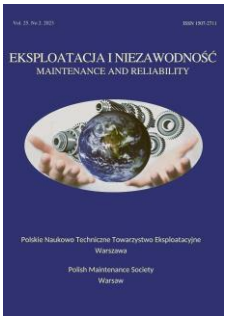

Eksploatacja i Niezawodnosc – Maintenance and Reliability

Volume 28 (2026), Issue 4

journal homepage: http://www.ein.org.pl

Article citation info:
Xiang Q, Cross synergy model-based accuracy improvement for manipulator trajectory planning in dynamic environments, Eksploatacja i Niezawodnosc – Maintenance and Reliability 2026: 28(4) http://doi.org/10.17531/ein/220209

Cross synergy model-based accuracy improvement for manipulator trajectory planning in dynamic environments

Indexed by: Web of Science Group

Qingyi Xiang[a,*]

[a] University of Wisconsin-Madison, United States

Highlights

- Introduces conditional diffusion generation, risk-aware correction, and a closed-loop mechanism of generation-optimization-online correction.
- Exhibits satisfactory robustness and deployment potential even in complex and highly dynamic scenarios.
- Proposed method effectively improves trajectory accuracy and safety margin.

Abstract

To address the common problems of degraded trajectory accuracy, insufficient safety margin, and limited real-time response capability in manipulator trajectory planning under dynamic environments, this study focuses on the collaborative optimization requirements of high precision, high safety, and high real-time performance in complex scenarios. This study proposes a Cross Synergy Model (CSM) empowered manipulator trajectory planning method and constructs a unified trajectory generation and optimization framework for dynamic environments to improve the stability and executability of trajectory planning in variable scenarios. Through the cross-fusion of multi-source environmental information and task constraints, this method introduces conditional diffusion generation, risk-aware correction, and a closed-loop mechanism of generation-optimization-online correction. It further combines dynamic constraint optimization and model predictive control to realize the collaborative linkage of a priori trajectory generation, local fine optimization, and real-time compensation during execution. The experimental results demonstrate that the proposed method effectively improves trajectory accuracy and safety margin while maintaining low planning latency, and exhibits satisfactory robustness and deployment potential even in complex and highly dynamic scenarios. Therefore, this study has certain reference significance for the field of manipulator trajectory planning in dynamic environments.

Keywords
cross synergy model, dynamic environment, manipulator, trajectory planning, generative model

1. Introduction

The manipulator is now one of the most important execution components in industrial automation systems due to the quick growth of intelligent manufacturing, flexible assembly, human-machine cooperative robots, and unmanned production lines [1,2]. In the multi-task, multi-station and highly flexible production scene, the manipulator needs to complete the traditional point-to-point movement. It also needs to deal with complex process path tracking (such as welding, spraying and polishing), high-speed dynamic obstacle avoidance, man-machine mixed cooperation and other tasks [3,4]. However, the core challenge of manipulator trajectory planning is how to maintain high accuracy, high robustness and real-time response planning ability in dynamic environment [5]. Traditional trajectory planning methods, such as sampling-based and optimization-based, can achieve higher path feasibility and better performance in static scenes [6,7]. However, when there

(*) Corresponding author.
E-mail addresses: Q. Xiang (ORCID: 0009-0009-0569-1887) xiangqingyi2027@163.com

are moving obstacles, unpredictable human behavior, sensor noise, model uncertainty and execution error accumulation in the environment, the traditional methods often have the following shortcomings:

(1) The response speed to environmental changes is insufficient, and it is difficult to meet the online re-planning requirements [8].
(2) The ability to deal with complex dynamic constraints and attitude constraints is limited, which leads to the decline of trajectory accuracy.
(3) Relying on accurate models or hand-designed features, it lacks adaptability to environmental uncertainty and high-dimensional input [9,10].
(4) It is difficult to maintain stable trajectory quality in dynamic, dense or semantically complex scenes.

In recent years, Generative Artificial Intelligence (GAI) and large-scale pre-training models have shown remarkable potential in robot path generation, motion control and decision planning. However, most of the existing models only learn single-mode or single-task information, and lack the unified fusion ability for multi-source information of manipulator (geometry, semantics, dynamics, task constraints, etc.) [11]. At the same time, the generation model often only pays attention to the "generation" of trajectory, but ignores the "enforceability", control stability and safety of trajectory in real dynamic environment [12,13]. Furthermore, although existing studies have made certain progress in generative trajectory modeling, obvious research gaps still remain. Firstly, most generative methods are mainly modeled for a single perceptual modality or local state features, and have not yet formed a unified fusion mechanism for environmental geometric information, dynamic semantic information, manipulator dynamic constraints, and task objective constraints. Thus, it is difficult to fully characterize the high-dimensional complexity of trajectory planning in dynamic environments. Secondly, existing methods often focus more on the generation quality of trajectory samples, while paying insufficient attention to whether the trajectories meet dynamic feasibility, control stability, and error compensation in the execution phase, resulting in a disconnect between the generated results and actual control execution. Thirdly, under the condition of continuous movement of dynamic obstacles and cumulative prediction uncertainty, many methods lack a constraint modeling mechanism for risk propagation and dynamic expansion of safety boundaries, leading to insufficient safety margins and degraded robustness in trajectory planning under complex scenarios.

Based on this, this study puts forward the concept of Cross Synergy Model (CSM), aiming at constructing a large-scale trajectory generation framework with the ability of generating, constraint understanding and environmental adaptability through the fusion of cross-modal, cross-task and cross-scene prior knowledge. The CSM can automatically generate high-quality trajectory priors after inputting environmental state, dynamic obstacle prediction, mission objectives and dynamic constraints. It provides high-precision initial solutions or reference trajectories for subsequent optimization controllers, and ultimately improves the trajectory planning accuracy and execution performance of the manipulator in a dynamic environment. Based on the above research gaps, this study further conducts research work in the following three aspects.

(1) Propose an overall CSM framework for manipulator trajectory planning in dynamic environments. This framework maps environmental perception information, task objective information, dynamic constraint information, and obstacle prediction information into a unified representation space. The cross-fusion of multi-source information improves the model's ability to understand complex dynamic scenes, thereby compensating for the deficiencies of existing generative methods in multi-source semantic fusion and task constraint representation.
(2) Construct an integrated trajectory planning mechanism of "conditional diffusion generation-risk-aware correction-dynamic constraint optimization-online error compensation". This mechanism focuses on the generation quality of trajectories and emphasizes the feasibility, control stability, and execution consistency of the generated results, forming a closed-loop linkage between the trajectory generation process and the back-end optimization control process, thus alleviating the problem that generated trajectories are difficult to execute directly in existing methods.
(3) Design safety constraints and a multi-source trajectory fusion strategy for the uncertainty of dynamic

environments. By explicitly introducing obstacle state prediction and its uncertainty into the trajectory planning process, combined with risk-aware adjustment and the multi-source fusion output mode of generation-optimization-online correction, the safety margin, robustness, and real-time response capability of the manipulator in complex dynamic scenes are improved.

Based on the above research contents, this study attempts to form systematic improvements on the three key issues of insufficient multi-source fusion capability, insufficient executability of generated results, and inadequate dynamic safety constraints, and provides a unified solution that balances accuracy, safety, and real-time performance for manipulator trajectory planning in dynamic environments.

2. Literature review

Trajectory planning of manipulator in dynamic environment involves multi-source information fusion, motion generation, environmental modeling and uncertainty processing [14]. From the perspective of existing research contexts, relevant literature can be further classified into three categories according to research themes, as detailed below:

(1) Traditional sampling-based and optimization-based trajectory planning methods and their limitations in dynamic scenarios.

Traditional trajectory planning research is mostly based on sampling algorithm or optimization method, which generates feasible paths through geometric information or gradient solution, which performs well in static environment, but its ability is limited in dynamic environment and high-dimensional constraints [15,16]. The effectiveness of the enhanced Rapid-Exploring Random Tree (RRT) method in manipulator multi-joint route planning was investigated by Xu and Lin. By introducing heuristic sampling strategy, the convergence speed was improved, and faster path search could be achieved in the medium-density obstacle environment [17]. However, this method depended on the geometric characteristics of the environment, was insensitive to the changes of dynamic obstacles, and was prone to path jitter and unstable convergence when dealing with complex constraints of joint space. It shows that traditional sampling-based methods have advantages in path feasibility and search flexibility, and are suitable for scenarios with clear structures and relatively fixed constraints. However, their main limitation is that the response to environmental changes relies heavily on replanning frequency, and they lack forward-looking modeling of the future states of dynamic obstacles. Therefore, their real-time performance and trajectory stability in dynamic scenarios are often insufficient. In contrast, although optimization-based methods can explicitly introduce smoothness, collision avoidance, and dynamic constraints through cost functions, they are susceptible to initial value quality and local optima under non-convex constraints and high-dimensional state spaces. Thus, the main limitations of traditional methods in dynamic environments are not only degraded accuracy, but also insufficient capability in handling time-varying scenes, adaptive replanning, and high-dimensional semantic constraints.

