Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Study on Dynamic Obstacle Avoidance Using the UR5 Robot Arm

□□ □
□□□□□□□□
□□
549152121@qq.com

Sichuan University of Science and Engineering

Research Article

Keywords: UR5 robotic arm, dynamic obstacle avoidance, model predictive control, fast expanded random tree, artificial potential field, online motion planning

Posted Date: August 18th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10707930/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

A Study on Dynamic Obstacle Avoidance Using the UR5 Robot Arm

Duan Jingye[1]·Tan Fei[1] *

[1] Sichuan Provincial Key Laboratory of Intelligent Perception and Control, School of Automation and Information Engineering, Sichuan University of Science and Engineering, Yibin, Sichuan 644000, China;

*Corresponding author(s). email: duanjingye123@gmail.com

Contributing authors: email: duanjingye123@gmail.com, 549152121@qq.com

ORCID: 0009-0003-0503-6942

†These authors contributed equally to this work.

Abstract

Collaborative robotic arms operating in shared dynamic workspaces must balance safety, trajectory smoothness, and real-time performance. Sampling-based global planning typically assumes a static environment; reactive methods, while computationally efficient, lack foresight; and model predictive control (MPC) on high-degree-of-freedom robotic arms is limited by the speed of online solution.This paper proposes a hierarchical online obstacle avoidance framework for a six-degree-of-freedom UR5 robotic arm, comprising offline planning, online optimization, and pre-submission safety verification. RRT* generates a collision-free global reference in a static environment; an Artificial Potential Field (APF)-based reference pre-bending adjusts the trajectory in advance based on the nominal motion trends of dynamic obstacles, shifting part of the obstacle avoidance burden to the offline phase;The online layer employs a two-layer soft-constrained linear MPC to coordinate the preferred distance and hard safety boundaries through differentiated relaxation weights; safety gating based on capsule-based gap scanning independently verifies commands and triggers a three-level fallback mechanism. MATLAB simulations show that, for comparable levels of safety and task completion, the joint jerk of this method is only 1/2.59 that of the APF method;No hard safety boundary breaches occurred in 100 random encounters, with a 100% success rate. The median and 95th percentile online solution times were 7.7 ms and 16.24 ms, respectively, meeting the 24-ms control cycle requirement, and both joint torques and velocities complied with UR5 constraints. Paired high-curvature experiments further demonstrated that prediction quality determines the safety margin, while independent gating maintains the hard safety boundary even when constant-speed extrapolation suffers from severe mismatch.

Keywords: UR5 robotic arm; dynamic obstacle avoidance; model predictive control; fast expanded random tree; artificial potential field; online motion planning

1 Introduction

With the development of smart manufacturing, flexible production, and human-robot collaboration, industrial robots are transitioning from structured production environments to dynamic, uncertain human-robot shared spaces. In addition to precisely tracking trajectories, robotic arms must also adjust their motion in real time in response to environmental changes to avoid collisions. Therefore, generating safe, smooth, and real-time executable motions under dynamic obstacles has become a critical issue for autonomous robotic operation[1–2].

Dynamic obstacle avoidance for robotic arms requires the coordination of global path planning, local motion adjustments, and real-time safety control. Existing research primarily includes sampling-based planning, reactive obstacle avoidance, and optimization-based predictive control.

The Rapidly-Exploring Random Tree (RRT) and its improved variants possess strong high-dimensional search capabilities and have been widely used for robotic arm path planning in complex environments[3–8]. RRT* improves path quality through incremental optimization[9–11], but it typically relies on static environmental models and struggles to actively adapt to changing obstacle states; consequently, offline global planning cannot consistently guarantee execution safety[12–13].

Reactive methods adjust robotic arm motion based on real-time perception[14–16].Artificial Potential Fields (APF) are simple in structure, computationally efficient, and easy to implement online[17]; however, their local decision-making relies on the current state and lacks prediction of future obstacle movements. In complex dynamic scenarios, they are prone to getting stuck in local optima, leading to trajectory oscillations or a decline in continuity[18]. They also typically require large evasion maneuvers to ensure a safety margin, sacrificing execution efficiency and smoothness.

Model Predictive Control (MPC) can predict future states and explicitly handle state, control, and environmental constraints, and has been applied to dynamic obstacle avoidance for robotic arms[19–21]. While its forward-looking nature offers advantages over reactive methods, the online optimization of robotic arms with six or more degrees of freedom involves a large number of kinematic and collision constraints, resulting in high computational complexity. How to complete the solution within a finite control cycle remains a key challenge for the engineering deployment of MPC.

Other trajectory generation methods include Covariant Hamiltonian Optimization (CHOMP)[22], Stochastic Trajectory Optimization (STOMP)[23], Dynamic Motion Primitives[24], and Online Trajectory Generation[25]; control obstacle functions[26–27], their integration with MPC[28–29], and robotic arm safety filtering[30] provide explicit safety constraints.Integrating global planning, local optimization, and safety constraints can complement each other's strengths, but three types of problems remain:

(1) There is a lack of effective integration between global planning and online obstacle avoidance.

Insufficient integration between global planning in static environments and online dynamic obstacle avoidance results in real-time controllers bearing an excessive burden of obstacle avoidance.

(2) There is a clear trade-off between safety and real-time performance.

Expanding the prediction horizon or adding constraints can increase the safety margin, but this prolongs solution time and compromises real-time performance.

(3) There is uncertainty between the optimization results and actual execution.

Model and prediction errors can cause the theoretically optimal trajectory to lose safety during actual execution, thus requiring independent verification.

A single algorithm struggles to simultaneously satisfy the requirements of global search, local response, and real-time safety. Hierarchical motion generation clarifies the responsibilities of each algorithm, reduces coupling, and enhances engineering applicability.

This paper proposes a hierarchical safety-aware dynamic obstacle avoidance framework for a six-degree-of-freedom UR5 robotic arm, integrating global planning, reference optimization, online predictive control, and execution verification.RRT* generates a global reference in a static environment; APF-assisted pre-bending adjusts the trajectory in advance based on the nominal trend of dynamic obstacles, shifting part of the obstacle avoidance burden forward; a two-layer soft-constrained linear MPC coordinates tracking and safety margins using differentiated relaxation weights; and pre-submission safety gating independently verifies the optimized output through continuous gap scanning to address prediction errors and model uncertainties.

(1) A hierarchical safety-aware motion generation framework is constructed to decouple global planning, reference adjustment, online predictive control, and safety verification.

(2) Combine reference pre-bending with a two-level soft-constrained linear MPC to improve smoothness and real-time performance while maintaining a safety margin.

(3) Design an independent pre-commitment safety gate that combines continuous sweep detection with multi-level fallback to enhance execution robustness under dynamic uncertainties.

(4) Systematically evaluate safety, real-time performance, and engineering feasibility through baseline comparisons, ablation studies, sensitivity analysis, stochastic statistical analysis, and dynamic experiments.

2 System Description and Problem Definition

2.1 UR5 Kinematic Model and Workspace

2.1.1 Kinematic Modeling and Joint Parameterization

The subject of this study is the Universal Robots UR5 six-degree-of-freedom serial robotic arm, which consists of six rotational joints. This paper adopts the modified Denavit–Hartenberg (modified DH) convention for modeling; the link parameters are shown in Table 1.

Table 1 Modified DH Parameters for the UR5

Joint i	θ_i	d_i	a_{i-1}	α_{i-1}	$offset(red)$
1	q_1	89.2	0	0	0
2	q_2	0	0	−90	-π/2
3	q_3	0	425	0	0
4	q_4	−109.3	392	0	π/2
5	q_5	94.75	0	90	0
6	q_6	82.5	0	90	0

Using the coordinate system established in the table above, the forward kinematic transformation matrix of the robotic arm from the base (Link 0) to the end effector (Link 6) is shown in Equation (1)

$$T_0^6 = \prod_{i=1}^{6} T_{i-1}^{i} \tag{1}$$

The pose transformation between adjacent joints in the robotic arm coordinate system and the reference coordinate system is achieved through rotation and translation, where the transformation matrix for each link is calculated using D-H parameters, as shown in Equation (2)

$$T_{i-1}^{i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i\sin\alpha_{i-1} \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i\cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where `$\theta_i$` represents the angle of the `$i$` th joint, `$\alpha_{i-1}$` represents the rotation angle of the `$i-1$` th link, and the joint offset vector `$offset = [0, -\pi/2, 0, \pi/2, 0, 0]$` is used to align the model origin with the physical origin of the robotic arm.The homogeneous transformation between adjacent link coordinate systems is given by the standard modified DH formula, and the forward kinematics from the base to the end effector is the product of the transformations of each link. Let the joint variables be denoted as $q = [q_1, \cdots . q_6]^T \in \mathbb{R}^6$. It should be noted that the modified DH model in this paper differs from the official URDF model of the UR5 in terms of the definitions of zero position and orientation for joints 2, 3, and 4; all planning and control operations are performed under the DH convention.

