each stage corresponds to a different local objective, such as approach, posture adjustment, pushing or loosening, grasping, extraction, insertion, and return. Let the target pose set for the k-th stage be denoted by $\mathcal{T}_k$. Then the full planning problem is not to generate one single path between a start and an end configuration, but to construct a sequence of collision-free and executable trajectory segments that connect the stage targets in order. This stage-wise structure is essential because failure frequently occurs not only inside a segment, but also at the transition between neighboring stages.

2.3. Unified Safety Constraints and Planning Objective

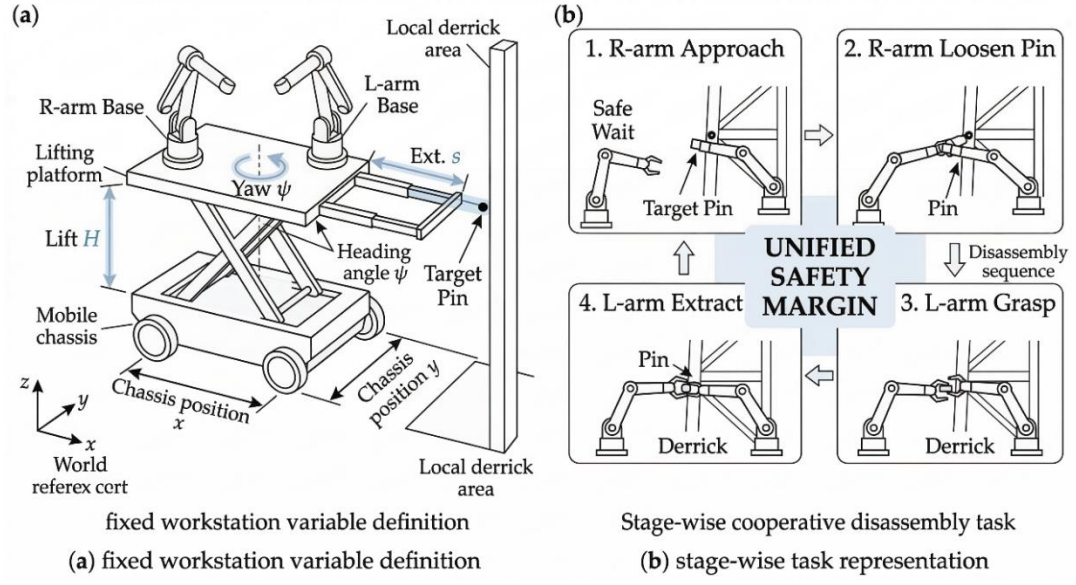


Figure 3. Unified safety constraints for dual-arm trajectory planning

The confined derrick environment introduces multiple classes of safety constraints. In this work, three types of

distances are considered in a unified manner: the distance between the robot and the environment, the distance between the two manipulators, and the self-collision distance within each manipulator system. Let these distances be denoted respectively by d_{env}, d_{inter}, and d_{self}. For a given robot configuration $\mathbf{q}$, a unified safety margin can be written as

$$\mu(\mathbf{q}) = \min\{d_{env}(\mathbf{q}) - d_{env}^{\min}, d_{inter}(\mathbf{q}) - d_{inter}^{\min}, d_{self}(\mathbf{q}) - d_{self}^{\min}\} \quad (2)$$

Where $d_{env}^{\min}$, $d_{inter}^{\min}$, and $d_{self}^{\min}$ are the corresponding minimum allowable safety thresholds. A positive value of $\mu(\mathbf{q})$ indicates that all safety requirements are satisfied at the configuration level, while a negative value indicates a safety violation. For a trajectory τ, the minimum safety margin along the whole path is defined as

$$\mu_{\min}(\tau) = \min_{\mathbf{q}\in\tau} \mu(\mathbf{q}) \quad (3)$$

This unified expression is useful because it provides a consistent criterion for diagnosis, repair, and final path verification. Instead of treating different collision relations separately at every planning step, the planner can identify the most critical bottleneck through the minimum safety margin and then trigger the corresponding repair mechanism.

Under a fixed workstation input $\boldsymbol{\xi}^*$, the planning objective of this study can be stated as follows: generate a stage-wise dual-arm continuous trajectory set

$$\mathrm{T} = \{\tau_1, \tau_2, \ldots, \tau_K\} \quad (4)$$

Table 1. Main variables and symbols used in the planning formulation

Symbol	Meaning
$\xi = [x, y, s, h, \psi]^{\mathrm{T}}$	Fixed workstation variable
x	Planar chassis position in the x-direction
y	Planar chassis position in the y-direction
s	Telescopic extension of the upper platform
h	Lifting height of the upper platform
ψ	Heading angle of the platform
$\mathcal{T}_s$	Target set of the s-th task stage
τ_s	Trajectory segment of the s-th stage
q	Dual-arm joint configuration
$d_{env}(\mathrm{q})$	Robot–environment distance
$d_{inter}(\mathrm{q})$	Inter-arm distance
$d_{self}(\mathrm{q})$	Self-collision distance
$\mu(\mathrm{q})$	Unified safety margin at configuration q
$\mu_{\min}(\tau_s)$	Minimum safety margin along τ_s
k_s^*	Bottleneck index of the s-th stage trajectory
κ_s^*	Dominant bottleneck constraint of the s-th stage
k_s^{vio}	First violation index of the s-th stage trajectory
κ_s^{vio}	Dominant violation type of the s-th stage