(2) Dynamic environment modeling and prediction methods and their hypothetical boundaries

Researchers started focusing on how to implement real-time environment sensing, obstacle prediction, and online obstacle avoidance in dynamic environments to enhance the safety and adaptability of trajectory planning as human-machine cooperation and mobile manipulator scenarios increased [18,19]. Bektemessov proposed a dynamic obstacle modeling method based on Kalman motion prediction, which improved the prediction accuracy of obstacle trajectory by fusing lidar and visual data, and achieved good performance in obstacle avoidance of mobile robots [20]. However, the prediction model assumed that the motion of obstacles obeyed a linear Gaussian process, and the prediction deviation was large in the face of complex human motion behavior, which affected the safety of subsequent trajectory planning. The core value of this line of research lies in advancing environmental “perception” to environmental “prediction”, enabling trajectory planning to no longer rely solely on static observations at the current moment, but to conduct forward-looking decision-making using future state estimation. However, from a comparative perspective, existing dynamic environment modeling methods are generally based on several idealized assumptions, such as stable obstacle motion patterns, specific noise distributions, and controllable sensor observation errors. Such assumptions can improve prediction efficiency in regular scenarios, but when facing irregular, multi-agent, and highly interactive human-robot

hybrid environments, prediction errors often accumulate rapidly, thereby weakening the safety and robustness of the trajectory planning module. Therefore, the key issue of dynamic environment modeling methods is not only prediction accuracy itself, but also whether its hypothetical boundaries are applicable to complex industrial environments and whether prediction uncertainty can be further integrated into trajectory planning constraints.

(3) Advantages and limitations of generative/diffusion-based trajectory strategies in robotic trajectory planning

Researchers are starting to investigate ways to employ large models to acquire more universal trajectory priors to enhance the planning quality in complicated settings, as generative models have demonstrated significant potential in trajectory generation, motion prediction, and robot control in recent years [21,22]. Based on Diffusion Model (DM), Chu et al. developed a framework for robot trajectory generation that produced a variety of candidate paths by learning trajectory distribution and attained good initial trajectory quality under challenges [23]. However, the model had insufficient semantic fusion ability for the input environment, and the executability and dynamic feasibility of the generated trajectory still needed back-end optimization compensation, and the overall real-time performance was still not ideal. Compared with traditional sampling and optimization methods, the prominent advantage of generative strategies, especially diffusion-based strategies, is that they can learn the trajectory distribution laws in complex scenarios from large-scale datasets, thereby generating a candidate trajectory set with greater diversity, prior rationality, and generalization ability. From a comparative perspective, such methods have obvious advantages in prior representation ability, complex scenario adaptability, and potential compatibility with multimodal inputs. However, their shortcomings are equally significant: On the one hand, many generative methods still focus mainly on optimizing the "generation quality" of trajectories, without deeply embedding dynamic feasibility, control stability, and execution safety into the generation process. On the other hand, DMs usually incur high iterative sampling costs, which are easily limited by inference efficiency in real-time trajectory planning tasks. Therefore, the research focus of generative methods in robot trajectory planning is gradually shifting from "whether trajectories can be generated" to "whether executable, interpretable, and real-time deployable trajectories can be generated".

By synthesizing the three categories of studies above, it shows that traditional sampling/optimization methods lack adaptability in dynamic scenarios, dynamic environment modeling methods are limited by idealized prediction assumptions. Although generative/diffusion-based methods show advantages in trajectory prior learning, they still have obvious gaps in multi-source semantic fusion, dynamic feasibility, control stability, and dynamic safety constraints. Thus, existing studies have not yet formed a unified framework that can simultaneously integrate environmental states, dynamic prediction, task semantics, and dynamic constraints, while balancing generative capability, optimization capability, and online correction capability.

Therefore, in this study, environmental state, dynamic prediction, task semantics and dynamic constraints are uniformly coded to construct a comprehensive trajectory generation prior. At the same time, through time series prediction, diffusion generation and risk awareness coding, the trajectory is robust to environmental changes. In addition, this study also takes the generated trajectory as the initial optimization solution, which significantly reduces the difficulty of solving and improves the final trajectory accuracy. Furthermore, this study focuses on trajectory generation itself, and takes the generated trajectory as an initial optimization solution, and integrates it with dynamic constraint optimization, risk-aware correction and online error compensation in a closed-loop coupling manner to improve the accuracy, safety and real-time executability of manipulator trajectory planning in dynamic environments.

3. Research method

3.1. Overall framework design of CSM

To improve the trajectory planning accuracy of manipulator in dynamic environment, this study constructs an overall framework of CSM for trajectory generation and optimization. This framework focuses on the unified representation of multi-source information, and further emphasizes the cross-coupling relationships among multimodal features, spatial-temporal dependencies, and tasks and constraints. A collaboratively

linked closed-loop planning mechanism is formed through conditional diffusion generation, trajectory evaluation and screening, local optimization control, and online error feedback. Specifically, the term “cross” in this study mainly includes three levels: first, modal cross, referring to the cross-modal alignment and fusion among environmental geometric information, semantic information, manipulator state information, and task constraint information. Second, spatial-temporal cross, referring to the joint modeling of the correlation between the temporal variation of obstacles and the spatial state of the manipulator in dynamic environments. Third, task-constraint cross, integrating target pose, process path requirements, safety distance, dynamic boundaries, etc., into a unified conditional representation space. “Synergy” is reflected in the information backflow and feedback update among the generation module, evaluation module, optimization module, and online correction module, making trajectory generation no longer a one-way inference process but a closed-loop system driven by prior generation, feasibility screening, control compensation, and execution feedback. Its specific architecture is shown in Table 1.

Table 1. Module composition and function description of the overall framework of CSM.

Module name	Main input	Main output	Core function	Key technologies and key points of implementation
Environment perception and scene representation module	camera/lidar point cloud, occupancy grid, obstacle detection results	Scene semantic feature vector, dynamic obstacle state	Multi-source sensor data is transformed into compact spatio-temporal and semantic representation, which provides environmental priors for trajectory generation.	Sensor fusion, target detection and tracking, time sequence coding, occupation grid/semantic grid construction.
Task and constraint coding module	Task instructions, process constraints, safety constraints	Task-constraint joint coding vector	Task requirements and motion/safety constraints are embedded into vector space, which is convenient to drive trajectory generation together with environmental information.	Text/structured task description coding, multi-dimensional constraint vectorization, normalization and weight modeling, conditional coding network
Multi-source information cross fusion and representation learning module	Scene semantic feature vector, task-constraint coding vector, and the current state of the manipulator.	Global context representation vector	The cross fusion of "environment-task-robot state" is realized in the unified representation space, and the global trajectory generation conditions are obtained.	Multi-modal fusion network, attention mechanism, residual connection and feature alignment, time-sequence-space joint coding
Cross synergy trajectory generation module	Global context representation, historical trajectory fragments, random noise or prior distribution.	Candidate trajectory set or trajectory distribution parameters	Based on the CSM, diverse candidate trajectories that meet the task requirements and are consistent with the environment/constraints are generated, and trajectory priors are formed.	Generating network, conditional generating mechanism, diversity regularization, trajectory parameterization.
Trajectory evaluation and screening module	Candidate trajectory set, environment model, constraint model	High-quality trajectory and evaluation index after screening	The optimal or suboptimal trajectory in terms of accuracy, safety and enforceability is selected from a plurality of candidate trajectories.	Collision detection, cost function design, multi-index weighted score, Pareto ranking.
Trajectory optimization and control execution module	High-quality trajectory prior, dynamic model of manipulator, real-time feedback in control cycle	Finally execute trajectory, joint control instruction and online correction result	Based on generating prior, local optimization and control are carried out to realize fine adjustment and online correction of trajectory and complete actual execution.	Constrained optimization, feedback control, online re-planning mechanism, error compensation model.