2.1.2 Capsule Approximation of Link Geometry

To balance accuracy and efficiency in obstacle avoidance calculations, this paper approximates the link geometry as a capsule: an envelope formed by expanding outward from a uniform radius r_L = 45 mm, with the polyline segments connecting adjacent joint nodes serving as axes.A key advantage of using a capsule is that the distance from a spatial point to the link is given by the exact distance from the point to the line segment; this distance is continuous along the link axis with no blind spots, and thus does not depend on the sampling density of control points. Let the set of joint nodes under configuration q be denoted as $\{p_0, \cdots, p_6\}$. Then, the clearance $D(q, o)$ between a spatial point o and the robotic arm's body is defined as the minimum distance from the point to each link segment in o .

$$D(q,o) = \min_{i=0,\ldots,6} dist\left(o, \overline{P_i P_{i+1}}\right) \tag{3}$$

q Joint configuration vector (6-dimensional); o spatial point/obstacle center coordinates; P_i represents the Cartesian position of the i th joint node; $\overline{P_i P_{i+1}}$ represents the i th link segment; $dist(\cdot,\cdot)$ Euclidean distance from the point to the segment

2.1.3 Workspace and Obstacle Layout

The test scenario includes static and dynamic obstacles, representing fixed structures and randomly entering moving objects, respectively. Static spheres and cuboids are known during the planning phase and are avoided by the global planner.The two dynamic spheres, each with a radius of 100 mm, move in a straight line at a nominal speed with superimposed sinusoidal perturbations to simulate non-uniform motion. Their initial positions are calculated backward based on the desired encounter time and passing distance, ensuring that obstacle avoidance events occur inevitably while maintaining a controllable level of difficulty. The system architecture is shown in Figure 1.

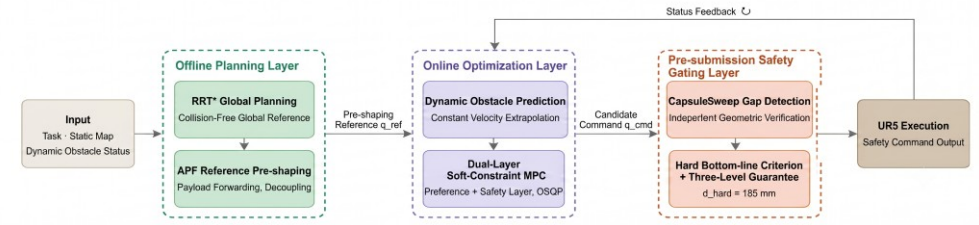


Figure 1 System Hierarchical Framework

Based on the capsule radius and obstacle radius, three levels of distance thresholds are defined (all referring to the distance from the obstacle center to the connecting rod axis), as listed in Table 2.

Table 2 Three-Level Safety Distance Thresholds

Threshold	Value (mm)	Components	Meaning
d_{hard}	185	$r_o + r_L + 40$	A Firm Safety Bottom Line That Must Not Be Crossed
d_{safe}	265	$r_o + r_L + 120$	Soft Constraint Primary Defense Layer (Rises with Speed)
d_{pref}	405	$r_o + r_L + 260$	Preference Spacing Layer

2.2 Problem Description

2.2.1 Problem Statement

Given the initial state of a robotic arm q_{start} , the target configuration q_{goal} , known static obstacles, dynamic obstacles whose positions can be measured in real time but whose future trajectories are unknown, and a control cycle of Δt (24 ms) along with joint kinematic constraints, solve the online motion generation problem.

2.2.2 Constraint Definitions

(1) Safety Constraints: The clearance between the robotic arm's body and all obstacles must be no less than the hard safety threshold, and this constraint must hold over a continuous time interval rather than only at discrete sampling times. Since the robotic arm and obstacles move simultaneously within a single control cycle, checking only at sampling points would overlook intrusions occurring between two points; therefore, this is formulated as a sweeping clearance requirement for the entire motion segment:

$$\min_{s\in[0,1]} D(q_k + s(q_{k+1} - q_k), o(t_k + s\,\Delta t)) \geq d_{\text{hard}} \tag{4}$$

s where p_i is the interpolation parameter for the subinterval; q_k, q_{k+1} represents the joint configurations at the current and next time steps; $o(t)$ represents the position of the obstacle at time t ; Δt represents the control cycle; d_{hard} represents the hard safety clearance;

(2) Kinematic constraints: $q_{\min} \leq q_k \leq q_{\max}$, $\left\| \dot{q}_k \right\| \leq \dot{q}_{\max}$; during the planning phase, $\dot{q}_{\max}$ = 1.0 rad/s, and during the execution phase, 1.5 red / s ; these are decoupled to preserve additional velocity margin during obstacle avoidance maneuvers.

(3) Dynamic feasibility: The drive torques at each joint are given by $|\tau_j| \leqslant \tau_{j,\max}$, where $\tau = M(q)\ddot{q} + C(q,\dot{q}) + g(q)$. This constraint is not directly incorporated into the online optimization to reduce computational load and to lay the groundwork for feasibility verification.

(4) Real-time Constraint: All computations within a single control cycle must be completed within the sampling period.

2.2.3 Optimization Objective

Subject to the above constraints, the control objective is to minimize the weighted sum of the tracking error, the control increment, and the safety margin within the prediction time window:

$$\min_{u} \sum_{i=1}^{N_p} (\| q_{k+i} - q_{k+i}^{ref} \|_R^2 + \| \Delta u_i \|_R^2) + w_{pref} \sum_i \varepsilon_i^{pref} + w_{safe} \sum_i \varepsilon_i^{safe} \tag{5}$$

U Let N_p be the decision vector for joint velocity increments within the prediction time window, $\|\cdot\|_Q^2$ be the squared norm weighted by matrix Q, ΔU_i be the control increment at step i , and R be the control increment weighting matrix. ε_i^{pref} and ε_i^{safe} correspond to the violation quantities of the preference margin layer and the safety margin layer, respectively, with weights w_{pref} = 300 and w_{safe} = 8000.the first two terms represent the tracking cost and the control smoothing cost, respectively, while the latter two terms represent the relaxation penalties for the two-layer soft constraints.

2.2.4 The Core Trade-off of the Problem

The four types of constraints are mutually restrictive: increasing the avoidance margin improves safety but increases joint acceleration and peak torque; extending the prediction time horizon improves foresight and tracking quality but expands the scale of the quadratic planning problem; relaxing safety constraints maintains solution feasibility but may breach hard safety limits when obstacles approach at high speeds. Therefore, the hierarchical framework aims to achieve an engineering balance among safety, real-time performance, and dynamic feasibility.

3 Hierarchical Obstacle Avoidance Framework

3.1 Overall Structure of the Framework

The framework described in this paper is divided into three layers: offline, online, and commit. Each layer is independent of the others in terms of time scale and information sources. The offline layer uses RRT* to generate collision-free paths in a static obstacle environment; after pruning and time parameterization, these paths form a reference trajectory. Potential field pre-bending is based solely on the nominal motion of dynamic obstacles, causing the reference trajectory to deviate from its channel in advance.The online layer predicts obstacles by extrapolation at a constant speed, using a two-layer soft-constraint MPC to track the pre-bent reference trajectory and dynamically avoid obstacles. The commit layer independently verifies the capsule sweep clearance between the current configuration and the candidate command; upon failure, it sequentially triggers three levels of fallback mechanisms.

This design achieves fault isolation through separation of responsibilities: the offline layer provides the global path, the online layer handles dynamic responses, and independent geometric verification safeguards the safety baseline, preventing a failure in a single layer from directly resulting in a collision.

3.2 APF Reference Pre-bend

3.2.1 Design Motivation and Nominal Constant-Speed Model

If the system relied entirely on online MPC for obstacle avoidance, the controller would need to make significant adjustments within a limited prediction horizon and control authority as an obstacle approaches, increasing the difficulty of QP solution and potentially causing abrupt movements. Given that in engineering scenarios, the nominal motion plan of an obstacle is often available in advance, this paper pre-bends the reference trajectory during the offline phase.This module uses only the nominal constant-speed model $\tilde{o}_{k(t)} = o_{\langle k,0\rangle} + v_k t$, excluding sinusoidal disturbances; otherwise, the offline layer would have prior knowledge of future information unavailable to the online layer, leading to distorted performance attribution.