Such that each trajectory segment satisfies joint constraints and geometric safety constraints, all neighboring stages are continuously connectable, and the final output remains executable throughout the full process. Therefore, the problem addressed here is not workstation optimization itself, but the conversion of a pose-level feasible workstation result into a verified continuous motion output under unified safety constraints.

3. Fixed Workstation Input for Continuous Planning

The continuous planner developed in this paper does not search the workstation from scratch. Instead, it takes a fixed workstation obtained by prior screening as the geometric input of the planning stage. This design follows the practical decomposition of the overall derrick task. Before continuous planning is attempted, candidate workstations are generated in the operation-pose search space and then screened sequentially according to endpoint reachability, endpoint safety, and discrete transition feasibility. The feasible workstation set is finally ranked using a unified safety margin, and the best available workstation is selected as the fixed input for subsequent planning.

This treatment has two advantages. First, it isolates the continuous planning problem from large-scale workstation search, which significantly reduces the planning scope in the final motion-generation stage. Second, it ensures consistency between upstream pose-level feasibility analysis and downstream trajectory generation, because the same task-pose definition and safety-constraint representation are preserved. In this way, the proposed method focuses on the core issue that remains after workstation determination, namely, whether the selected pose-level feasible workstation can actually support a safe and executable full-process dual-arm motion.

It should be emphasized that a feasible workstation does not automatically guarantee executable continuous motion. The upstream screening only indicates that the target poses and several key transitions are feasible in a discrete sense. During actual continuous planning, however, local safety-margin drops, narrow-passage effects, inter-arm coupling, and blocked stage transitions may still appear. Therefore, the fixed workstation should be regarded as a reliable planning input rather than a complete solution. This observation is the starting point of the method proposed in the next section.

Based on the above formulation, the remaining task is to generate, diagnose, repair, and smooth the dual-arm trajectories under the selected workstation input. The following section introduces the diagnosis-driven stage-wise planning framework developed for this purpose.

4. Proposed Method

4.1. Stage-wise Planning Framework

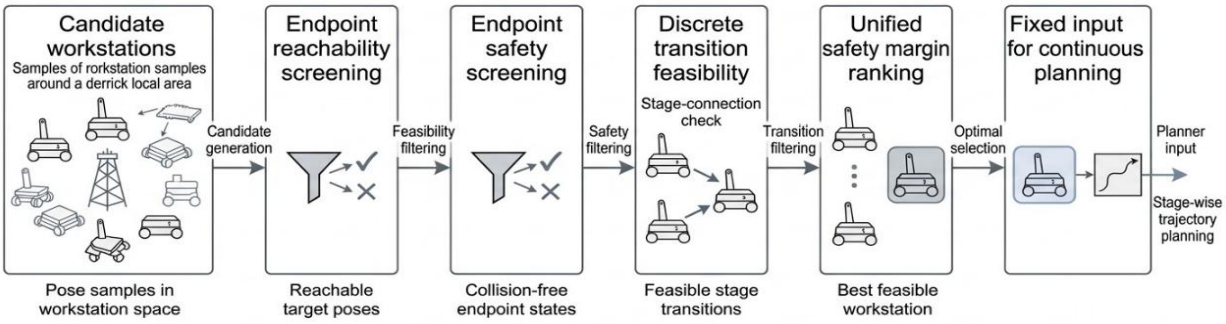


Figure 4. Overall framework of the proposed diagnosis-driven stage-wise trajectory planning method

Under the fixed workstation input determined in the previous stage, the proposed method focuses on generating an executable continuous dual-arm motion for the whole derrick pin disassembly or assembly process. Instead of treating the task as a single global connection problem, the full operation is decomposed into ordered planning stages according to task semantics and arm-role switching. In each stage, the dual-arm system is assigned a stage target, and the planner attempts to construct a feasible trajectory segment in joint space. The final output is therefore a sequence of stage trajectories connected in order, rather than one monolithic path from an initial configuration to a terminal configuration. This stage-wise organization is consistent with the method positioning in your thesis: the optimal workstation is fixed first, and then continuous planning is performed under unchanged task-pose and unified-safety definitions.