In the CSM, environmental perception features, task and constraint features, and manipulator state features are not simply superimposed, but interact hierarchically in a unified representation space. Environmental geometric information and semantic information are first encoded intra-modally, and then cross-modally aligned with task objectives, process requirements, safety boundaries, and the current state of the manipulator, gradually forming a global context representation that can simultaneously characterize scene changes, task requirements, and execution constraints. Thus, “cross” is

mainly reflected in three aspects: First, the modal cross between environmental geometric information and semantic information. Second, the spatio-temporal cross between the temporal variation of dynamic obstacles and the spatial state of the manipulator. Third, the task-constraint cross among task objectives, process requirements, and safety constraints.

The trajectory generation stage does perform one-shot inference based on a single scene input, and continuously injects environmental states, historical trajectories, task objectives, and constraint information into the generation process, enabling candidate trajectories to reflect both external environmental changes and internal execution requirements. The generated trajectories are not directly output, but enter the subsequent evaluation and screening stage, where they are comprehensively ranked according to collision risk, trajectory smoothness, target deviation, and other factors. The trajectories with higher feasibility and stability are selected as the input for back-end optimization and control. “Synergy” is mainly reflected in the closed-loop linkage among various modules. Updates in environmental perception and task representation directly affect the results of candidate trajectory generation. Feasibility information obtained in the trajectory evaluation stage further influences the subsequent optimization and control process. Error feedback generated during execution contributes to trajectory correction and local replanning. In this way, perception, fusion, generation, evaluation, and control do not form an isolated serial structure, but continuously conduct information backflow and collaborative updating around dynamic environmental changes and execution states.

Overall, the CSM integrates trajectory prior learning, dynamic environment adaptation, and execution-stage control compensation organically through the unified framework of "multi-source cross-fusion-conditional generation-evaluation and screening-optimization and control-feedback correction". This enables the manipulator to generate trajectories in dynamic scenarios, and produce trajectories that better meet task requirements, have greater safety margins, and can be executed stably.

3.2. Design of trajectory planning method for manipulator in dynamic environment

This study puts forward an overall planning method of "prediction-generation-optimization-correction", and its core consists of four levels:

(1) Dynamic environmental state modeling.
(2) Trajectory generation of CSM.
(3) Optimal control coupling solution.
(4) Online correction and uncertainty compensation.

The environment in which the manipulator is located usually contains many obstacles moving with time, so it is necessary to estimate the environmental state in the future by using the prediction model before the actual planning [24,25]. In this study, the linear system is used to describe the time sequence change of obstacles:

$$\mathrm{o}_{t+1} = \mathrm{A}\mathrm{o}_t + \mathrm{B}\mathrm{u}_t + \mathrm{w}_t \tag{1}$$

In Equation (1), o_t is the current state of the obstacle, u_t is the possible external action influence, and w_t is the Gaussian process noise, which reflects the unpredictable influence. A and B are state transition and control matrices. To obtain the future environmental trend at the moment of trajectory generation, iterative prediction is needed:

$$\hat{\mathrm{o}}_{t+\tau} = \mathrm{A}^{\tau}\mathrm{o}_t + \sum_{k=0}^{\tau-1} \mathrm{A}^{k}\mathrm{B}\mathrm{u}_{t+k} \tag{2}$$

In Equation (2), $\hat{\mathrm{o}}_{t+\tau}$ is the obstacle state prediction, τ is the joint moment, t is the moment, and then the propagation of uncertainty is considered:

$$\Sigma_{t+\tau} = \mathrm{A}\Sigma_{t+\tau-1}\mathrm{A}^{\mathrm{T}} + \Sigma_w \tag{3}$$

In Equation (3), $\Sigma_{t+\tau}$ is the covariance matrix of uncertainty and Σ_w is the covariance matrix of process noise. It shows that the manipulator can obtain the predicted future environmental state and its uncertainty before planning, which will directly enter the conditional space of trajectory generation model [26,27]. Linear state prediction models offer the advantages of simple implementation, low computational cost, and convenient online update in regular motion scenarios. However, their prediction errors may accumulate rapidly when facing complex human behaviors, sudden direction changes, or multi-agent highly interactive scenarios. To address this limitation, this study adopts it as an environmental prior with uncertainty bounds as input to subsequent modules. The subsequent trajectory generation and optimization process uses both the predicted mean and covariance simultaneously. It expands the influence range of high-risk regions through a risk cost function, and further mitigates prediction deviations via safety margin expansion terms, adaptive weight adjustment, and the online

correction module, thereby reducing the impact of linear prediction errors on the safety of the final trajectory. After the dynamic environment prediction is completed, the next step is to generate the most feasible trajectory under the known environmental constraints (See Equation 3):

$$p(\mathrm{X}|\mathrm{C}) = p(\mathrm{x}_{1:T} \mid \mathrm{o}_{1:T}, \mathrm{q}_0, \mathrm{g}, \mathrm{c}) \quad (4)$$

In Equation (4), $p(\mathrm{X}|\mathrm{C})$ is the conditional distribution of trajectory generation, $\mathrm{o}_{1:T}$ is the environmental state, g is the task target, c is the constraint information, C is the unified conditional space, and $\mathrm{x}_{1:T}$ is the complete trajectory. The backward calculation process of DM is as follows:

$$\mathrm{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathrm{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathrm{x}_t, \mathrm{C})\right) + \sigma_t \mathrm{z} \quad (5)$$

In Equation (5), α_t is the noise retention coefficient in the diffusion process. x_t is the trajectory vector, σ_t is the noise retention coefficient, $\bar{\alpha}_t$ is the cumulative product of the noise retention coefficient, z is an independent and identically distributed Gaussian noise vector, and θ is the network parameter. However, there is a risk area in the dynamic environment, so it is necessary to make risk avoidance correction when generating the trajectory [28]. In this study, the risk gradient term is added:

$$\mathrm{x}'_t = \mathrm{x}_t - \lambda_r \nabla_{\mathrm{x}_t} \mathcal{R}(\mathrm{x}_t, \hat{\mathrm{o}}_t) \quad (6)$$

In Equation (6), $\mathcal{R}$ is the risk cost function and λ_r is the risk adjustment weight coefficient. The trajectory generated has good geometric feasibility, but it still needs to ensure that it meets the dynamic constraints, safety distance constraints and trajectory smoothness. In addition, the risk cost function is not a single distance term, but consists of three components: obstacle distance risk, prediction uncertainty risk, and semantic risk regions. Obstacle distance risk describes the proximity between trajectory points and dynamic obstacles. Prediction uncertainty risk reflects the demand for safety boundary expansion caused by the temporal propagation of prediction errors. Semantic risk regions represent high-cost spaces such as no-crossing zones, personnel-sensitive areas, and equipment hazard zones. Under the combined effect of the three components, the risk function increases significantly near obstacles, in prediction-unstable regions, and in semantically sensitive areas, so that the risk gradient can continuously guide the trajectory away from high-risk positions during the diffusion sampling process. In terms of implementation, the gradient is not modified once after diffusion sampling, but embedded in each or several reverse denoising steps and applied iteratively as an additional adjustment term after denoising updates. This enables the trajectory to obtain risk guidance early in the generation stage, rather than being passively corrected in the back-end optimization phase, which helps reduce subsequent optimization pressure and improve the initial feasibility of candidate trajectories. For scenarios with large prediction errors, this study appropriately increases the risk adjustment weight to make the diffusion sampling stage more sensitive to high-risk regions. Therefore, this study uses the generated trajectory as the initial value of optimization to construct a multi-objective optimization problem:

$$\min_{\mathrm{X}} J = \lambda_e \parallel \mathrm{X} - \mathrm{X}^{gen} \parallel^2 + \lambda_s \sum_{t=1}^{T-1} \parallel \ddot{\mathrm{x}}_t \parallel^2 + \lambda_c \sum_{t=1}^{T} \mathcal{C}(\mathrm{x}_t) \quad (7)$$

In Equation (7), J is the objective function of trajectory optimization. X^{gen} is the prior trajectory generated by CSM. λ_e is the weight coefficient of consistency term. $\ddot{\mathrm{x}}_t$ is the motion change rate. λ_s is the weight coefficient of smoothness term. λ_c is the weight coefficient of collision risk. In terms of weight selection, this study follows the principles of accuracy priority, risk enhancement, and adaptive balancing. In sparse dynamic scenarios with few environmental conflicts, the weights focus more on consistency with the generative prior and smoothness. In medium-dynamic scenarios, the weight of the collision cost term is moderately increased to strengthen obstacle avoidance. In dense dynamic scenarios, the weights of safety-related terms are further raised, while the relative influence of the prior consistency term is appropriately reduced to prevent the trajectory from being overly constrained by the generative prior and losing flexibility in obstacle avoidance. In other words, the weights are not fixed but can be adaptively adjusted according to scene density, obstacle velocity, and prediction uncertainty. In the experiments of this study, the weights are determined using a hierarchical setting method based on scene density levels. When scene density increases or prediction covariance rises, the weights of risk and collision terms are synchronously enhanced, while the smoothing term is maintained within a relatively stable range to ensure that the trajectory does not oscillate violently due to excessive avoidance. And the optimization also needs to satisfy the dynamic equation of manipulator:

$$\mathrm{M}(\mathrm{q})\ddot{\mathrm{q}} + \mathrm{h}(\mathrm{q}, \dot{\mathrm{q}}) = \tau \quad (8)$$