3.2.2 Calculation of Potential Field Offset for Pre-bending

Pre-bending is updated sample-by-sample based on a time index. For each intermediate sample, the distances between all control points and the nominal obstacle trajectory are calculated, and the closest match is selected:

$$(c^*,k^*) = arg\min_{c,k} \| p_c(q_i^b) - \tilde{o}_k(t_k) \|_2, d_{\min} = \min_{c,k} \| \cdot \|_2 \tag{6}$$

$$\hat{\mathbf{n}} = \frac{\mathbf{p}_{c^*} - \tilde{\mathbf{o}}_{k^*}}{|\mathbf{p}_{c^*} - \tilde{\mathbf{o}}_{k^*}|_2}, \quad \Delta\mathbf{x}_i = (d_{\text{bend}} - d_{\min})\hat{\mathbf{n}} \in \mathbb{R}^3 \tag{7}$$

c^*, k^* For the index of the nearest (control point, obstacle) pair, $arg\min$ is the independent variable at which the minimum value is obtained, $p_c(q_i^b)$ is the th$c$ control point under the pre-curve profile, $\tilde{o}_k$ is the nominal constant-speed predicted position of the th$k$ obstacle, $\hat{n}$ is the unit direction vector pointing from the obstacle to the control point, Δx_i is the Cartesian offset of the th$i$ sample, d_{bend} is the target pre-curve spacing, and $\mathbb{R}^3$ is the three-dimensional real space;Position translation is applied only when the closest distance $d_{\min}$ is less than the pre-bend target d_{bend} . By using the closest pair rather than the sum of forces from multiple obstacles, we avoid translation failure caused by repulsive forces canceling each other out. Subsequently, the Cartesian displacement required to push the control point to the target gap is constructed along the unit direction $\hat{n}$ pointing from the obstacle to the control point, and is mapped to joint increments via a damped least-squares mapping:

$$\Delta\mathbf{q}_i = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}} + \lambda\mathbf{I}_3)^{-1}\Delta\mathbf{x}_i, \quad \lambda = 10^3\ \mathrm{mm}^2 \tag{8}$$

Δq_i where the joint increments for sample i ,J is the Jacobian matrix of the control point's position, λ is the damping coefficient, and I_3 is the 3×3 identity matrix

After the joint increments are normalized, they are moving-averaged along the time axis using a window width of 5 and multiplied by end-point fading weights to preserve the start and end profiles; if the updated sample collides with a static obstacle, the offset is reversed. After three iterations, the reference velocity is recalculated using central differencing.

3.2.3 Decoupling from the Online Preference Layer

The pre-bend target and the online preference distance are explicitly decoupled: `$d_{bend}$ = 320 mm <$d_{pref}$ = 405 mm`. If `$d_{bend} \geq d_{pref}$ `, the pre-bend would push the reference trajectory beyond the online preference layer threshold, causing the preference layer to remain inactive throughout the entire process and causing the two-layer structure to degrade into a single layer. Maintaining this interval ensures that the online preference layer still has usable adjust bandwidth, allowing the two layers to perform distinct functions rather than overlapping.

3.3 Model Predictive Control with Two-Layer Soft Constraints

3.3.1 Rolling Linearization and Constraint Linearization

The obstacle-avoidance constraints are non-convex in joint space. This paper adopts the Sequential Quadratic Programming (SQP) approach to perform rolling linearization within each control cycle, with the SQP iteration count set to 2. The decision variable is the joint velocity increments $U = \left[\Delta u_1^T, \cdots \Delta u_{N_p}^T\right]^T$ within the prediction time window, and the predicted pose is given by the lower-triangular integration operator:

$$\mathbf{q}_{k+i} = \mathbf{q}_k + \Delta t \sum_{j=1}^{i} \left(\dot{\mathbf{q}}_{k+j}^{\mathrm{ref}} + \Delta\mathbf{u}_j\right) \tag{9}$$

q_{k+i} where i denotes the joint position at step within the prediction horizon, $\dot{q}^{ref}$ represents the reference joint velocity, and Δu_j denotes the joint velocity increment at step j . For prediction step i , control point c , and obstacle m , the distance D from the predicted control point position (under zero control input) to the center of the obstacle and the unit normal vector $\hat{n}$ are calculated. The first-order variation of the distance along the normal direction yields the constraint row vector:

$$a_{i,c,m}^{\mathrm{T}} = (\hat{n}_{i,c,m}^{\mathrm{T}} J_c)\Delta t L_i \tag{10}$$

where $a_{i,c,m}^T$ is the constraint row vector for step i , control point c , and obstacle m ; $J_c \in \mathbb{R}^{3\times 6}$ is the Jacobian of the control point position; and L_i is the i th block row of the lower triangular block integrator.

3.3.2 Construction of Two-Level Soft Constraints

Obstacle-avoidance constraints are not imposed as hard inequalities; instead, two levels of relaxation variables are introduced, which share the same constraint row vector but differ only in their thresholds and relaxation methods:

$$D_{i,c,m} + \mathbf{a}_{i,c,m}^{\mathrm{T}}\mathbf{U} \geqslant d_{\mathrm{pref}} - \varepsilon_i^{\mathrm{pref}}, \quad \varepsilon_i^{\mathrm{pref}} \geqslant 0 \tag{11}$$

$$D_{i,c,m} + \mathbf{a}_{i,c,m}^{\mathrm{T}}\mathbf{U} \geqslant d_{\mathrm{safe}} - \varepsilon_i^{\mathrm{safe}}, \quad \varepsilon_i^{\mathrm{safe}} \geqslant 0 \tag{12}$$

The complete decision variable is $Z = [U^T, (\varepsilon^{pref})^T, (\varepsilon^{safe})^T]^T$, and the entire quadratic programming problem is written in standard form:

$$\min_{\mathbf{z}} \frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{H}\mathbf{z} + \mathbf{f}^{\mathrm{T}}\mathbf{z} \quad \text{s.t.} \quad \mathbf{A}_{\mathrm{ineq}}\mathbf{z} \leqslant \mathbf{b}_{\mathrm{ineq}}, \quad \mathrm{lb} \leqslant \mathbf{z} \leqslant \mathrm{ub} \tag{13}$$

$D_{i,c,m}$ where is the currently predicted gap value, d_{pref}, d_{safe} are the thresholds for the preference layer and safety layer, respectively, $\varepsilon_i^{pref} \geqslant 0, \varepsilon_i^{safe} \geqslant 0$ are the non-negative slack variables for the two layers, Z is the complete decision vector, H is the Hessian matrix, f is the linear term vector, $A_{ineq}.b_{ineq}$ are the inequality constraint matrices and vectors, and lb, ub are the lower and upper bounds of the decision variables.

The two relaxation layers share constraint row vectors determined solely by geometric relationships; therefore, although the number of constraints doubles, the computational complexity for geometric calculations remains unchanged.When far from obstacles, the relaxations for both layers are zero; upon entering the preference distance `$d_{pref}$`, the low-cost preference layer gently increases the clearance; as the robot further approaches the safety threshold, the high-cost safety layer prioritizes maintaining safety. Static spheres and cuboids are also always included in the constraints to prevent the robot arm from being pushed into static obstacles while avoiding dynamic obstacles.

3.3.3 Advantages of the Two-Layer Soft Constraint Over the Single-Layer Soft Constraint

The two-layer soft constraint structure allows for a differentiated trade-off between safety margin and tracking performance. A single-layer structure cannot distinguish between routine approaches and approaches to the hard safety threshold, and may reduce the clearance by relaxing the constraints when an obstacle approaches at high speed. In this paper, the low-weight preference layer handles routine approaches, while the high-weight safety layer safeguards the safety threshold.Both the standard MPC and CBF-QP baselines experienced hard-limit breaches, whereas the proposed method showed no breaches across 55 parameter sweeps; ablation results indicate that the two-layer structure increases the minimum clearance by 37–106 mm.

3.4 Pre-Submission Safety Gating and Three-Level Fallback

3.4.1 Design Motivation and Positioning

The optimizer may fail due to numerical issues, and linearization errors can also cause actual execution to deviate from predictions. To address this, this paper introduces an independent geometric check prior to command submission.This mechanism is a single-step reactive filter rather than a safety guarantee in the sense of a control invariant set: if a high-speed obstacle has already been swept into the robot arm's path and the robot arm cannot disengage in time, it cannot prevent a violation. Therefore, this paper positions it as a depth-limiting filter that restricts intrusion depth and duration.