Let the full task contain Sstages, and let the trajectory of the s-th stage be denoted by τ_s. The overall planning output can then be written as

$$\mathcal{T} = \{\tau_1, \tau_2, \dots, \tau_S \tag{5}$$

For each stage, a nominal trajectory is first generated under the current boundary conditions. This nominal result is not immediately accepted as the final output. Instead, it is passed to a diagnosis-driven cascade repair procedure, which evaluates whether the segment is safe and continuously executable under the unified safety constraints. If no failure is detected, the trajectory is kept. Otherwise, the planner progressively triggers local segment repair, endpoint-neighborhood repair, and finally global recovery. In this sense, the method is not a simple one-shot planner but a staged planning-and-repair framework designed to transform pose-level feasibility into executable continuous output. That is also exactly how Chapter 5 is positioned in your thesis.

4.2. Trajectory-risk Diagnosis and Bottleneck Identification

The key idea of DDCR is that trajectory failure should not be treated as an undifferentiated "planning failure." Instead, each stage trajectory is first diagnosed to determine where the risk is concentrated and what constraint is dominant. In your thesis, the diagnosis output is explicitly organized around five quantities: the minimum safety margin of the stage trajectory, the bottleneck position index, the dominant bottleneck relation, the first violation index, and the dominant violation type. These fields provide the basis for the subsequent repair hierarchy.

For a discretized stage trajectory $\tau_s = \{\mathbf{q}_{s,0}, \mathbf{q}_{s,1}, \dots, \mathbf{q}_{s,N_s}\}$, the minimum safety margin is defined as

$$\mu_{\min}(\tau_s) = \min_{0 \le k \le N_s} \mu(\mathbf{q}_{s,k}) \tag{6}$$

Where $\mu(\mathbf{q}_{s,k})$is the unified safety margin introduced in Section 2. The bottleneck index of the stage is then

$$k_s^* = \arg \min_{0 \le k \le N_s} \mu(\mathbf{q}_{s,k}) \tag{7}$$

And the dominant bottleneck relation is denoted by

$$\kappa_s^* = \kappa(\mathbf{q}_{s,k_s^*}) \tag{8}$$

Where $\kappa(\cdot)$identifies the tightest constraint source at the current sample, such as robot–environment proximity, inter-arm interference, or self-collision. These expressions are a compact English reformulation of the diagnosis quantities already defined in your thesis and listed in Table 5-3.

When a trajectory segment violates the safety requirement, DDCR further identifies the first location at which the violation appears. This quantity is used to distinguish early failure from merely having a narrow bottleneck near the boundary of feasibility. The first violation index is written as

$$\mathrm{k_s^{vio}} = \min \{\mathrm{k} \mid 0 \le \mathrm{k} \le \mathrm{N_s}\} \tag{9}$$

And the dominant violation type is

$$\kappa_s^{\mathrm{vio}} = \kappa(\mathbf{q}_{s,k_s^{\mathrm{vio}}}) \tag{10}$$

Ogether, $\mu_{\min}(\tau_s)$, k_s^*, κ_s^*, k_s^{vio}, and κ_s^{vio} convert a nominal stage trajectory into structured repair evidence. This is the central methodological move in your Chapter 5: the planner does not immediately rebuild the whole segment, but first diagnoses whether the problem is concentrated in a local bottleneck region, in the endpoint neighborhood, or in a broader stage-level connectivity failure.

Table 2. Diagnosis outputs and corresponding repair actions in the proposed method

Diagnosis output / condition	Typical failure characteristic	Triggered action
$\mu_{\min}(\tau_s) \geq 0$	Stage trajectory satisfies unified safety constraints	Accept the nominal stage trajectory
Bottleneck located in the interior segment, while stage boundaries remain feasible	Local safety-margin degradation concentrated near a bottleneck window	Local segment repair
Endpoint neighborhood is unsafe or unsuitable for stable connection	Boundary-related infeasibility near stage start or end	Endpoint-neighborhood repair
Local repair fails to remove the bottleneck	Interior conflict exceeds the coverage of local replacement	Escalate to endpoint-neighborhood repair or global recovery
Endpoint repair fails to restore feasible stage connection	Boundary reselection is insufficient	Global recovery
Broader stage-level connectivity failure	Failure cannot be resolved by local or boundary repair	Global recovery with adaptive step size
Recovered stage passes unified safety check	Stage trajectory becomes feasible after repair	Accept repaired trajectory and continue to next stage
Recovered stage still violates safety constraints	No executable stage output obtained	Mark stage as unresolved / terminate stage chain

4.3. Local Segment Repair

If the diagnosis result shows that the failure is concentrated around the bottleneck neighborhood while the stage endpoints remain acceptable, DDCR first invokes local segment repair. The motivation is explicit in your thesis: directly switching to global recovery would increase computational cost and unnecessarily destroy the portions of the trajectory that have already passed verification. Therefore, the first repair layer keeps the stage boundaries unchanged and only adjusts a local window around the bottleneck. Its purpose is not to reconstruct the entire path, but to restore safety and passability with the minimum possible modification range.

