MS-PPO: Mean Standard Deviation Proximal Policy Optimization for Reliable Parking Space Search in Structured Environments

Haoming Chen, Hongliang Guo*

College of Computer Science, Sichuan University
Chengdu, Sichuan 610207, China
{chenhaoming1, guohongliang}@scu.edu.cn

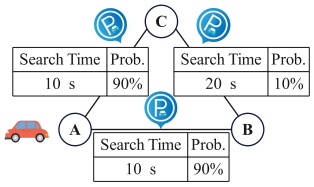
Abstract

This paper investigates the reliable parking space search problem in structured environments, with the objective of minimizing the linear combination of mean and standard deviation (mean-std) parking space search time. While canonical parking space search algorithms usually target the minimal expected search time, we argue that risk-averse users would like to trade expectation with its variance, leading to the reliable parking space search problem, which minimizes the mean-std search time. However, the non-additive nature of standard deviation makes the reliable parking space search problem difficult to solve with canonical search algorithms. To address the challenge, we propose a model-free reinforcement learning algorithm, namely mean standard deviation proximal policy optimization (MS-PPO), which simultaneously estimates the mean and standard deviation of the current decision-making policy's search time, and performs policy optimization via clipped mean-std advantage function minimization. MS-PPO is compared with several baseline parking space search algorithms as well as canonical reinforcement learning algorithms in a range of representative parking lot networks, and achieves the best overall performance in terms of the mean-std parking space search time. We also validate the effectiveness of MS-PPO in a real parking garage by deploying it to an autonomous vehicle testbed.

1 Introduction

Finding a free space to park in a crowded parking lot has been a common yet frustrating experience for many drivers. Upon entering, they often find themselves aimlessly circling through rows of parked cars, searching for an available spot. The process not only wastes significant time and increases stress, but also leads to unnecessary fuel consumption. As a result, the problem of parking space search has emerged as an important research topic and has attracted increasing attention from both academic researchers and industrial practitioners over the past several decades.

While a brief literature review of parking space search will be presented in Section 2, here, we wish to note that most existing algorithms focus solely on minimizing the expected search time, overlooking the impact of search time variance on risk-averse users. In practice, such users usually prefer a slightly longer but more predictable search process, rather than the one with smaller average search time but large variance. This motivates the study of *reliable* parking space search, which aims at minimizing a risk-sensitive objective defined as the linear combination of mean and standard deviation search time.



(a) A Simple Scenario

Method	Solution	Mean-Std (ζ=1)
Canonical Parking Alg.	A→B→C→A	12.9+8.75= 21.65
MS-PPO	A→B→A→C→B	12.2+6.87= **19.07**

(b) Search Time Comparison

Figure 1: (a) a simple yet illustrative example of the parking space search problem. The ego vehicle starts at Node A and needs to find an edge with free space to park itself. Each edge's search time and vacancy probability are depicted in the figure. (b) mean-std search time comparison between the canonical park space search algorithm and MS-PPO: (1) For the canonical parking space search algorithm, the chosen path is A $\rightarrow$ C $\rightarrow$ B $\rightarrow$ A, with a mean search time of 12.9, the variance is 76.59, and the mean-std search time is $12.9 + \sqrt{76.59} = 21.65$. (2) For MS-PPO, the selected path is A $\rightarrow$ B $\rightarrow$ A $\rightarrow$ C $\rightarrow$ B, with the mean search time as 12.2, the variance value is 47.16, and the mean+std search time is $12.2 + \sqrt{47.16} = 19.07$.

However, the non-additive characteristic of standard deviation makes the problem difficult to solve with canonical parking space search algorithms. Fig. 1a presents a simple yet illustrative example, showcasing that canonical search algorithms, *e.g.*, Euler path finding (Li et al. 2023), which minimize the mean search time, may produce a suboptimal solution to the reliable parking space search problem.

In this paper, we formulate the reliable parking space search problem within the Markov decision process (MDP) framework, and propose a model-free reinforcement learning (RL) algorithm, namely MS-PPO, as its solution. MS-PPO consists of (1) a mean standard deviation temporal difference (MS-TD) module, which simultaneously estimates the mean and standard deviation of the search time under

*Corresponding author: Hongliang Guo.

a given policy, (2) a mean standard deviation policy gradient (MS-PG) module, which estimates the policy gradient with respect to the mean-std objective, and (3) a mean-std proximal policy optimization (MS-PPO) module, which updates the decision-making policy by minimizing the clipped mean-std advantage function.