In Equation (8), $\mathrm{M}(\mathrm{q})$ is the inertia matrix, h is the nonlinear dynamic term, q is the joint angle vector of the manipulator, $\dot{\mathrm{q}}$ is the joint angular velocity vector, and $\ddot{\mathrm{q}}$ is the joint angular acceleration vector. At the same time, to cope with the uncertainty of obstacle prediction, a safety distance constraint is added:

$$d(\mathrm{x}_t, \hat{\mathrm{o}}_t) \geq d_{min} + \kappa\sqrt{\Sigma_t} \quad (9)$$

In Equation (9), d is the distance function, κ is the confidence expansion coefficient, The confidence expansion coefficient is generally set to 2.0, corresponding to the safety boundary expansion range under an approximately 95% confidence level. This setting ensures the trajectory safety margin while maintaining satisfactory planning feasibility and execution efficiency. Meanwhile, for the systematic deviation that may be caused by the linear prediction model, this study first expands the feasible safety domain through covariance expansion and confidence boundary enlargement, so that potential risks are amplified and avoided in advance at the optimization stage. When the prediction error further increases, the scene-adaptive weighting mechanism synchronously enhances the influence of the risk cost term and the collision term, thereby driving the trajectory planning toward more conservative and safer solutions. The final optimized trajectory is used as the reference trajectory of Model Predictive Control (MPC) controller:

$$\mathrm{u}_{1:K}^{*} = \arg\min_{\mathrm{u}_{1:K}} \sum_{k=1}^{K} \left(\| \mathrm{x}_{t+k} - \mathrm{x}_{t+k}^{ref} \|_Q^2 + \| \mathrm{u}_{t+k} \|_R^2 \right) \quad (10)$$

In Equation (10), $\mathrm{u}_{1:K}^{*}$ is the optimal control input sequence. $\mathrm{u}_{1:K}$ is the control input sequence. x_{t+k}^{ref} is the state of the reference trajectory at time $t+k$. Q and R are symmetric positive definite matrices. In a dynamic environment, even if the planning is good, the actual execution may still deviate from the ideal trajectory due to noise, external interference or sudden obstacles [29]. At the closed-loop interface, the trajectory state sequence output by the back-end optimization module is directly used as the reference trajectory state input for MPC. The velocity, acceleration, and safety boundary information obtained in the optimization stage are further converted into control input constraints and state constraints to limit the feasible action space of the controller within the prediction horizon. For sudden changes of dynamic obstacles, MPC preferentially updates the control input based on the current reference trajectory and the latest state estimation, while the online correction module further undertakes rapid error compensation to reduce the superimposed impact of prediction errors and execution disturbances. Therefore, real-time error compensation is needed. Firstly, an online correction module based on feedback error is designed:

$$\mathrm{x}_t^{corr} = \mathrm{x}_t + K_p(\mathrm{g} - \mathrm{x}_t) + K_d(\dot{\mathrm{g}} - \dot{\mathrm{x}}_t) \quad (11)$$

In Equation (11), x_t^{corr} is the trajectory state after online error compensation, g is the state of the target end effector, $\dot{\mathrm{g}}$ is the expected speed of the task target, K_p is the proportional gain matrix, and K_d is the differential gain matrix. To integrate the advantages of "generation-optimization-correction", this study constructs a weighted fusion model of the final execution trajectory:

$$\mathrm{x}_t^{final} = \beta_1 \mathrm{x}_t^{gen} + \beta_2 \mathrm{x}_t^{opt} + \beta_3 \mathrm{x}_t^{corr} \mathrm{s.t.} \beta_1 + \beta_2 + \beta_3 = 1 \quad (12)$$

In Equation (12), x_t^{final} is the final execution trajectory state. x_t^{opt} is the trajectory point obtained after multi-objective optimization. β_1, β_2 and β_3 are all weights. In terms of weight allocation, this study defaults to the optimized trajectory as the main execution benchmark to ensure overall feasibility and stability. When the scene changes rapidly or local disturbances are significant, the weight of the online correction component can be appropriately increased to enhance the system's responsiveness to sudden risks. In relatively stable environments, a high weight of the generative prior component is retained to exploit the prior advantage of the generative model in global trajectory shape.

4. Experimental Design

To verify the effectiveness of the proposed CSM in improving the accuracy of manipulator trajectory planning in dynamic environments, this study adopts the public Physics-Informed Trajectory Network (PITNet) manipulator trajectory dataset as the source of original trajectories, and constructs an enhanced experimental set for dynamic environment evaluation based on it. The original data mainly contain 7-DoF manipulator joint-space trajectory sequences, which are generated by superimposing sinusoidal perturbations and Gaussian noise on a cubic polynomial baseline path to simulate trajectory fluctuations caused by load variations, environmental disturbances, and control errors in actual industrial scenarios. The dataset is available at DOI: 10.17632/25h8bjz4w5.2.

In the experiments, a total of 7200 trajectory samples is used, among which the training set, validation set, and test set contain 5040, 1080, and 1080 samples respectively. The original trajectory lengths mainly range from 246 to 254 sampling points, which are uniformly resampled to 250-time steps in this study to ensure consistent input lengths for different samples during generation, optimization, and control. The sampling frequency is set to 50 Hz, corresponding to an average duration of approximately 5 s per trajectory. Since the original PITNet dataset mainly provides manipulator trajectory information without directly including dynamic obstacle annotations, this study further constructs dynamic environment enhanced annotations based on the original trajectories. For each trajectory sample, the initial position, velocity, size, future predicted trajectory, and semantic risk region labels of dynamic obstacles are additionally recorded. The motion patterns of dynamic obstacles are generated in a hybrid manner, including uniform linear motion, piecewise steering motion, and random walk with boundary constraints. The first two types are used to simulate regularly moving objects such as industrial equipment and transport vehicles, while the latter is used to simulate pedestrians or irregularly moving objects. To avoid overly idealized dynamics, smooth constraints are imposed on velocity and direction changes in the random walk mode, enabling obstacle trajectories to exhibit continuity and local abrupt behavior characteristics.

To improve the interpretability of the experimental results, this study divides the dynamic scenarios into three categories: Sparse Dynamic Scenario, Moderate Dynamic Scenario, and Dense Dynamic Scenario, which are quantitatively defined by the number of obstacles, trajectory overlap rate, and trajectory crossing frequency. All three scenarios are constructed by stratified sampling from the enhanced test set, with each scenario containing 360 test trajectories. The quantitative definitions and complexity statistics are presented in Table 2.

Table 2. Quantitative definitions and complexity statistics of three dynamic scenes.

Scenario	Number of Dynamic Obstacles	Trajectory Overlap Rate	Average Obstacle Density (obj/m²)	Trajectory Crossing Frequency
Sparse Dynamic Scenario	1–2	< 15%	0.42	0.67
Moderate Dynamic Scenario	3–4	15%–35%	0.86	1.43
Dense Dynamic Scenario	5–6	> 35%	1.27	2.38

Among them, the trajectory overlap rate is used to measure the overlapping proportion between the manipulator reference path and the future motion envelope of dynamic obstacles within the time window, and the trajectory crossing frequency is used to count the average number of potential crossings between the manipulator path and obstacle motion trajectories per unit trajectory. It shows that the three scenarios are clearly distinguishable in terms of obstacle density and crossing complexity, which provides a quantitative basis for the subsequent analysis of model performance differences under different dynamic intensities.