Let the bottleneck-centered window be denoted by $[k_s^-, k_s^+]$, where k_s^- and k_s^+ are selected around k_s^*. The local repair process replaces the original sub-trajectory

$$\tau_s^{\mathrm{loc}} = \{\mathbf{q}_{s,k_s^-}, \dots, \mathbf{q}_{s,k_s^+}\} \tag{11}$$

With a repaired local segment

$$\hat{\tau}_s^{\mathrm{loc}} = \{\hat{\mathbf{q}}_{s,k_s^-}, \dots, \hat{\mathbf{q}}_{s,k_s^+}\} \tag{12}$$

While preserving the original stage start and end conditions. In implementation terms, the replacement segment is generated by directed adjustment in the bottleneck neighborhood rather than by globally re-running the whole stage planner. This matches the logic shown in your thesis, where local repair is defined as a bottleneck-neighborhood adjustment under unchanged endpoints.

The advantage of this design is twofold. First, it preserves the already feasible non-bottleneck portions of the stage trajectory. Second, it provides a targeted remedy for geometry-driven local failures, such as local obstacle attachment or short-range inter-arm proximity. In the method narrative, local repair therefore acts as the first and least disruptive repair layer. Only when this layer fails does the planner move to the next level.

4.4. Endpoint-neighborhood Repair

The second repair layer is used when the diagnosis indicates that the failure is associated with the stage start or end neighborhood rather than with an interior bottleneck segment. In your thesis, this mechanism is clearly differentiated from local segment repair: local repair keeps the boundaries fixed and adjusts the middle portion, whereas endpoint repair does not change the stage semantics but reselects the boundary condition itself. This distinction is important and should be kept in the English paper because it is one of the clearest structural features of your DDCR method.

Suppose the original stage boundary pair is $(\mathbf{q}_s^{\mathrm{st}}, \mathbf{q}_s^{\mathrm{ed}})$. Endpoint-neighborhood repair searches for an alternative admissible boundary pair

$$(\hat{\mathbf{q}}_s^{\mathrm{st}}, \hat{\mathbf{q}}_s^{\mathrm{ed}}) \tag{13}$$

Within the allowable endpoint neighborhood, while preserving the stage action semantics. For insertion- or extraction-related segments, the candidate endpoints must additionally satisfy the pose-corridor or directional constraints defined at the task level, so that endpoint replacement does not destroy the required motion semantics. This condition is stated explicitly in your thesis and is useful in the English paper because it shows that endpoint repair is not a free geometric perturbation but a task-consistent boundary replacement.

After endpoint replacement, the trajectory is not directly accepted. Instead, the stage with the updated boundary condition is fed back into the fast diagnostic screening procedure. If the nominal interpolated trajectory under the new endpoint pair can pass the unified safety check, the stage is resolved at the endpoint-repair layer. Otherwise, the updated diagnosis quantities are retained and the planner escalates to the final global recovery layer. In this way, local repair and endpoint repair form a clean two-level hierarchy: the former addresses interior geometric conflict, and the latter addresses unusable or unsafe boundary configurations.