The paper's primary contributions can be summarized as follows: (1) We propose a new parking space search problem for risk-averse users, termed as *reliable* parking space search, which aims at minimizing the mean-std parking space search time; (2) We propose MS-PPO as an RL-based solution, which simultaneously estimates the mean and standard deviation of search time and updates the policy through clipped mean-std advantage function minimization; (3) We benchmark the reliability performance (mean-std parking space search time) of MS-PPO against several baseline search algorithms as well as mainstream RL algorithms across various parking lot networks, and further deploy it to an autonomous vehicle testbed to demonstrate its real-world applicability in a parking garage environment.

The remainder of the paper is organized as follows: we present a brief literature review of parking-related path planning and the mean standard deviation shortest path problem in Section 2, followed by the reliable parking space search problem formulation in Section 3. Section 4 presents the MS-PPO methodology, its pseudo code and the computational complexity analysis, followed by the reliability performance comparison among MS-PPO, baseline search algorithms and canonical RL algorithms in Section 5. We end the paper with conclusion and future work in Section 6 and put MS-PPO's real-world application and a demonstrative video in our GitHub repository due to page limitations.

2 Literature Review

This paper studies the mean-std parking space search problem, however, to the best of our knowledge, there exists no dedicated parking space search algorithms for the minimal mean-std objective. Therefore, this section conducts literature review on the two research topics separately, (1) parking space search; and (2) mean standard deviation navigation.

2.1 Parking Space Search

Path planning for parking is a crucial component of the overall parking process, as the quality of the planning algorithm directly affects both efficiency and user experience. (Han et al. 2024). Currently, parking path planning algorithms can be roughly classified into three categories: (1) graph theory (GT)-based algorithms, (2) heuristic algorithms, and (3) RL-based algorithms.

1) GT-based Algorithms: GT-based algorithms are either improved versions of traditional graph theory algorithms, such as Breadth-First Search (BFS) (Hassan et al. 2020) or Dijkstra's algorithm (Ata et al. 2021), or they model the parking path planning problem as classical graph theory problems, such as the Coverage Path Planning problem (CPP) (Ádám, Kocsány, and Szádeczky-Kardoss 2021) or the Traveling Salesman Problem (TSP) (Kocsány and Szádeczky-Kardoss 2022).

2) Heuristic Algorithms: Heuristic-based algorithms are typically optimized for specific problems encountered during the parking process. For instance, to address the issue of numerous disconnected paths in parking lots, a "fallback strategy" is introduced to prevent algorithms from falling into infinite loops or failing to converge (Wang, Shi, and Zhang 2020). For large underground parking garages with multiple entrances/exits and multi-objective parking space allocation and path planning problems, wolf pack foraging behavior is simulated for global search and local optimization (Dou, Lian, and Guo 2024). Mainstream algorithms include ant colony optimization (ACO) (Wang, Shi, and Zhang 2020) and genetic algorithm (GA) (Hao et al. 2020).

3) RL-based Algorithms: RL-based algorithms require the collection of actual driver parking trajectory and model the parking space search problem as a Markov decision process (MDP). These approaches utilize canonical reinforcement learning algorithms such as deep Q-Learning (DQN) (Jang and Lee 2022; Chen 2021), asynchronous advantage actor-critic (A3C) (Jang, Huang, and Chiu 2020; Tiong et al. 2022), and an enhanced Q-learning algorithm with optimized search (ARE-QL) (Zhang et al. 2025) to achieve real-time parking path planning.

2.2 Mean Standard Deviation Navigation

The mean-standard deviation shortest path is a type of reliable shortest path (RSP), which aims at finding the most "reliable" path in stochastic transportation networks. In the context of the parking space search problem, "reliability" can be defined as the minimum linear combination of the mean and standard deviation (Guo, Hou, and Peng 2022). In the following, we will present the mean-std navigation problem along the dimensions of its objectives, travel-time assumptions and mainstream methodologies.

1) Objective: The mean-std shortest path problem aims to find a path with the minimum linear combination of mean and standard deviation of travel time, explicitly balancing the trade-off between mean travel time and its variability (Song and Cheng 2022).

2) Travel-Time Assumptions: Travel time stochasticity the main source of uncertainty in transportation networks. Existing research can be categorized as either model-based, which assumes complete travel-time distributions (Guo et al. 2022) or requires first and second-order statistics (Hou, Guo, and Zhang 2020), or model-free, which directly uses empirical samples without the travel-time distribution assumptions (Xing and Zhou 2011). While most studies assume stationary travel time distributions (Guo, Hou, and Peng 2022), recent research increasingly considers the time-varying travel-time distributions (Chen et al. 2020).