The experimental platform of this study consists of a simulation environment and a deep learning training environment. The manipulator model used in the experiments is Franka Emika Panda (7-DoF), whose joint-space dynamic parameters are consistent with the official calibration parameters, used to verify the accuracy of trajectory planning in joint space and Cartesian space. The operating system of the main experimental platform is Ubuntu 20.04 LTS, the robot middleware is ROS Noetic, the motion planning framework is MoveIt1, and the 3D physical simulation software is Gazebo11, which is used to construct dynamic obstacle scenarios and simulate the interaction between the manipulator and the environment. The main hardware configuration for deep learning training and inference is as follows: GPU: NVIDIA GeForce RTX 4090 with 24 GB video memory; CPU: Intel Core i9-13900K; System memory: 64 GB DDR5. In addition to the main platform, an embedded platform NVIDIA Jetson AGX Orin (64 GB) is introduced in the hardware sensitivity analysis experiments to evaluate the real-time performance changes of the model under edge deployment conditions. The proposed CSM adopts a conditional generation structure based on the combination of Transformer and diffusion process. The encoder layer count is set to 8, with 8 self-attention heads per layer, hidden dimension 256, feed-forward network dimension 1024, GELU activation function, and dropout rate of 0.1 in the

attention module to balance model expressiveness and overfitting prevention. The time step of the diffusion generation process is set to 1000, the noise schedule adopts a linear beta scheme, the initial noise parameter is set to 1.0×10^{-4}, and the final noise parameter is set to 0.02.

The contrast models chosen in the experiment are PITNet and Diffusion Policy (DP). To ensure the rigor of dynamic environment evaluation, all models are trained and tested under the same trajectory sample split, the same dynamic obstacle augmentation rules, and the same three types of scenario definitions.

For comparison metrics, the average trajectory error is used to measure the mean deviation between the trajectory generated by the model and the reference trajectory over the entire time series. A smaller value indicates higher overall trajectory fitting accuracy. The terminal position error reflects the deviation between the final arrival point of the trajectory and the target point, and is mainly used to evaluate whether the manipulator can accurately reach the task target. The trajectory smoothness error measures the stability of the trajectory during execution; in this study, it is statistically calculated based on the second-order variation between adjacent time steps of the trajectory. A smaller value means a smoother trajectory with less jitter, which better meets practical execution requirements. The minimum safety distance error describes the minimum distance between the manipulator trajectory and dynamic obstacles, reflecting the safety margin of the trajectory in a dynamic environment. A larger value generally indicates stronger obstacle avoidance capability and lower risk.

In terms of real-time performance and resource consumption indicators, planning time represents the total time consumed by the model from receiving scene input to outputting complete trajectory results, and is used to evaluate the overall planning efficiency. Average single-step inference latency represents the average time required for the model to update the state or correct the trajectory at each step during online execution, and is used to reflect the response speed in the real-time control stage. GPU memory usage represents the level of video memory resources occupied by the model during the inference stage, and CPU utilization represents the occupation degree of processor computing resources during model operation. These two indicators are jointly used to evaluate the deployment cost and hardware adaptability of the model.

Both GPU memory usage and CPU utilization are measured after the model enters a stable inference state, rather than at the moment of program startup. The resource occupancy is recorded after the model completes loading, preheating, and continuous operation for a period of time, so as to avoid instantaneous fluctuations in the initialization stage from affecting the representativeness of the results. The reported GPU memory usage mainly corresponds to the peak memory level during the stable inference phase, including the main memory overhead caused by model parameters, runtime cache, and intermediate feature maps. CPU utilization is taken as the average occupancy level during stable operation, which is used to reflect the processor load under continuous inference conditions. Such a setting is closer to the actual deployment state, and can particularly reflect the running cost of the model on edge devices or embedded platforms more realistically.

5. Experimental comparison and analysis

5.1. Hardware sensitivity analysis experiment

To further evaluate the deployment potential of the proposed model on different computing platforms, this study adds hardware sensitivity analysis experiments. Under the same test set and the same parameter settings, PITNet, DP, and the proposed optimized model are run respectively on a desktop GPU platform (NVIDIA GeForce RTX 4090) and an embedded GPU platform (NVIDIA Jetson AGX Orin). The planning time and average single-step inference latency are statistically analyzed under the Dense Dynamic Scenario. The purpose of this experiment is to analyze the sensitivity of different models to changes in hardware computing power and further identify the main computational bottlenecks of their real-time performance. The experimental results are shown in Table 3.

The results in Table 3 show that on the desktop GPU platform, the proposed optimized model outperforms PITNet and DP in both planning time and average single-step inference latency. When the computing platform is switched to Jetson AGX Orin, the real-time performance of all three methods degrades, but the optimized model in this study still maintains the shortest planning latency and the lowest single-step inference delay. This indicates that although the CSM adopts a multi-stage structure including generation, optimization and

correction, the collaboration between high-quality trajectory priors at the front end and local optimization at the back end reduces the number of invalid iterations, thus retaining relatively better real-time stability under reduced computing power. Further analysis shows that the performance degradation of PITNet on the embedded platform is more significant, mainly because its physical consistency constraints and back-end correction process rely heavily on local iterations. The computational bottleneck of DP mainly comes from the multi-step sampling process of the diffusion strategy, whose inference overhead in highly dynamic scenarios is more easily restricted by embedded computing power. In contrast, although the optimized model also adopts the diffusion generation mechanism, the introduction of risk-aware correction and multi-source fusion output after candidate trajectory generation enables the back-end control stage to perform local correction based on a higher-quality initial trajectory, resulting in a more compact overall computational pipeline. It can therefore be concluded that the main computational bottleneck of the model lies in the diffusion sampling and multimodal fusion stages, while the back-end MPC and online correction modules have a relatively small impact on the total latency.

Table 3. Hardware sensitivity analysis under dense dynamic scenario.

Method	PT on RTX 4090 [ms]	AIL on RTX 4090 [ms]	PT on Jetson AGX Orin [ms]	AIL on Jetson AGX Orin [ms]
PITNet	102.91	11.37	216.48	24.16
DP	94.87	10.24	189.73	20.83
The proposed algorithm	71.92	7.39	148.62	15.47

5.2. Comparison and analysis of trajectory planning accuracy

This study performs performance comparison experiments from several aspects, including trajectory accuracy, trajectory smoothness, and dynamic environment security, in order to methodically assess the trajectory planning effect of the manipulator under the CSM. The results are shown in Figure 1.

The results in Figure 1 show that the average trajectory errors of the optimized model in Sparse, Moderate and Dense are 1.27, 1.56 and 1.98 respectively, which are obviously lower than the baseline model. For example, in Dense Dynamic Scenario with dense obstacles, the MTE of PITNet has risen to 2.87, while the error of DP in the same scene is 2.53. In contrast, the average trajectory error of the proposed optimized model can still be controlled within 2.00, which shows that the CSM has stronger trajectory fitting ability and environmental adaptability in complex environment. In addition, the EPE of the proposed optimized model is 0.63, 0.92 and 1.09 in three scenarios, respectively. Compared with the terminal error of 1.12 in sparse scene and 1.31 in moderate dynamic scene, the proposed method can reach the target pose more accurately, especially in the case of many obstacles and strong path constraints, it can still maintain a low terminal deviation. At the same time, the TSE of the proposed optimized model on Sparse, Moderate and Dense subsets is 0.49, 0.63 and 0.81 respectively. Compared with the smoothness error of DP in Dense Dynamic Scenario of about 1.13, this method significantly reduces the fluctuation level of joint acceleration, and it is also better than PITNet's 0.96 in moderate dynamic scenes, which shows that the transcendental trajectory generated by CSM can effectively suppress high-frequency control jitter after combining with back-end optimization. Furthermore, the optimized model's minimum safety distance MSDE is larger than the baseline approach in each of the three scenarios—0.56, 0.51, and 0.47, respectively. For example, in Dense Dynamic Scenario, the minimum safety distance of PITNet is only 0.28, and the DP is only 0.33, but the safety margin of this method can still be maintained at around 0.47. This shows that in the complex environment where dynamic obstacles are frequently interspersed, this model can actively open the safety interval between the manipulator and obstacles through the uncertainty modeling and risk perception generation mechanism, and reduce the potential collision risk.

To further verify that the proposed optimized model is superior to the comparative model in statistical reliability in trajectory planning accuracy, significant experiments are also carried out, and the results are shown in Table 4.

Table 4's findings demonstrate that the optimized model's performance benefits over PITNet and DP in four assessment indices have survived rigorous statistical tests. It further proves

the effectiveness and reliability of the CSM in the accuracy and safety of manipulator trajectory planning in dynamic environment.