4.5. Global Recovery with Adaptive Step Size

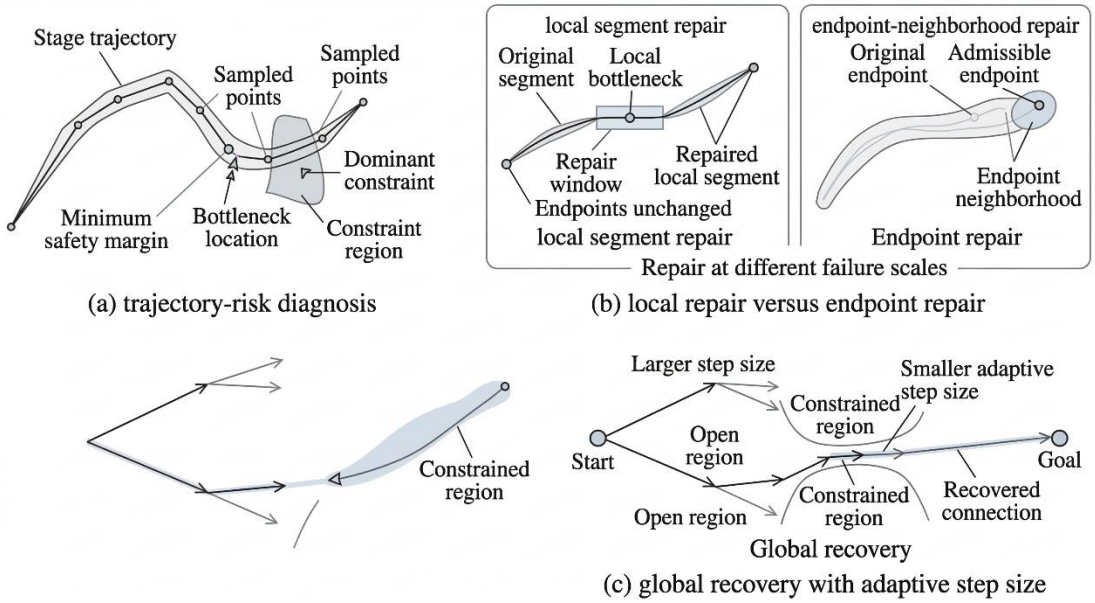


Figure 5. Trajectory-risk diagnosis and cascade repair logic

When neither local segment repair nor endpoint-neighborhood repair is able to restore stage feasibility, the risk source is regarded as exceeding the coverage of local replacement and boundary reselection. At this point, DDCR activates the final recovery layer, namely global reconnection between the stage endpoints. In the structure of your thesis, this is the last repair stage and corresponds to "global recovery and continuous connection restoration based on envelope-margin adaptive step size."

The purpose of this layer is not merely to retry planning with different random seeds. Instead, it rebuilds the stage-level connection under a more conservative and risk-aware expansion strategy. The use of adaptive step size is motivated by the fact that confined derrick cases often contain narrow bottleneck regions where an aggressive global expansion can repeatedly fail. By linking the step size to the local envelope or safety-margin condition, the planner can remain relatively efficient in open regions while becoming more cautious near tight constraints. This description is a direct methodological inference from your section title and the surrounding explanation of why the global layer is only triggered after the two lighter repair layers fail.

Once a stage is globally reconnected, the repaired segment is again subjected to unified diagnosis and checking. In this way, all three repair layers share a common output interface: trigger status, repair success flag, repaired minimum safety margin, and updated diagnosis quantities. This consistent interface is useful for a paper version because it lets the whole method read as a coherent cascade rather than as several disconnected heuristics. That consistency is also reflected in your original thesis text, where diagnosis, local repair, endpoint repair, and final global recovery are explicitly organized as one stage-level planning chain.

4.6. Trajectory Smoothing and Final Output

After all stage segments have passed diagnosis and repair, the resulting geometric trajectories are not used directly as execution commands. As stated in your abstract and Chapter 5 structure, piecewise quintic polynomial interpolation is applied for time parameterization of the output trajectory. The purpose of this step is not to alter the geometric trend of the planned path or the associated safety-constraint relationships, but to improve continuity and execution smoothness in the final output. In the English paper, this part should be written as the final output layer rather than as a separate innovation point.

For the s-th repaired geometric stage trajectory, let the joint path be denoted by $\mathbf{q}_s(\lambda)$, where $\lambda \in [0,1]$ is a geometric path parameter. Time parameterization introduces a smooth mapping $\lambda = \lambda(t)$, so that the final executable joint trajectory becomes

$$\mathbf{q}_s(t) = \mathbf{q}_s(\lambda(t)), t \in [t_s^0, t_s^f] \tag{14}$$

Using a piecewise quintic form for $\lambda(t)$ makes it possible to keep the trajectory position, velocity, and acceleration profiles smoother at stage boundaries. This is fully consistent with the role assigned to quintic interpolation in your thesis: it is the final refinement layer that converts a geometrically verified trajectory into a smoother executable motion sequence without changing the verified safety relation of the original path.