3.4.2 Sweep Gap Verification and Lipschitz Fast Screening

The core of the verification is to ensure that safety constraints hold over the entire control cycle interval. Given the current pose q_k and a candidate pose q^{cand} , the robot arm and the obstacle are advanced synchronously within a subinterval, and the minimum margin over the entire interval is taken:

$$m(\mathbf{q}_k \to \mathbf{q}^{\mathrm{cand}}) = \min_{s \in S} \left[D(\mathbf{q}_k + s\,\Delta\mathbf{q}, \mathbf{o}(t_k + s\,\Delta t)) - d_{\mathrm{hard}} \right] \tag{14}$$

$m(\cdot \to \cdot)$ This represents the minimum safety margin for the entire path from the current pose to the candidate pose, where q^{cand} is the candidate pose, s is the set of subsampled points, and $\Delta q = q^{cand} - q_k$, where s is the set of equidistant sampling points on $[0, 1]$; three subsampled points are selected during the control phase to balance real-time performance;In the evaluation in Section 5, the sampling density was increased to 8, meaning the results were scored using criteria stricter than those of the controller. To reduce the positive kinematic overhead of a full sweep, an endpoint fast screening method based on the Lipschitz bound was introduced: if the gap between endpoints exceeds the sum of the displacement of the arm point and the center of the obstacle, the entire interval is necessarily safe, and the full sweep can be skipped:

$$D(\mathbf{q}^{\mathrm{cand}}, \mathbf{o}(t_k + \Delta t)) \geqslant d_{\mathrm{hard}} + 1.1\left(\sum_i |\Delta q_i| R_i + v_{\mathrm{obs,max}} \Delta t\right) \tag{15}$$

$R_i\, i$ represents the conservative lever arm for the joint, used to estimate the maximum displacement of the arm point; $v_{obs,\ \max}$ is the maximum speed of the obstacle; 1.1 is the safety factor

- $\sum_i |\Delta q_i| R_i$ is the upper bound of the arm point's displacement; $v_{obs,\ \max} \Delta t$ is the upper bound of the obstacle's displacement

3.4.3 Three-Level Safety-First Strategy

When a candidate command fails validation, the system sequentially executes a three-level strategy: First, it performs a braking path search along the direction from the current configuration to the candidate configuration to find the first non-negative margin configuration; if maintaining the current position is also unsafe, it applies a potential field repulsion away from the obstacle to generate an active evasion configuration; if this is still infeasible, it submits the configuration with the largest margin among all candidates and issues an alarm to prevent further intrusion.All fallback commands are subject to a single-step speed limit.

4 Numerical Experiments and Results

4.1 Simulation Setup and Evaluation Metrics

All experiments were conducted in MATLAB. The robot arm's kinematics and visualization were implemented using the Peter Corke Robotics Toolbox; quadratic programming was solved by OSQP [31] with backtracking using the quadprog interior-point method; and dynamic verification utilized the calibrated inertia parameters from the official UR5 model. Unless otherwise specified, comparisons used deployable constant-speed extrapolation prediction.

A unified evaluation metric is adopted. The gap is measured as the minimum distance during a continuous sweep along the capsule's body line segment, with the robotic arm and the obstacle advancing synchronously between adjacent trajectory points; while the control gate uses 3 subsampling points, the evaluation uses 8 to ensure a more rigorous assessment.OSQP employs a deterministic configuration and a tightened state criterion. The proposed method, APF, single-layer soft-constraint MPC, and CBF-QP all utilize the same static obstacle information; the optimization methods employ the same solver and tolerance, and the repulsive gain of APF and the parameters of CBF-QP were both tuned via scanning.The CBF-QP baseline in this paper employs a single-layer soft constraint with relaxation penalty, aiming to isolate the contributions of the two-layer structure and independent gating rather than to verify the theoretical safety of CBF. The dynamic window method was not included in the comparison due to repeated convergence failures.

4.2 Qualitative Behavior

In a typical task, the robotic arm first navigates around a static obstacle, then intersects with two dynamic obstacles, and finally reaches the target. The gap curve exhibits a three-stage response: when there is no collision, the slack of both layers is zero; as the obstacle approaches and enters the preference zone, the low-cost preference layer is activated; throughout the entire intersection process, the hard safety threshold is never breached, consistent with the design objectives of the two-layer structure.

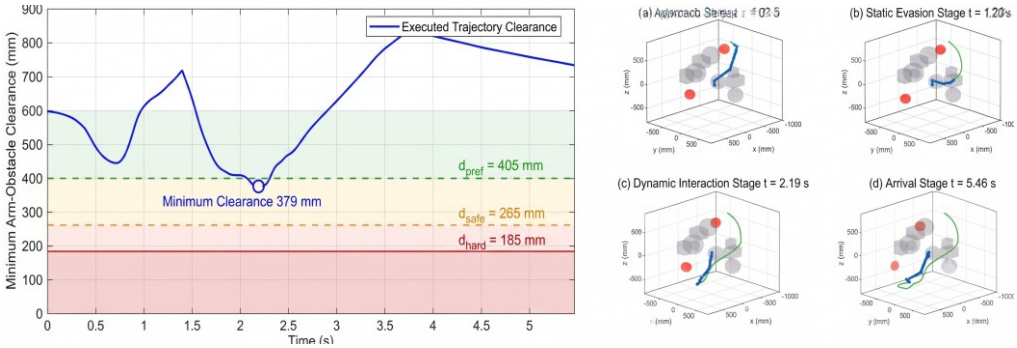


Figure 2 Trajectory snapshots at key moments and obstacle distance versus time curves for a typical task

4.3 Comparison with Baseline Methods

Under identical scenarios, solvers, and environmental information, our method was compared with three types of baselines; the quantitative results are shown in Table 3.

Table 3 Quantitative Comparison of the Four Methods

Metric	Ours	APF	MPC-std/CBF-QP	Test
Task Success Rate	94.2%	93.3%	0%/0%	p=1.00
Hard Threshold Breakthrough Rate	0%	0%	100%/100%	—
Joint jerk (rad/s³)	295	765	—	p=2.7e-7
Joint path length (rad)	5.8	6.2	—	p=0.08
Mean minimum clearance (mm)	401	425	—	p=7.8e-4
End-of-track RMSE (mm)	299	244	—	p=1.3e-7

The RMS of end-effector offset in Table 3 is measured relative to the original, obstacle-free reference generated by RRT*, serving as a metric for the detour cost of dynamic obstacle avoidance rather than the tracking error of the controller relative to its own reference. The proposed method forms a wider, gentler detour arc through offline pre-curving; while the offset is slightly greater than that of APF, the joint jerk is only 1/2.59 that of APF;APF produces a smaller offset but results in more abrupt local avoidance. Therefore, while maintaining comparable safety, the proposed method trades a moderate detour for smoother motion that is more favorable for the actuators.

Both MPC-std and CBF-QP—which employ single-layer soft constraints and lack submission gating—are unusable: they exhibit a 100% hard-bound violation rate, a 0% success rate, and frequent solution degradation.When conflicts arise between obstacle avoidance constraints and joint speed limits, both methods reduce the clearance at the cost of penalties; the lack of independent geometric verification causes the relaxation to directly translate into actual breaches. Since neither method successfully reached the target, their jerk, clearance, and tracking metrics are no longer defined. The parameter scan of CBF-QP still failed to eliminate breaches, indicating that, under the conditions encountered in this paper, single-layer soft optimization must be supplemented with independent submission gating.

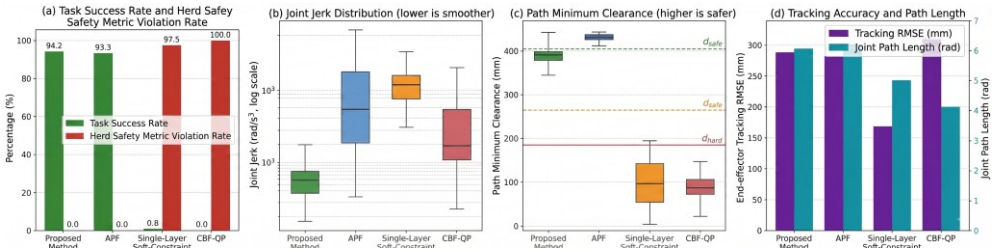


Figure 3 Comparison of Key Performance Metrics for the Four Methods

4.4 Verification of Robustness Against Prediction Errors

To examine the impact of prediction error, this paper constructs a high-curvature ambush scenario: a dynamic obstacle performs high-frequency, large-amplitude sinusoidal direction changes at the critical moment of convergence, causing significant mismatch in constant-speed extrapolation. For 100 random seeds, both constant-speed prediction and perfect prediction were applied; the obstacle trajectories were identical for both, differing only in the online prediction method. The primary metric was the minimum gap over the entire path, and the evaluation criteria were consistent with those in Table 4.

Figure 4 and Table 4 show that the median minimum clearance for Perfect Prediction was 400.9 mm, higher than the 390.6 mm for Constant-Speed Prediction; the median paired difference was 11.2 mm. The Holm-corrected Wilcoxon signed-rank test yielded a p-value of <0.001, and 64 out of 100 seeds resulted in a larger clearance, indicating that the prediction quality can improve the safety margin.However, the constant-speed prediction did not breach the hard bottom line, with the minimum gap being 213.6 mm—still higher than 185 mm; in contrast, the perfect prediction did breach the hard bottom line in one seed, with the minimum gap being 118.8 mm. Therefore, the perfect prediction did not improve performance regarding adherence to the hard bottom line.