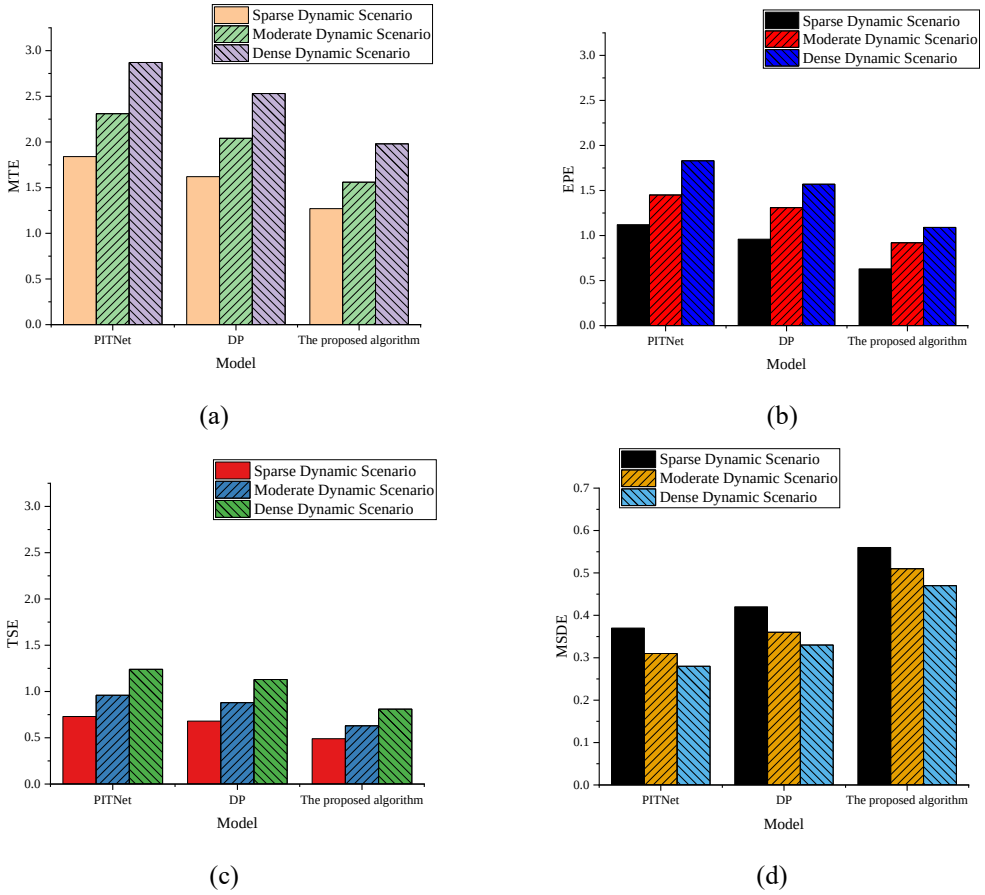

Figure 1. Comparison of trajectory planning accuracy: (a) MTE (Mean Trajectory Error) (b) EPE (End-Point Error) (c) TSE (Trajectory Smoothness Error) (d) MSDE (Minimum Safety Distance Error).

Table 4. Significance test.

Metric	Comparison (Methods)	p-value	Cohen's d	Effect Size
MTE	Proposed vs PITNet	0.008	1.34	Large
	Proposed vs DP	0.021	0.97	Large
EPE	Proposed vs PITNet	0.004	1.52	Large
	Proposed vs DP	0.017	1.08	Large
TSE	Proposed vs PITNet	0.011	1.16	Large
	Proposed vs DP	0.033	0.84	Large
MSDE	Proposed vs PITNet	0.006	1.41	Large
	Proposed vs DP	0.019	1.03	Large

5.3. Experiment of real-time and resource consumption

To further verify the usability of the CSM in the actual deployment scenario, this section compares different trajectory planning methods from two dimensions of real-time performance and resource consumption, and the results are shown in Figure 2.

From Figure 2, the PT of the proposed optimized model is 41.56ms, 55.73ms and 71.92ms in three dynamic scenarios, respectively, which is obviously lower than that of similar methods. For example, in the sparse scene, the PT of DP is about 58.34ms, but the proposed method can keep lower latency, which shows that it has higher computational efficiency in the trajectory generation stage. In terms of online inference latency, the AIL of the optimized model in Sparse, Moderate and Dense scenarios is 4.26ms, 5.17ms and 7.39ms respectively, while PITNet's AIL in Dense Dynamic Scenario is about 11.37 ms. In contrast, the proposed optimized model can still maintain a low latency in a highly dynamic environment, which is helpful to improve the response speed of online adjustment. GMU shows

that the video memory requirements of the proposed optimized model are 2.63GB, 3.14GB and 3.78GB in turn, which is more economical than 5.38GB of PITNet in Moderate scenario, which is especially important for embedded GPU or multi-model parallel deployment. The results of CPUU further verify the advantages of this method in resource efficiency. The occupancy rates in Sparse, Moderate and Dense scenarios are 47.33%, 53.18% and 61.42% respectively, while the CPU occupancy of DP in Dense Dynamic Scenario is close to 78%. It shows that through the deep coupling structure of trajectory generation, optimization and online correction, the proposed method can significantly reduce the dependence on processor resources while improving the operation efficiency. The same study's significance test is conducted, and Table 5 displays the findings.

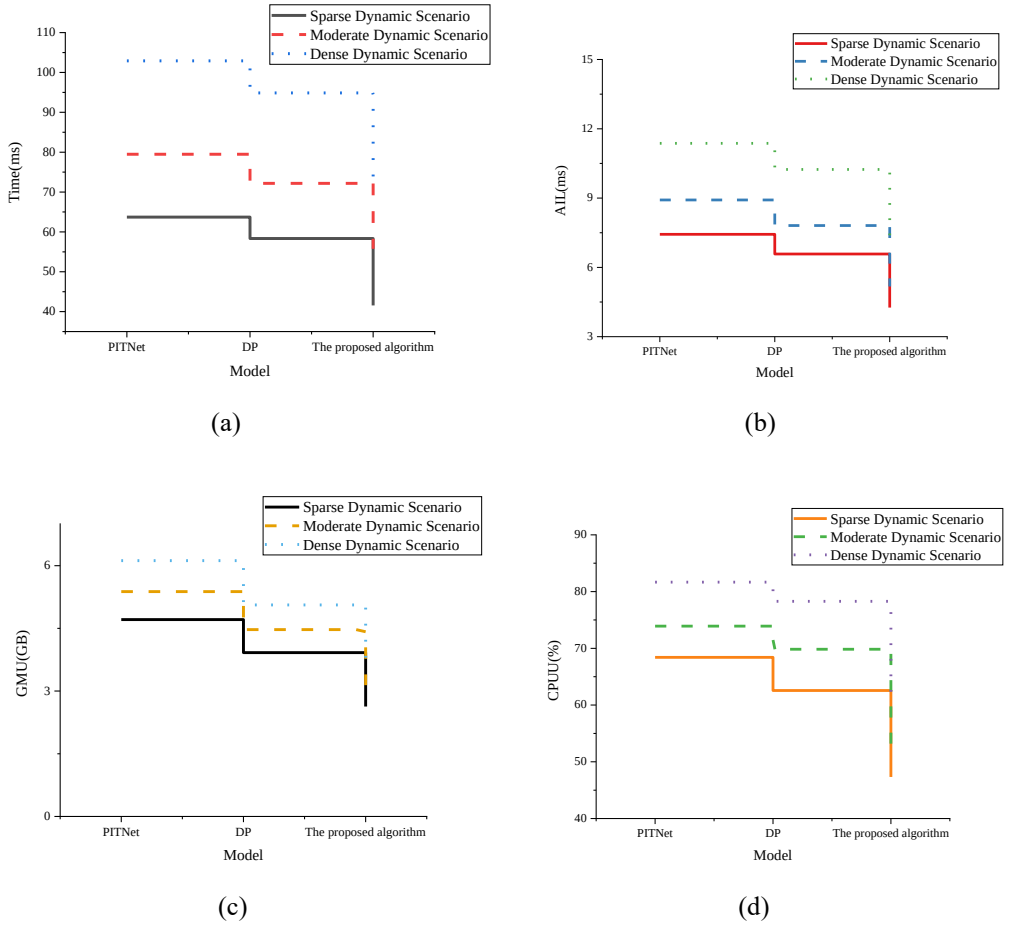

Figure 2. Real time performance and resource consumption (a) PT (Planning Time) (b) AIL (Average Inference Latency) (c) GMU (GPU Memory Usage) (d) CPUU (CPU Utilization).

Table 5. Significance test.