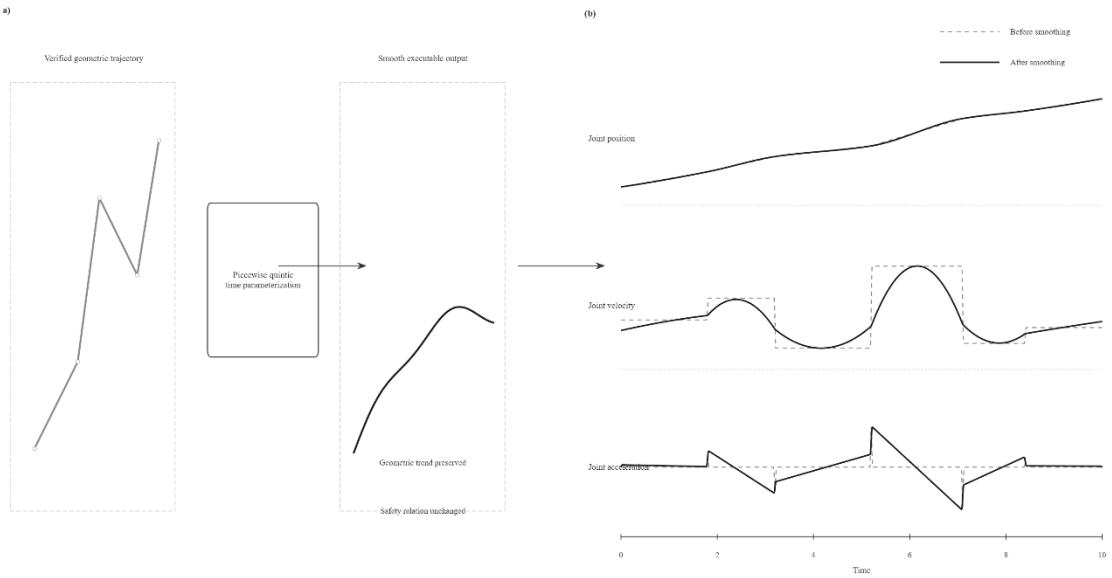


Figure 6. Piecewise quintic time parameterization for final trajectory smoothing

5. Implementation Procedure and Method Characteristics

5.1. Overall Execution Logic

The proposed method is executed after a feasible workstation has been determined at the pose level. Accordingly, the input of the planner is not a raw task object or an unconstrained scene description, but a fixed workstation configuration, a sequence of stage targets, the initial dual-arm configuration, and the unified safety thresholds. Under this setting, the planner does not attempt to solve workstation search and continuous motion generation simultaneously. Instead, it focuses on the remaining problem, namely, how to transform a pose-level feasible workstation result into a safe and executable full-process dual-arm trajectory. This positioning is fully consistent with the original thesis, in which the optimal workstation output of Chapter 4 is explicitly used as the fixed input of Chapter 5, while the continuous planner keeps the task-pose definition and unified safety constraints unchanged.

At the implementation level, the planner proceeds stage by stage. For each stage, a nominal trajectory is first generated according to the current boundary conditions and the task-role assignment of the two manipulators. The resulting segment is then screened by the trajectory-level diagnosis module. If the segment passes the unified safety check, it is accepted as the stage output. If not, the diagnosis result is passed to the cascade repair chain. The planner first attempts local segment repair, then endpoint-neighborhood repair, and finally global recovery with adaptive step size if the previous two layers fail. After all stages have been resolved, the geometric paths are concatenated in sequence and subsequently smoothed by piecewise quintic time parameterization. In this way, the proposed framework maintains a clear progression from nominal generation to diagnosis, layered repair, global reconnection, and final executable output. This progression follows the method organization already established in your thesis framework figure and Chapter 5 headings.

5.2. Stage-level Planning Procedure

To make the method easier to implement and present in a journal paper, the planning logic can be summarized as a stage-level iterative procedure. Let the fixed workstation be $\boldsymbol{\xi}^*$, and let the ordered stage target set be $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_S\}$. The planner starts from the initial dual-arm configuration $\mathbf{q}^0$ associated with the fixed workstation. For the first stage, it generates a nominal trajectory from $\mathbf{q}^0$ to a feasible configuration associated with $\mathcal{T}_1$. For each subsequent stage, the terminal configuration of the previous stage becomes the starting boundary condition of the current stage. The resulting stage output therefore has an inherently sequential dependency structure, which reflects the staged task organization emphasized in the thesis for disassembly and assembly operations.

Formally, the stage-level procedure can be expressed as follows. For the s-th stage, a nominal segment τ_s^0 is first generated:

$$\tau_s^0 = \mathcal{P}(\mathbf{q}_s^{\mathrm{st}}, \mathbf{q}_s^{\mathrm{ed}}, \boldsymbol{\xi}^*, \mathcal{T}_s) \tag{15}$$

Where $\mathcal{P}(\cdot)$ denotes the nominal stage planner, $\mathbf{q}_s^{\mathrm{st}}$ is the stage start configuration, and $\mathbf{q}_s^{\mathrm{ed}}$ is the stage target boundary candidate. The nominal segment is then diagnosed:

$$\mathcal{D}_s = \mathcal{D}(\tau_s^0) \tag{16}$$

Where $\mathcal{D}(\cdot)$ returns the minimum safety margin, bottleneck index, dominant bottleneck relation, first violation index, and dominant violation type. If $\mathcal{D}_s$ satisfies the acceptance condition, that is,

$$\mu_{\min}(\tau_s^0) \geq 0 \tag{17}$$

The nominal segment is accepted directly. Otherwise, the planner enters the cascade repair chain:

$$\tau_s = \begin{cases} \mathcal{R}_{\mathrm{loc}}(\tau_s^0, \mathcal{D}_s), & \text{if local repair succeeds,} \\ \mathcal{R}_{\mathrm{end}}(\tau_s^0, \mathcal{D}_s), & \text{if endpoint repair succeeds,} \\ \mathcal{R}_{\mathrm{glob}}(\tau_s^0, \mathcal{D}_s), & \text{otherwise.} \end{cases} \tag{18}$$

This formulation is not intended as a new theoretical abstraction beyond your thesis; it is simply the compact paper-style restatement of the diagnosis-repair chain already defined by the Chapter 5 structure and its associated output fields.