The results indicate that predictive optimization differs from submission gating: prediction quality determines the safety margin, while independent gating safeguards the hard threshold.More accurate predictions increased the gap by 11.2 mm ($p < 0.001$), demonstrating the value of predictive optimization relative to purely reactive methods; even when constant-speed extrapolation suffered severe mismatches in high-curvature encounters, the gating mechanism still achieved zero hard-limit breaches. This separation design does not rely on perfect predictions and thus balances performance with robustness.

Table 4 Head-to-head comparison of prediction methods in high-curvature scenarios (100 seeds)

Metric	Constant-Speed Prediction (const_vel)	Perfect Prediction (perfect)	Pairwise Comparison
Hard Baseline Breakthrough Rate	0/100	1/100	McNemar p = 1.00
Median Minimum Gap (mm)	390.6	400.9	—
Mean minimum gap (mm)	381.1 ± 40.2	396.3 ± 45.6	—
Minimum Gap (mm)	213.6	118.8	—
Minimum Clearance Pairing Deviation (mm)	—	Median +11.2	Wilcoxon-Holm p < 0.001, r = 0.47

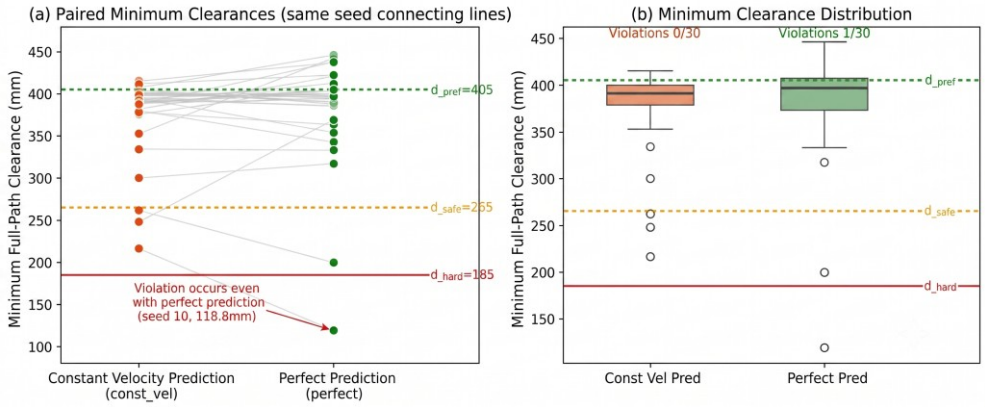


Figure 4 Comparison of paired results between constant-speed prediction and perfect prediction in high-curvature scenarios.

4.5 Ablation Experiments

Using the full framework (A_Full) as the baseline, we disabled pre-bending (B_noBend), two-layer soft constraints (C_noTwoLayer), and submission gating (D_noGuard), respectively, and set up a bare baseline (E_Baseline). All five configurations were run in both nominal and stress scenarios, evaluated using a refined sweep with 8 subsampling points; none of the configurations breached the hard safety threshold, and the minimum clearance remained above the threshold.

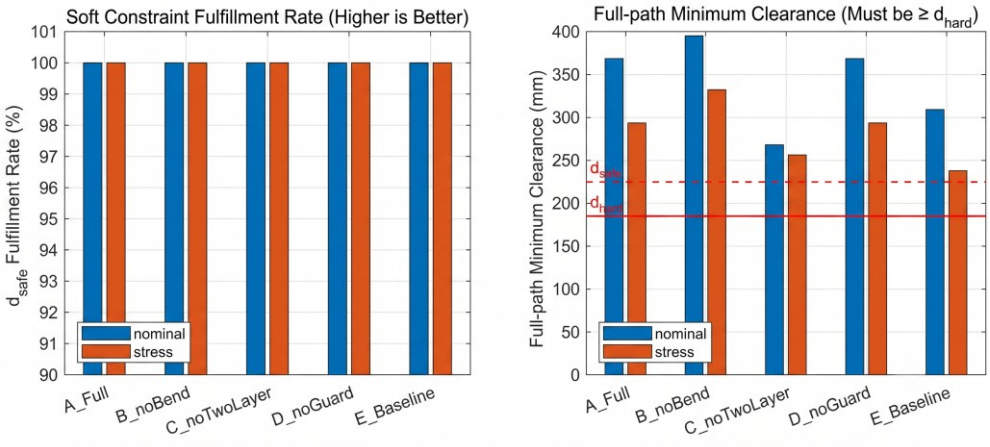


Figure 5 Safety of Ablation Experiments: Comparison of soft constraint maintenance rate (left) and minimum gap throughout the run (right) under nominal and stress scenarios

4.5.1 Contribution of the Two-Layer Soft Constraint

The two-layer soft constraint is the primary factor in improving the safety margin. After disabling this structure, the minimum clearance in both scenarios decreased significantly; the net improvements across the four scenario-prediction combinations were 100.6, 77.9, 37.2, and 106.3 mm, respectively. The preference layer releases clearance at a low cost during the conventional approach phase, thereby raising the clearance baseline for the convergence process.

4.5.2 Shifting the Obstacle Avoidance Burden from the Online Layer to the Offline Reference

Pre-bending does not directly increase the absolute safety margin: after disabling pre-bending, the minimum clearance in the nominal scenario actually increased from 368.6 mm to 395.3 mm. Its function is to shift the obstacle avoidance burden from online optimization to the offline reference.Measuring the online burden by the peak relaxation of the preference layer, pre-bending reduces it from 380 to 0 in the nominal scenario and from 324 to 178 (approximately 45%) in the stress scenario. This division of labor suppresses abrupt command changes and preserves a margin for the online layer to handle prediction mismatches.

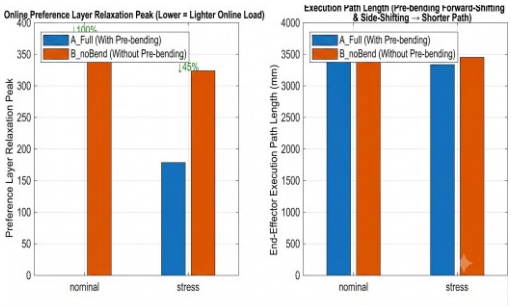


Figure 6: Pre-bending shifts the obstacle avoidance burden from the online QP to the offline reference: peak online preference layer relaxation (left) and executed path length (right)

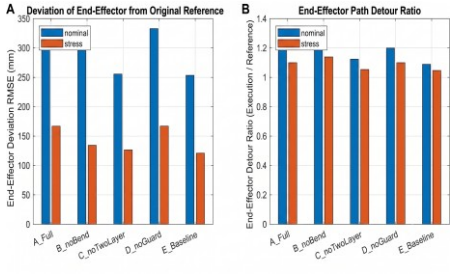


Figure 7. Obstacle-avoidance cost in ablation experiments: end-point offset relative to the original reference (left) and end-point path detour ratio (right)

4.5.3 Submission Gating Is Not Activated in Deterministic Scenarios

In deterministic scenarios, the metrics for A_Full and D_noGuard are identical, with zero instances of gating activation, indicating that the two-layer soft constraints have already stabilized the margin well above the hard threshold. To evaluate the gating mechanism, 80 additional adversarial trials were conducted.After enabling the gate, the intrusion momentum decreased from 163 to 126, the average intrusion depth decreased from 70.1 mm to 52.4 mm, the maximum intrusion decreased from 176.3 mm to 147.2 mm, and the violation duration decreased from 28 steps to 17 steps; furthermore, intrusion did not deepen in any of the trials, and the number of breakthrough trials decreased only from 19 to 18.In 14 cases, the arm encountered a physically infeasible situation due to high-speed obstacle scanning; since the current configuration was already unsafe when gating was applied, it should be classified as a depth-limiting filter rather than a violation eliminator.

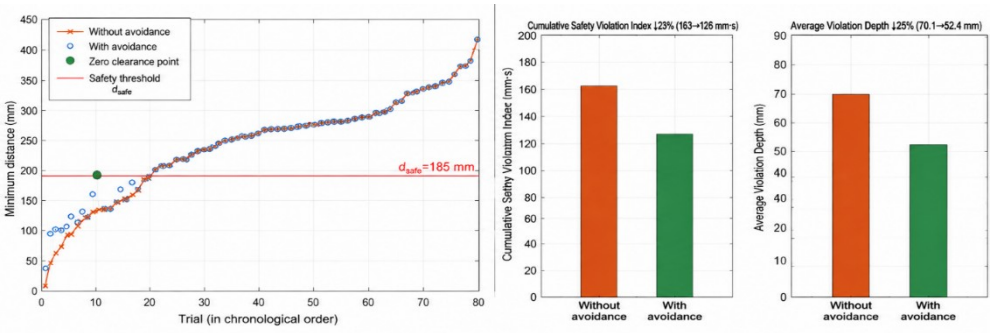


Figure 8 Comparison of depth-limiting effectiveness and minimum clearance for gated systems

4.6 Parameter Sensitivity Analysis

The effects of key parameters were analyzed using one-factor analysis (OFAT) and two-dimensional grid analysis. None of the 55 configuration sets covering the 5 parameters resulted in hard-limit breaches, secondary planning failures, or kinematic violations, and peak joint velocities remained below 1.5 rad/s, indicating structural robustness in safety within the study range.