Metric	Comparison (Methods)	p-value	Cohen's d	Effect Size
PT	Proposed vs PITNet	0.009	1.27	Large
	Proposed vs DP	0.018	0.94	Large
AIL	Proposed vs PITNet	0.006	1.38	Large
	Proposed vs DP	0.027	0.89	Large
GMU	Proposed vs PITNet	0.012	1.11	Large
	Proposed vs DP	0.031	0.78	Medium
CPUU	Proposed vs PITNet	0.004	1.49	Large
	Proposed vs DP	0.023	0.96	Large

The results of significance test show that in the four indicators of real-time and resource consumption, the proposed optimized model has obvious advantages compared with the baseline method. Whether it is the overall trajectory PT, AIL,

GMU and CPUU, the performance improvement of this method has passed the statistical test, which shows that these differences are not caused by random disturbance, but the stable improvement brought by the model structure itself.

5.4. Ablation experiment

To analyze the contribution of each key module of the CSM to the overall performance, this study designs an ablation experiment, and compares the performance differences between the complete model and some deleted versions. The model set up in the study is displayed in Table 6.

The final output of w/o TF directly adopts the optimization stage trajectory, while the final output of other variants adopts the fused execution trajectory. The experimental results are presented in Table 7.

Table 6. Model settings of ablation experiment.

Method	Model details
The proposed algorithm (Full model)	A complete model, including risk perception generation module Risk-aware Generation (RG), uncertainty-aware safety constraint uncertainty-aware safety constraint (USC) and multi-source trajectory fusion mechanism trajectory fusion (TF). The final output adopts the fused execution trajectory.
w/o RG	Remove the risk perception generation module RG, and generate trajectory priors only based on basic condition diffusion; Keep USC and TF, and the final output adopts the fused execution trajectory. This variant lacks the ability to actively avoid high-risk areas in the generation stage.
w/o USC	Remove the uncertain perceptual security constraint USC, and only use the fixed safety distance boundary for optimization; RG and TF are reserved, and the final output adopts the fused execution trajectory. This variant lacks the ability of adaptive security expansion for prediction uncertainty.
w/o TF	Remove the multi-source trajectory fusion mechanism TF and directly use the optimized trajectory as the final execution trajectory.
w/o RG & USC	At the same time, the risk awareness generation module RG and the uncertain awareness security constraint USC are removed, and only the basic condition diffusion generation, back-end optimization and trajectory fusion processes are retained; The final output adopts the fused execution trajectory. This variant lacks both the ability of active risk avoidance in the generation stage and the ability of uncertain security boundary modeling in the optimization stage, and can be used to analyze the synergy between RG and USC.

Table 7. Results of ablation experiment.

Method	MTE (Overall)	PT (Overall) [ms]
The proposed algorithm (Full model)	1.59	56.73
w/o RG	1.88	59.84
w/o USC	1.93	60.91
w/o TF	1.77	62.37
w/o RG & USC	2.11	64.28

From Table 7, the average trajectory error of the optimized model in the whole scene is 1.59, which is obviously improved compared with 1.88 of w/o RG and 1.93 of w/o USC. At the same time, compared with w/o TF (1.77) which only removes the multi-source trajectory fusion mechanism. Both risk perception generation and uncertainty modeling can successfully increase the trajectory fitting accuracy in the dynamic environment, as evidenced by the lower error of the entire model. The improvement of MTE is the most obvious after removing USC, which shows that it is difficult to obtain sufficient stable trajectory quality only by relying on geometric obstacle avoidance constraints under the condition of prediction error and dynamic obstacle uncertainty. Furthermore, after jointly removing RG and USC, the overall MTE of the model rises to 2.11, which is significantly higher than that in any single-module removal case, indicating that the two modules do not exhibit obvious redundancy in improving trajectory accuracy but show strong synergistic effects. Among them, RG focuses on guiding the trajectory away from high-risk areas in advance during the generation stage, while USC further explicitly converts prediction errors into safety boundary constraints in the optimization stage. Only through their combined effect can a dual protection mechanism from "prior avoidance" to "constraint enhancement" be formed in dynamic environments. In addition, the overall PT of the proposed optimized model is 56.73 ms, which is significantly lower than 62.37 ms of w/o TF and slightly better than 59.84 ms and 60.91 ms of w/o RG and w/o USC. It demonstrates that by fully utilizing the complementarity between the generation prior and the local optimization, the multi-source trajectory fusion

mechanism shortens the overall PT and minimizes invalid iterations in the optimization process without appreciably increasing the additional computational overhead. Compared with the variant of removing RG or USC, the complete model can maintain better accuracy and lower planning latency, which shows that there is a good synergy between modules in design, rather than simple superposition. Meanwhile, the overall planning time of w/o RG & USC further increases to 64.28 ms, indicating that when risk guidance is missing in the generation stage and uncertainty-aware constraints are absent in the optimization stage, the back-end solving process requires more iterations to compensate for insufficient front-end priors, resulting in higher time overhead. It shows that RG and USC complement each other in terms of accuracy, and have a synergistic effect in computational efficiency.

6. Discussion

Combined with the experimental results, it shows that the proposed CSM exhibits better comprehensive performance in dynamic environments compared with PITNet and DP. This advantage is reflected in a single indicator, and in a more balanced and coordinated improvement among trajectory accuracy, safety margin, and real-time performance. Further analysis of the baseline models shows that the advantage of PITNet mainly lies in its strong physical consistency constraints, which can maintain good trajectory smoothness and certain dynamic executability. However, its ability to adapt to dynamic environmental changes is insufficient, especially when obstacle movements are frequent or prediction deviations are large, and trajectory accuracy and safety tend to degrade. DP has strong capabilities in distribution learning and diverse trajectory generation, and can form favorable candidate trajectories in complex scenarios. Nevertheless, the connection between its generation results and back-end control constraints is not tight enough, and the diffusion strategy itself incurs high inference overhead, resulting in shortcomings in real-time performance and execution stability. In contrast, CSM uniformly encodes multi-source environmental information, task semantics, and constraints, and continuously introduces risk and control information in the generation, optimization, and online correction stages. Therefore, it can achieve more stable performance advantages in complex dynamic scenarios.

From the perspective of internal model mechanism, the optimal results achieved in this study do not come from a simple superposition of multiple modules, but from the independent division and complementary coordination among each module. The risk-aware generation module mainly acts on the formation of front-end trajectory priors, so that candidate trajectories tend to avoid high-risk areas at the initial generation stage. The uncertainty-aware safety constraint module acts on the back-end optimization stage by explicitly incorporating prediction errors and safety boundaries into the constraint system, thereby enhancing the robustness of trajectories to dynamic uncertainty. The multi-source trajectory fusion module further integrates generation results, optimization results and online correction results at the execution level, enabling the system to maintain global planning capability while ensuring local control stability. This indicates that each module targets problems at different levels: the front-end focuses on advance avoidance, the middle-end on constraint enhancement, and the back-end on execution compensation. Therefore, the coexistence of these modules forms a continuous capability chain rather than functional redundancy.

From the perspective of engineering application, although the proposed method has shown favorable efficiency in resource consumption experiments, its practical deployment can be further optimized. Based on the existing results, future work can be promoted from three directions:

- Reduce the computational overhead of the generation module and fusion module through pruning, knowledge distillation, or low-rank decomposition to achieve model lightweighting.
- Adopt a distributed deployment strategy, placing environment perception and trajectory generation on edge computing nodes, and deploying MPC and online correction on the local control side to balance generative capability and control real-time performance
- Dynamically adjust the sampling steps and risk correction intensity according to scenario complexity, reducing inference cost in sparse scenarios and prioritizing safety in dense scenarios.

Overall, CSM demonstrates strong performance advantages at the experimental level, and has the potential to be further applied to practical dynamic robotic systems.