After a repaired stage trajectory τ_s is obtained, the planner performs a final unified check before passing its terminal configuration to the next stage. If no feasible repaired segment can be produced, the stage is marked as unresolved and the full process is terminated at that point. In practical

manuscript writing, this rule is useful because it makes the method read as a deterministic workflow rather than a loose combination of heuristics. It also aligns with the original role of Chapter 5 in your thesis: the point is not just to "try several repairs," but to construct a complete pose-to-trajectory conversion procedure under fixed workstation input.

5.3. Practical Meaning of The Three Repair Layers

One strength of the proposed method is that the three repair layers are not redundant. Each layer addresses a different failure scale.

The local segment repair layer is intended for failures concentrated around an interior bottleneck region. In this case, the stage endpoints remain meaningful and only a limited neighborhood around the bottleneck needs to be replaced. This avoids unnecessary destruction of the already feasible parts of the trajectory and keeps the adjustment range small. In the context of confined derrick motion, such failures often correspond to short-range geometric conflicts caused by local obstacle attachment, temporary inter-arm proximity, or a narrow passage located away from the stage boundaries. The role of this layer in your thesis is exactly to preserve most of the nominal stage path while lifting the local safety margin around the bottleneck.

The endpoint-neighborhood repair layer is used when the major difficulty is not in the interior part of the stage but near the stage entry or exit. In such cases, the original stage boundary may be kinematically valid in a discrete sense but unsuitable for stable continuous connection. Instead of forcing the planner to reconnect through an unfavorable boundary condition, the method searches for a nearby admissible endpoint pair that still preserves the stage semantics. This is especially important for insertion, extraction, and approach-related segments, because the endpoint is not a purely geometric waypoint but a task-constrained motion boundary. Your thesis distinguishes this layer from local segment repair very clearly, and that distinction should be retained in the paper because it is one of the most readable parts of the method design.

The global recovery layer is the final safeguard. Once both local bottleneck adjustment and endpoint reselection fail, the planner treats the current stage as a broader connectivity problem and rebuilds the segment under a more cautious expansion strategy. In the original thesis, this layer is explicitly described as global recovery and continuous connection restoration based on envelope-margin adaptive step size. In the paper version, the essential point is that the planner becomes conservative near tight constraints and relatively efficient in open regions. This design is not merely a fallback retry. Rather, it is the mechanism that allows the stage planner to remain usable in confined environments where narrow bottleneck regions repeatedly break nominal stage connections.

From a manuscript perspective, the practical value of this three-layer design is that it gives the method a clear inner logic: fix the stage if the problem is local, replace the boundary if the problem is at the interface, and rebuild the connection if the problem is global. That is much easier for a journal reviewer to understand than a single undifferentiated planning loop. It is also more faithful to your thesis than writing the method as a generic "improved RRT," because your actual contribution is the diagnosis-driven repair organization rather than a single sampling formula.

5.4. Role of Time Parameterization in The Final Output

In the final stage of the method, the verified geometric trajectories are transformed into executable joint trajectories through piecewise quintic polynomial time parameterization. In the thesis, this step is not introduced as an independent planning method, but as the final output processing layer applied after the diagnosis-repair chain has already produced a safe geometric path. That role should remain unchanged in the journal manuscript. The quintic interpolation is therefore not presented here as the core contribution, but as a necessary output refinement mechanism.

The practical reason for this design is straightforward. A trajectory that is geometrically feasible under collision and task constraints may still contain local corners, abrupt joint variation, or non-smooth transitions between neighboring stages. Such defects do not necessarily invalidate geometric feasibility, but they reduce execution quality and make the final output less suitable for real robotic motion. By introducing a smooth time law $\lambda(t)$over each geometric path segment, the output trajectory can maintain continuity in position, velocity, and acceleration at the stage boundaries:

$$\mathbf{q}_s(t) = \mathbf{q}_s(\lambda(t)), t \in [t_s^0, t_s^f] \tag{19}$$

In your thesis, the piecewise quintic polynomial is written explicitly as the interpolation model of each joint segment, and the accompanying explanation states that the purpose is to smooth the trajectory without changing the overall geometric feasibility and safety relationship of the repaired path. That exact role is the one retained here.

5.5. Method Scope and Paper-Level Positioning

For this journal version, the method should be positioned as a trajectory-generation framework under fixed workstation input, rather than a full-chain system paper covering object modeling, workstation search, planning, and large-scale validation with equal emphasis. Your thesis clearly contains the full chain from system modeling and task-pose construction to workstation determination and continuous planning, but it also explicitly states that the core role of Chapter 4 is to provide a stable pose-level input for Chapter 5, while Chapter 5 is responsible for transforming that input into executable continuous motion output. This internal relation is exactly why the method can be extracted as an independent journal paper.

Accordingly, the present manuscript focuses on the following claim: given a fixed workstation that is already feasible at the pose level, the proposed diagnosis-driven stage-wise planner can bridge the gap between discrete feasibility and full-process executability in confined dual-arm derrick operations. This focus is narrower than the full thesis scope, but it is clearer as a journal-paper contribution and better matches the structure of the method itself.