4.6.1 Preference Distance: Clearance Adjustment Amount and Its Inflection Point

The dynamic minimum clearance increases with the preferred distance (d_{pref}), rising from 397 mm to 493 mm; however, beyond approximately 455 mm, the tracking cost increases sharply: the joint tracking RMS increases from 0.48 rad to 1.13 rad, the end-effector peak error increases from 609 mm to 1210 mm, and the end-point error increases from near zero to 0.13 rad.This paper adopts 405 mm, which falls within the robust range in front of the knee point.

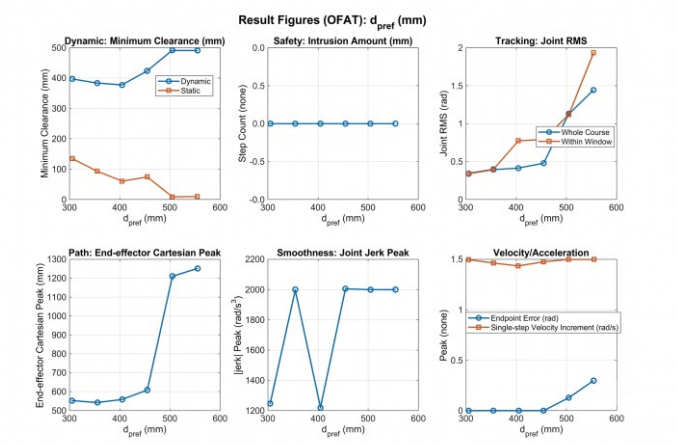


Figure 9 Single-factor sensitivity analysis for the preferred distance ( )

4.6.2 Prediction Time Window and Command Low-Pass Coefficient

The prediction time window primarily improves tracking and has a limited impact on the safety margin: when the number of prediction steps (N_p) is increased from 10 to 20, the tracking RMS decreases from 1.00 rad to 0.42 rad; it stabilizes after reaching 20, and there is no clear trend in the minimum gap.Therefore, a value of 20 is selected to balance foresight, accuracy, and real-time performance. The command low-pass coefficient α primarily adjusts smoothness: when the coefficient increases from 0.15 to 1.00, the peak jerk rises approximately linearly from 520 rad/s³ to 4773 rad/s³, while the minimum gap fluctuates only slightly; it can be tuned independently based on requirements for smoothness and response speed.

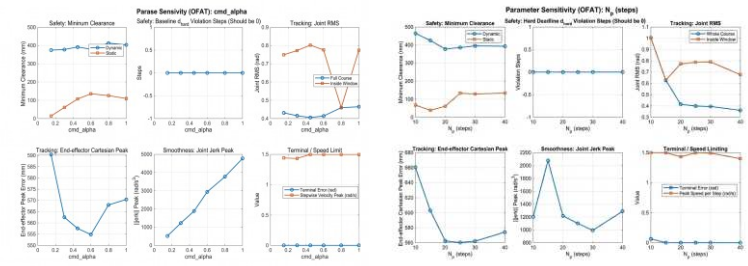


Figure 10 Single-factor sensitivity of the prediction time window and the command low-pass coefficient

4.6.3 Relaxation Weight w_{safe} and w_{pref}

Increasing the safety layer weight w_{safe} increases the minimum clearance, but the benefit saturates after approximately 5×10^4, while jerk and tracking RMS continue to deteriorate, with the latter reaching a maximum of 1.41 rad. Therefore, a value of 8,000 is selected, which lies before the inflection point for excessive conservatism. The preference layer weight is insensitive to all metrics within the scanning range and does not require fine-tuning; in this paper, w_{safe} is set to 300.

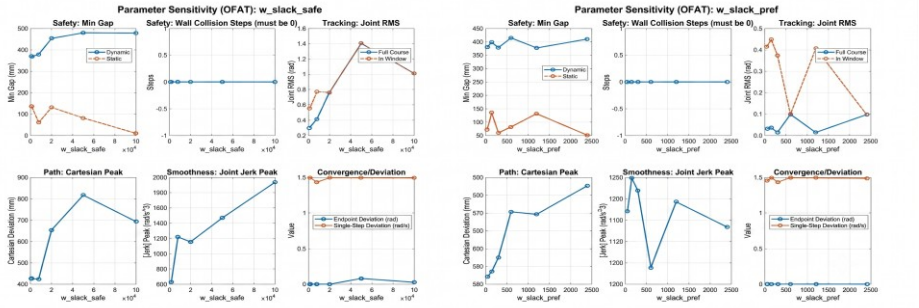


Figure 11 Single-factor sensitivity of the safety layer weight ( ) and the preference layer weight ( )

4.6.4 Two-Dimensional Coupling of Dual-Layer Weights

Figure 12 shows a 5×5 two-dimensional grid on the × plane, consistent with the OFAT results: the dynamic minimum clearance and jerk are primarily governed by the safety-layer weights, while the preference-layer weights have a limited influence; the number of steps to exceed the hard line is 0 for all combinations. Therefore, the safety-layer weights should be tuned first, and the preference-layer weights can be selected within a wider range.

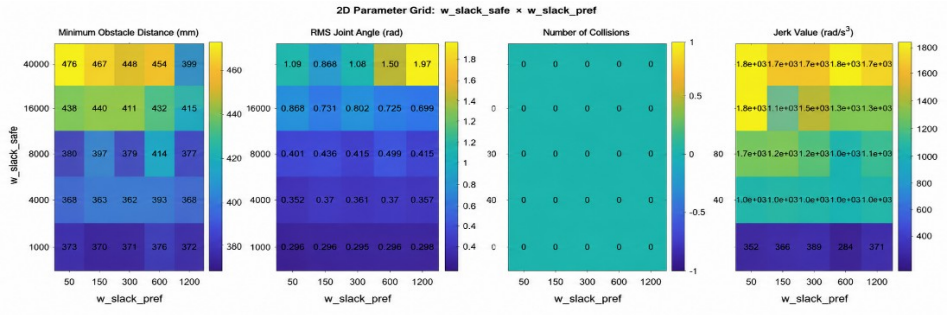


Figure 12 Two-dimensional grid for dual-layer relaxation weights×

4.7 Monte Carlo Statistical Validation

To evaluate statistical reliability, 100 independent random trials were conducted, with the intersection time, passing distance, and obstacle velocity multiplier perturbed in each trial. The results are summarized in Table 5.

Table 5 Monte Carlo Safety Statistics (100 Random Trials)

Metric	Result
Dynamic Hard-Line Breach Rate	0/100 = 0%

Metric	Result
Static Hard-Endpoint Failure Rate	0/100 = 0%
Mission Success Rate (Arrival Without Breach)	100/100 = 100%
Secondary planning failure/kinematic violation	0/0
Peak joint velocity	All ≤ 1.5 rad/s
Dynamic minimum clearance (median/mean/minimum)	404.4/398.2 ± 43.9/279.5 mm
5th percentile of dynamic minimum clearance	303.1 mm
Static minimum clearance (median/minimum)	75.9/10.0 mm

Dynamic and static clearances are defined differently. Dynamic clearance is the distance from the center of the obstacle sphere to the axis of the link capsule; the threshold already includes a 100 mm obstacle radius and a 45 mm link radius, with a hard lower limit of 185 mm;Static clearance is the shortest distance from the capsule surface to the static obstacle surface; a breach occurs when this distance is less than zero. Therefore, a minimum static clearance of 10.0 mm indicates that there is still 10.0 mm remaining to the obstacle surface, which corresponds to a static breach rate of 0%, and cannot be directly compared to the 185 mm dynamic threshold.

4.7.1 Safety and Clearance Distribution

No breakthroughs were observed in 100 random encounters, but this does not guarantee zero breakthroughs; the corresponding confidence upper bound is 3.7%.The minimum dynamic clearance was 279.5 mm, which is 94.5 mm higher than the hard threshold, and the 5th percentile also exceeds the safety primary defense layer. Neither the left tail of the dynamic nor the static clearance distributions approached their respective thresholds, and the static penetration count was 0. The median delay in arrival time relative to the reference without obstacle avoidance was -0.26 s, indicating that obstacle avoidance generally did not delay the mission.