7. Conclusion

This study constructs a new framework of trajectory planning which integrates multi-source information fusion, condition generation, optimal control and online correction. The overall framework of CSM is put forward theoretically. It maps multi-source information such as environmental perception, task and constraint coding, robot's own state into a unified representation space, and realizes the modeling of trajectory prior distribution in dynamic environment through multi-modal cross-fusion and conditional diffusion generation. On the method level, this study designs some key modules, such as risk perception generation, uncertain security constraints and multi-source trajectory fusion, which make the generative prior and optimal control form a closed-loop coupling, and further move from "generative" to "executable" and "deployable". Although this study has made some progress in theoretical modeling and experimental results, there are still some shortcomings, which need to be further improved in the follow-up research. On the one hand, the modeling of environmental uncertainty mainly relies on Gaussian hypothesis based on covariance, and there are still limitations in dealing with non-Gaussian disturbances and emergencies in extreme scenes. Future research will focus on two core aspects: introducing more abundant environmental priors and incorporating advanced risk measurement methods. Specific candidate methods include distributed robust optimization-based joint modeling, scene generation confrontation networks, and multimodal perception techniques (e.g., vision, semantics, force perception). The goal of these improvements is to ensure that trajectory planning maintains reliable performance in more complex and open environments. On the other hand, the current CSM is mainly aimed at the trajectory planning of single manipulator, and the expansion of higher-dimensional and multi-agent scenes such as multi-manipulator cooperation and mobile chassis-manipulator integrated system remains to be studied. In the future, multi-agent collaborative planning and game constraints can be introduced based on the existing framework, thus supporting cooperative obstacle avoidance and task decomposition within complex operation units. In addition, although this study has achieved some optimization in real-time performance and resource consumption, it still relies mainly on a single high-performance computing platform. The future research can explore lightweight technologies such as model pruning, quantification, distillation, etc., and compress the CSM into embedded platforms or edge computing devices. It can also combine the distributed computing and scheduling mechanism to reasonably distribute the trajectory generation and optimization tasks among the cloud or multiple nodes to further enhance the overall scalability and deployment flexibility of the system. Through the continuous improvement in the above direction, it is expected that the proposed trajectory planning framework will be extended to larger and more complex real robot application scenarios.

References

1. Zhao S, Chen C, Li J, Gao S, Guo X. Trajectory Planning of Aerial Robotic Manipulator Using Hybrid Particle Swarm Optimization. Applied Sciences 2022; 12: 10892. https://doi.org/10.3390/app122110892.
2. Jia X, Zhao B, Liu J, Zhang S. A trajectory planning method for manipulators based on improved dynamic motion primitives. Industrial Robot: the international journal of robotics research and application 2024; 51(5): 847–856. https://doi.org/10.1108/IR-12-2023-0322.
3. Feng M, Dai J, Zhou W, Xu H, Wang Z. Kinematics analysis and trajectory planning of 6-DOF hydraulic manipulator in driving side pile. Machines 2024; 12(3): 191. https://doi.org/10.3390/machines12030191.
4. Zhang S, Xia Q, Chen M, Cheng S. Multi-objective optimal trajectory planning for manipulators using deep reinforcement learning. Sensors 2023; 23(13): 5974. https://doi.org/10.3390/s23135974.
5. Li X, Gu Y, Wu L, Sun Q, Song T. Time and energy optimal trajectory planning of wheeled mobile dual-arm robot based on tip-over stability constraint. Applied Sciences 2023; 13(6): 3780. https://doi.org/10.3390/app13063780.
6. Yang M, Zeng G, Ren Y, Lin L, Ke W. Accessibility and trajectory planning of cutter changing manipulator for large-diameter slurry shield. Mechanika 2023; 29(3): 214–224. https://doi.org/10.5755/j02.mech.30386.
7. Huda A N, Pebrianti D. Integrated Robotic Arm Control: Inverse kinematics, trajectory planning, and performance evaluation for automated welding. Asian Journal Science and Engineering 2023; 2(2): 82-100.

8. Song B Y, Li J Q, Liu X Y, Wang G L. A Trajectory Planning Method for Capture Operation of Space Manipulator Based on Deep Reinforcement Learning. Journal of Computing and Information Science in Engineering 2024; 24(9): 091003. https://doi.org/10.1115/1.4065814.
9. Long H, Li G, Zhou F, Chen T. Cooperative dynamic motion planning for dual manipulator arms based on RRT* Smart-AD algorithm. Sensors 2023; 23(18): 7759. https://doi.org/10.3390/s23187759.
10. Liu R, Pan F. A multi-objective trajectory planning method of the dual-arm robot for cabin docking based on the modified cuckoo search algorithm. machines 2024; 12(1): 64. https://doi.org/10.3390/machines12010064.
11. Ayazbay A A, Balabyev G, Orazaliyeva S, Gromaszek K, Zhauyt A. Trajectory Planning, Kinematics, and Experimental Validation of a 3D-Printed Delta Robot Manipulator. International Journal of Mechanical Engineering and Robotics Research 2024; 13(1): 113–125. http://doi.org/10.18178/ijmerr.13.1.113-125.
12. Scalera L, Giusti A, Vidoni R. Trajectory planning for intelligent robotic and mechatronic systems. Applied Sciences 2024; 14(3): 1179. https://doi.org/10.3390/app14031179.
13. Wang C, Yao X, Ding F, Yu Z. A trajectory planning method for a casting sorting manipulator based on a nature-inspired genghis khan shark optimized algorithm. Mathematical Biosciences and Engineering 2024; 21(2): 3364–3390. http: //doi.org/10.3934/mbe.2024149.
14. Guo H, Qiu Z, Gao G, Wu G, Chen H, Wang X. Safflower picking trajectory planning strategy based on an ant colony genetic fusion algorithm. Agriculture 2024; 14(4): 622. https://doi.org/10.3390/agriculture14040622.
15. Batista J G, Ramalho G L B, Torres M A, Oliveira A, Ferreira D. Collision avoidance for a selective compliance assembly manipulator manipulator using topological path planning. Applied Sciences 2023; 13(21): 11642. https://doi.org/10.3390/app132111642.
16. Yue C F, Lin T, Zhang X, Chen X, Cao X. Hierarchical path planning for multi-arm spacecraft with general translational and rotational locomotion mode. Science China Technological Sciences 2023; 66(4): 1180-1191. https://doi.org/10.1007/s11431-022-2275-2.
17. Xu Q, Lin Y. Research on high-precision motion planning of large multi-arm rock drilling robot based on multi-strategy sampling rapidly exploring random tree. Sensors 2025; 25(9): 2654. https://doi.org/10.3390/s25092654.
18. Xu K, Wang Z. The design of a neural network-based adaptive control method for manipulator trajectory tracking. Neural Computing and Applications 2023; 35(12): 8785–8795. https://doi.org/10.1007/s00521-022-07646-y.
19. Tian Y, Yue X, Wang L, Feng Y. Vibration suppression of collaborative robot based on modified trajectory planning. Industrial Robot: the international journal of robotics research and application 2023; 50(1): 45–55. https://doi.org/10.1108/IR-01-2022-0017.
20. Bektemessov A. A Survey of Path Planning and Obstacle Avoidance Techniques in Mobile Robotics. Engineering, Technology & Applied Science Research 2025; 15(6): 29632–29640. https://doi.org/10.48084/etasr.13885.
21. Wang P, Ma H, Zhang Y, Cao X, Wu X, Wei X, Zhou W.. Trajectory planning for coal gangue sorting robot tracking fast-mass target under multiple constraints. Sensors 2023; 23(9): 4412. https://doi.org/10.3390/s23094412.
22. Tang X, Zhou H, Xu T. Obstacle avoidance path planning of 6-DOF manipulator based on improved A* algorithm and artificial potential field method. Robotica 2024; 42(2): 457–481. https://doi.org/10.1017/S0263574723001546.
23. Chu C, Zhang H, Wang P, Lu F. Simulating human mobility with a trajectory generation framework based on diffusion model. International Journal of Geographical Information Science 2024; 38(5): 847–878. https://doi.org/10.1080/13658816.2024.2312199.
24. Ye J, Hao L, Cheng H. Multi-objective optimal trajectory planning for robot manipulator attention to end-effector path limitation. Robotica 2024; 42(6): 1761–1780. https://doi.org/10.1017/S0263574724000481.
25. Li S, Zou Y, Lai X, Liu Z, Wang X. Performance-maximum optimization of the intelligent lifting activities for a polar ship crane through trajectory planning. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 2023; 237(4): 765–781. https://doi.org/10.1177/09544062221111707.
26. Huang H H, Cheng C K, Chen Y H, et al. The manipulator velocity planning based on reinforcement learning. International Journal of Precision Engineering and Manufacturing 2023; 24(9): 1707–1721.
27. Ni J, Mei J, Ding Y, Yu D, Duan Y, Le Y. A trajectory planning approach for Delta robots considering both motion smoothness and dynamic stress. Journal of Mechanisms and Robotics 2023; 15(4): 041012.
28. Dai G, Zhang Q, Xu B. A novel framework for trajectory planning in manipulator developed by integrating dynamical movement primitives with particle swarm optimization. Scientific Reports 2025; 15(1): 29656. https://doi.org/10.1038/s41598-025-14801-7.

29. Yi Z, Li H, Zhu J, Feng B, Cao J, Lu X, Wang B. Motion planning method and experimental research of medical moxibustion robot of double manipulator arms. Journal of the Brazilian Society of Mechanical Sciences and Engineering 2024; 46(9): 564. https://doi.org/10.1007/s40430-024-05139-8.