6. Conclusion

This paper presented a diagnosis-driven stage-wise trajectory planning method for a mobile dual-arm robotic system performing derrick pin disassembly in confined environments. Instead of treating pose-level feasibility as the final planning outcome, the method addressed the remaining gap between fixed-workstation feasibility and full-process

executable motion generation. Under a workstation input obtained in advance, the planner maintained a unified task-pose and safety-constraint representation and organized continuous motion generation as a sequence of stage-level planning, diagnosis, repair, reconnection, and smoothing steps. This problem formulation and method scope are directly consistent with the final methodological contribution identified in the original thesis.

The central idea of the method is to replace undifferentiated planning failure handling with structured trajectory-risk diagnosis and cascade repair. For each stage trajectory, the planner first identifies the minimum safety margin and the dominant bottleneck relation, then chooses among local segment repair, endpoint-neighborhood repair, and global recovery according to the scale and location of failure. After geometric feasibility is restored, piecewise quintic polynomial interpolation is applied to generate a smoother executable output trajectory. In this way, the method provides a complete conversion route from pose-level feasible input to task-consistent continuous dual-arm motion under confined constraints.

From the perspective of manuscript contribution, the proposed framework is not intended as a generic universal motion planner. Its value lies in providing a practically organized planning method for a specific class of confined dual-arm maintenance tasks in which task stages, arm-role switching, safety-margin degradation, and interface blocking must be considered together. For the derrick pin disassembly problem addressed here, the method offers a clear planning logic and a consistent output form that can connect upstream workstation determination with downstream executable motion generation.

References

[1] Sandakalum T, Ang M H Jr. Motion Planning for Mobile Manipulators—A Systematic Review. Machines, 2022, 10(2): 97.

[2] Abbas M, Narayan J, Dwivedy S K. A systematic review on cooperative dual-arm manipulators: modeling, planning, control, and vision strategies. International Journal of Intelligent Robotics and Applications, 2023, 7(4): 683–707.

[3] Sereinig M, Manzl P, Gerstmayr J. Task-Dependent Comfort Zone, a Base Placement Strategy for Mobile Manipulators Based on Manipulability Measures. Robotics, 2024, 13(8): 122.

[4] Shin D H, Hamner B, Singh S, Hwangbo M. Motion Planning for a Mobile Manipulator with Imprecise Locomotion. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2003: 847–853.

[5] Zhang H, Sheng Q, Sun Y, Sheng X, Xiong Z, Zhu X. A novel coordinated motion planner based on capability map for autonomous mobile manipulator. Robotics and Autonomous Systems, 2020, 129: 103554.

[6] Yamazaki K, Suzuki S, Kuribayashi Y. Approaching motion planning for mobile manipulators considering the uncertainty of self-positioning and object's pose estimation. Robotics and Autonomous Systems, 2022, 158: 104232.

[7] Takahama T, Nagatani K, Tanaka Y. Motion Planning for Dual-arm Mobile Manipulator—Realization of "Tidying a Room Motion". In: Proc. IEEE International Conference on Robotics and Automation (ICRA), 2004: 4338–4343.

[8] Cohen B, Chitta S, Likhachev M. Single- and Dual-Arm Motion Planning with Heuristic Search. The International Journal of Robotics Research, 2014, 33(3): 305–320.

[9] Rodríguez C, Montaño A, Suárez R. Planning manipulation movements of a dual-arm system considering obstacle removing. Robotics and Autonomous Systems, 2014, 62(12): 1816–1826.

[10] Orthey A, Chamzas C, Kavraki L E. Sampling-Based Motion Planning: A Comparative Review. Annual Review of Control, Robotics, and Autonomous Systems, 2024, 7(1): 285–310.

[11] Kuffner J J, LaValle S M. RRT-Connect: An efficient approach to single-query path planning. In: Proc. IEEE International Conference on Robotics and Automation (ICRA), 2000, 2: 995–1001.

[12] Karaman S, Frazzoli E. Sampling-based Algorithms for Optimal Motion Planning. The International Journal of Robotics Research, 2011, 30(7): 846–894.

[13] Gammell J D, Srinivasa S S, Barfoot T D. Informed RRT*: Optimal Sampling-based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014: 2997–3004.

[14] Ratliff N, Zucker M, Bagnell J A, Srinivasa S. CHOMP: Gradient Optimization Techniques for Efficient Motion Planning. In: Proc. IEEE International Conference on Robotics and Automation (ICRA), 2009: 489–494.

[15] Schulman J, Duan Y, Ho J, Lee A X, Awwal I, Bradlow H, Pan J, Patil S, Goldberg K, Abbeel P. Motion planning with sequential convex optimization and convex collision checking. The International Journal of Robotics Research, 2014, 33(9): 1251–1270.