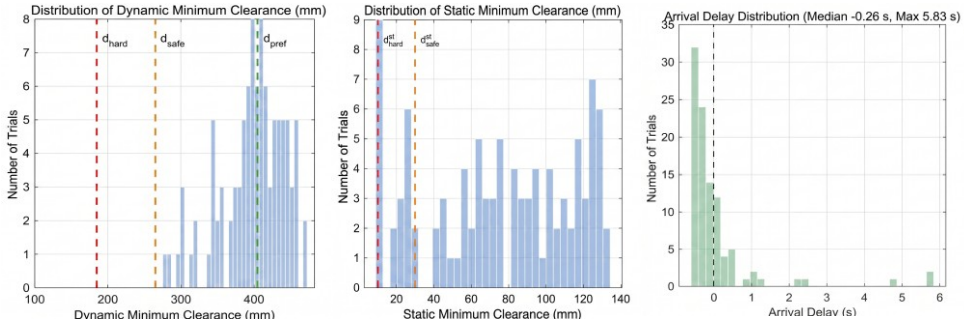


Figure 13 Distribution of dynamic minimum gaps, distribution of static minimum gaps, and distribution of arrival time delays from 100 random simulations

4.8 Real-Time Performance

To evaluate real-time performance, the deployable control time is defined as t_{ctrl} the sum of reference window construction, quadratic programming, potential field fallback, and submission verification, excluding post-processing statistics and plotting. The simulation was repeated nine times with the same configuration; the median across repetitions was calculated for each control step, and the initial few steps of each run were excluded. The results are shown in Table 6.

Table 6 Real-Time Performance Statistics

Metric	Value	Metric
Median Solution Time	7.7 ms	0.32
95th percentile (P95)	16.24 ms	0.68
99th percentile (P99)	19.34 ms	Compliance Rate: 99.56%

Metric	Value	Metric
Maximum single step	121.69 ms	Number of rollbacks ≈ 1
OSQP Solution Ratio	99.56%	Average number of constraint rows 45.6 (peak 140)

Under a 24 ms control cycle, the median control computation time is 7.7 ms, of which QP accounts for approximately 4.26 ms; the 95th percentile is 16.24 ms, with corresponding real-time factors of 0.32 and 0.68, respectively. Based on a 20 ms threshold, 99.56% of control steps meet the requirement and are solved directly by OSQP, with approximately 1 backtrack.

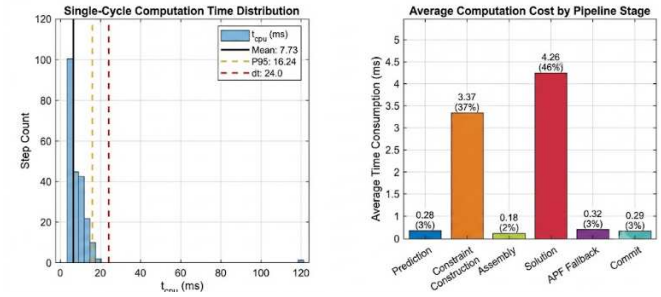


Figure 14 Histogram of single-cycle computation time distribution and average computation overhead for each stage

4.8.1 Worst-Case Single Step and Safety Degradation

The maximum single-step computation time was 121.69 ms, occurring when two dynamic obstacles approached simultaneously and the number of constraints reached its peak; based on a 20-ms threshold, only 0.44% of control steps (approximately 1 out of 227 steps) timed out.Three-level solution backtracking and independent commit gating prevent the issuance of unsafe instructions, and since instructions committed in the previous cycle have passed sweep checks, a single-step timeout results in performance degradation rather than a safety failure. This extreme value may also be influenced by MATLAB's just-in-time compilation and is expected to be reduced in a hardware implementation.

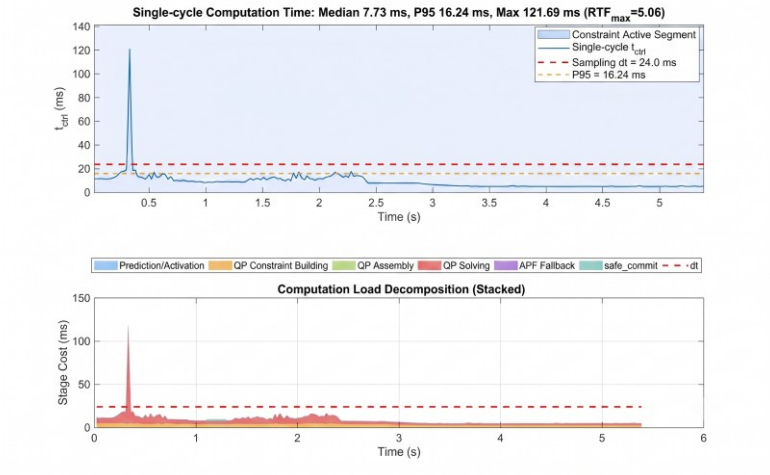


Figure 15 Time evolution of single-cycle computation time (top) and segmented stacking of computational load (bottom)

4.9 Verification of Dynamical Feasibility

Using the final joint trajectories as input, the data was interpolated using cubic splines to a 1-ms resolution. Velocity and acceleration were then obtained via numerical differentiation, and the inverse dynamics were computed using the official UR5 calibration model ($\tau = M(q)\ddot{q} + C(q,\dot{q}) + g(q)$). The rated joint torques and speed limits were set to the UR5 nominal values; the results are shown in Table 7.

Table 7 Joint Torque and Velocity Feasibility

Joint	Peak Torque $N \cdot m$	Rated $N \cdot m$	Utilization	Peak Speed (rad/s)
J_1(Base)	62.7	150	42%	1.41
J_2 (Shoulder)	116.8	150	78%	1.44
J_3(Elbow)	47.6	150	32%	1.16
J_4(Wrist 1)	9.5	28	34%	1.23
J_5(Wrist 2)	11.5	28	41%	1.15
J_6(Wrist 3)	0.5	28	2%	0.00

The peak torques at all six joints are below their rated values. The shoulder joint (J_2) has a peak torque of 116.8 and a utilization rate of 78%, leaving a 22% margin; as it is located upstream in the kinematic chain and bears the gravitational and inertial torques from the subsequent links, it acts as a bottleneck, which is consistent with physical expectations.The utilization rates of the remaining joints were all below 42%. The peak joint velocities did not exceed 1.44 rad/s, which is well below the±3.14 rad/s limit, and there was no saturation or clipping. The peak torques were concentrated during the phase where the two obstacles intersected and then rapidly declined, indicating that the trajectory satisfied the dynamic and velocity constraints.

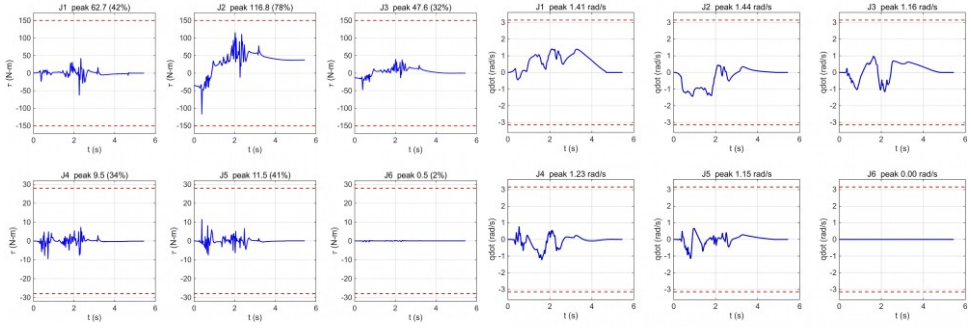


Figure 16 Time-dependent drive torques of the six joints and velocity profiles of the six joints (red dashed lines indicate ± velocity limits)

5. Conclusions

This paper proposes a hierarchical online dynamic obstacle avoidance framework based on RRT*–APF–MPC for a six-degree-of-freedom UR5 robotic arm, which separates responsibilities through offline reference pre-bending, online two-layer soft-constraint MPC, and pre-submission safety gating.In simulations, this method reduced joint jerk to 1/2.59 of its original value while maintaining safety and task completion rates comparable to those of a tuned APF; none of the 100 random encounters resulted in hard-limit violations, and all reached the target. The median control computation time was 7.7 ms, with 99.6% of control steps taking less than 20 ms; joint torques and velocities remained within rated limits. No hard-limit violations occurred across all 55 parameter configurations.

Ablation and adversarial tests clarified the contributions of the three design elements.The two-layer soft constraint increased the minimum clearance for the four scenario-prediction combinations by 37.2–106.3 mm, serving as the primary factor in increasing the safety margin. The reference pre-bend shifted the load forward to the offline phase, reducing the peak relaxation of the preference layer from 380 to 0 in nominal scenarios and by approximately 45% in stress scenarios, thereby suppressing command abrupt changes and preserving online adjustment margins.Submission gating does not trigger in conventional scenarios, but in adversarial encounters, it reduces intrusion momentum, average intrusion depth, and violation duration by 23%, 25%, and 39%, respectively, without deepening the intrusion. Thus, the offline layer decouples the load, the online layer balances safety, and the submission layer limits intrusion.

Paired high-curvature tests further distinguished the roles of prediction and gating. When only the online prediction method was altered, perfect prediction increased the median minimum gap throughout the entire journey by 11.2 mm ($p < 0.001$ after Holm correction), indicating that prediction quality can improve safety margins; even when constant-speed extrapolation resulted in severe mismatches at intersections, independent gating still achieved zero hard-line breaches.Therefore, prediction is responsible for increasing the margin, while gating is responsible for safeguarding the bottom line, enabling the system to balance performance and robustness even when predictions are imperfect.

This paper still has limitations: the submission gate is a depth-limited filter rather than a violation eliminator, and thus cannot guarantee escape from physically infeasible encounters such as high-speed obstacles sweeping the arm; a small number of computational peaks still occur during dual-obstacle intersections, although their proportion is only 0.44%, and they are mitigated by a three-level fallback and independent gating as a safety net.

ll data supporting the findings of this study are included within the article. No additional raw experimental datasets were generated for public sharing

Data availability: All data supporting the findings of this study are included within the article. No additional raw experimental datasets were generated for public sharing

References

[1] Liu J, Yap HJ, Khairuddin ASM (2024) Review on motion planning of robotic manipulators in dy namic environments. J Sens 2024:5969512. https://doi.org/10.1155/2024/5969512

[2] Mohanan MG, Salgoankar A (2018) A survey of robotic motion planning in dynamic environmen ts. Robot Auton Syst 100:171–185. https://doi.org/10.1016/j.robot.2017.10.011

[3] Kuffner JJ, LaValle SM (2000) RRT-connect: an efficient approach to single-query path planning. In: Proceedings of the 2000 IEEE International Conference on Robotics and Automation (ICRA), vol. 2, pp. 995–1001. https://doi.org/10.1109/ROBOT.2000.844730

[4] LaValle SM, Kuffner JJ (2001) Randomized kinodynamic planning. Int J Robot Res 20(5):378–40 0. https://doi.org/10.1177/02783640122067453

[5] Elbanhawi M, Simic M (2014) Sampling-based robot motion planning: a review. IEEE Access 2: 56–77. https://doi.org/10.1109/ACCESS.2014.2302442

[6] Gasparetto A, Boscariol P, Lanzutti A, Vidoni R (2015) Path planning and trajectory planning alg orithms: a general overview. In: Motion and Operation Planning of Robotic Systems, Mechanisms and Machine Science, vol. 29. Springer, pp. 3–27. https://doi.org/10.1007/978-3-319-14705-5_1

[7] Kingston Z, Moll M, Kavraki LE (2018) Sampling-based methods for motion planning with const raints. Annu Rev Control Robot Auton Syst 1:159–185. https://doi.org/10.1146/annurev-control-0 60117-105226

[8] Jiang L, Liu S, Cui Y, Jiang H (2022) Path planning for robotic manipulators in complex multi-ob stacle environments based on an improved RRT. IEEE/ASME Transactions on Mechatronics 27 (6):4774–4785. https://doi.org/10.1109/TMECH.2022.3165845

[9] Karaman S, Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. Int J Rob ot Res 30(7):846–894. https://doi.org/10.1177/0278364911406761

[10] Gammell JD, Srinivasa SS, Barfoot TD (2014) Informed RRT*: optimal sampling-based path pla nning focused via direct sampling of an admissible ellipsoidal heuristic. In: 2014 IEEE/RSJ Intern ational Conference on Intelligent Robots and Systems (IROS), pp. 2997–3004. https://doi.org/10.1 109/IROS.2014.6942976

[11] Gammell JD, Barfoot TD, Srinivasa SS (2020) Batch Informed Trees (BIT*): informed asymptoti cally optimal anytime search. Int J Robot Res 39(5):543–567. https://doi.org/10.1177/0278364919 890396

[12] Otte M, Frazzoli E (2016) RRTX: asymptotically optimal single-query sampling-based motion pl anning with quick replanning. Int J Robot Res 35(7):797–822. https://doi.org/10.1177/027836491 5594679

[13] Grothe F, Hartmann VN, Orthey A, Toussaint M (2022) ST-RRT*: asymptotically-optimal bidire ctional motion planning through space-time. In: 2022 IEEE International Conference on Robotics and Automation (ICRA), pp. 3314–3320. https://doi.org/10.1109/ICRA46639.2022.9811814

[14] Fiorini P, Shiller Z (1998) Motion planning in dynamic environments using velocity obstacles. In t J Robot Res 17(7):760–772. https://doi.org/10.1177/027836499801700706

[15] Fox D, Burgard W, Thrun S (1997) The dynamic window approach to collision avoidance. IEEE Robot Autom Mag 4(1):23–33. https://doi.org/10.1109/100.580977

[16] van den Berg J, Lin M, Manocha D (2008) Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE International Conference on Robotics and Automation, pp. 1928–1935. https://doi.org/10.1109/ROBOT.2008.4543489

[17] Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. Int J Robot R es 5(1):90–98. https://doi.org/10.1177/027836498600500106

[18]Koren Y, Borenstein J (1991) Potential field methods and their inherent limitations for mobile rob ot navigation. In: Proceedings of the 1991 IEEE International Conference on Robotics and Autom ation, pp. 1398–1404. https://doi.org/10.1109/ROBOT.1991.131810

[19] Krämer M, Rösmann C, Hoffmann F, Bertram T (2020) Model predictive control of a collaborati ve manipulator considering dynamic obstacles. Optim Control Appl Methods 41(4):1211–1232. ht tps://doi.org/10.1002/oca.2599

[20] Rösmann C, Hoffmann F, Bertram T (2017) Integrated online trajectory planning and optimizatio n in distinctive topologies. Robot Auton Syst 88:142–153. https://doi.org/10.1016/j.robot.2016.11. 007

[21] Merckaert K, Convens B, Nicotra MM, Vanderborght B (2024) Real-time constraint-based planni ng and control of robotic manipulators for safe human-robot collaboration. Robot Comput-Integr Manuf 87:102711. https://doi.org/10.1016/j.rcim.2023.102711

[22] Zucker M, Ratliff N, Dragan AD, Pivtoraiko M, Klingensmith M, Dellin CM, Bagnell JA, Srinivasa SS (2013) CHOMP: covariant Hamiltonian optimization for motion planning. Int J Robot Res 32(9–10):1164–1193. https://doi.org/10.1177/0278364913488805

[23] Kalakrishnan M, Chitta S, Theodorou E, Pastor P, Schaal S (2011) STOMP: stochastic trajectory optimization for motion planning. In: 2011 IEEE International Conference on Robotics and Automation, pp. 4569–4574. https://doi.org/10.1109/ICRA.2011.5980280

[24] Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P, Schaal S (2013) Dynamical movement primitives: learning attractor models for motor behaviors. Neural Comput 25(2):328–373. https://doi.org/10.1162/NECO_a_00393

[25] Kröger T, Wahl FM (2010) Online trajectory generation: basic concepts for instantaneous reactions to unforeseen events. IEEE Transactions on Robotics 26(1):94–111. https://doi.org/10.1109/TRO.2009.2035744

[26] Ames AD, Xu X, Grizzle JW, Tabuada P (2017) Control barrier function-based quadratic programs for safety-critical systems. IEEE Transactions on Automatic Control 62(8):3861–3876. https://doi.org/10.1109/TAC.2016.2638961

[27] Ames AD, Coogan S, Egerstedt M, Notomista G, Sreenath K, Tabuada P (2019) Control barrier functions: theory and applications. In: 2019 18th European Control Conference (ECC), pp. 3420–3431. https://doi.org/10.23919/ECC.2019.8796030

[28] Zeng J, Zhang B, Sreenath K (2021) Safety-critical model predictive control with discrete-time control barrier function. In: 2021 American Control Conference (ACC), pp. 3882–3889. https://doi.org/10.23919/ACC50511.2021.9483029

[29] Jian Z, Yan Z, Lei X, Lu Z, Lan B, Wang X, Liang B (2023) Dynamic control barrier function-based model predictive control for safety-critical obstacle avoidance in mobile robots. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 3679–3685. https://doi.org/10.1109/ICRA48891.2023.10160857

[30] Singletary A, Guffey W, Molnár TG, Sinnet R, Ames AD (2022) Safety-critical manipulation for collision-free food preparation. IEEE Robot Autom Lett 7(4):10954–10961. https://doi.org/10.1109/LRA.2022.3192634

[31] Stellato B, Banjac G, Goulart P, Bemporad A, Boyd S (2020) OSQP: an operator splitting solver for quadratic programs. Math Program Comput 12(4):637–672. https://doi.org/10.1007/s12532-020-00179-2