3) Methodologies: Approaches to the mean-std shortest path problem can be categorized into: (1) dynamic programming (DP)-based algorithms that use label pruning and setting techniques to iteratively update node values (Chen et al. 2020; Bertsekas 2018), (2) mathematical programming (MP)-based algorithms that formulate the problem within the optimization frameworks and use either commercial solvers or Lagrangian relaxation techniques (Zhang and Khani 2019) to yield the solution, and (3) learning-based

methods, which let the ego vehicle interact with the environment and learn to make optimal decisions (Guo, Hou, and Peng 2022).

3 Problem Formulation

This section presents the mathematical formulation of the reliable parking space search problem and then recasts it within the Markov decision process (MDP) framework. Table I lists the major notations used throughout the paper, and brief descriptions will be provided upon the first appearance in the main text.

TABLE I: List of major notations used in the paper

Notations	Descriptions
$\mathcal{G}(\mathcal{V},\mathcal{E})$	The parking lot network
$\mathcal{V}$	Set of vertices in the graph
$\mathcal{E}$	Set of edges in the graph
$v_i \in \mathcal{V}$	a vertex in $\mathcal{G}$
$e_{ij} \in \mathcal{E}$	The edge connecting vertices v_i and v_j
π	The navigation policy
p_{ij}	Probability that edge e_{ij} has a vacant parking space
t_{ij}	Edge e_{ij}'s travel time (mean: μ_{ij}, standard deviation: σ_{ij})
t_{tot}	The total parking space search time, a random variable
$V^\pi(s)$	Mean state value function under π
$\bar{V}^\pi(s)$	Variance state value function under π
$Q^\pi(s,a)$	Mean action value function under π
$\bar{Q}^\pi(s,a)$	Variance action value function under π
α	learning rate for mean value function
$\bar{\alpha}$	learning rate for variance value function

3.1 Parking Space Search Problem Formulation

The problem studied in this paper is to navigate the ego vehicle in the parking lot with the minimal mean-std parking space search time. The parking lot network is modeled as an undirected and connected graph, *i.e.*, $\mathcal{G}(\mathcal{V},\mathcal{E})$, where $\mathcal{V}$ represents the set of vertices, and $\mathcal{E}$ represents the set of edges in Graph $\mathcal{G}$. At each vertex $v_i \in \mathcal{V}$, the navigation policy π decides which edge $e_{ij} \in \mathcal{E}$ to execute in the next time step. For each edge $e_{ij} \in \mathcal{E}$, there is a probability (p_{ij}) that the edge has a vacant parking space, and the edge's travel time is a random variable (t_{ij}), whose mean and standard deviation values are denoted as μ_{ij} and σ_{ij}, respectively.

Whenever the ego vehicle executes an edge with a vacant parking space, the search problem terminates, and the total search time t_{tot} is deemed as the actual parking space search time. Note that t_{tot} is a random variable in that each edge's actual travel time (t_{ij}) and whether an edge has a vacant parking space are both random variables. The objective of the reliable parking space search problem is to find the optimal navigation policy π, which has the minimal mean-std parking space search time, *i.e.*, $\pi^* = \arg,\min_\pi \mathbb{E}(t_{\text{tot}}) + \zeta\,\sigma(t_{\text{tot}})$, where $\zeta > 0$ is the coefficient balancing the mean and standard deviation search time.

3.2 Transformation within the MDP Framework

Since MS-PPO is essentially an RL-based algorithm for the reliable parking space search problem, this subsection transforms the parking space search problem within the MDP framework by mapping MDP-related elements, *i.e.*, $\mathcal{S}, \mathcal{A}, R(s,a), \mathbb{P}(s'|s,a), \gamma$, into parking path planning related variables, and express the corresponding mean-std parking space search time objective with the MDP terminology.

The mapping process is as follows: (1) $s \in \mathcal{S}$ refers to the ego vehicle's state, which consists of the vehicle's current residing vertex, and its past trajectory, *i.e.*, all edges that have been executed by the vehicle; (2) $a \in \mathcal{A}(s)$ refers to the ego vehicle's action, which corresponds to the vehicle's executing edge in the next time step; (3) $r \in R(s,a)$ denotes the vehicle's immediate reward, which corresponds to the consumed travel time after executing the current edge; (4) $\mathbb{P}(s'|s,a)$ refers to vehicle's state transition process, which corresponds to the next step state's probability distribution given the current state s and action a[1]; (5) γ refers to the discount factor, and is set to be 1 in the parking space search context. The total search time t_{tot} corresponds to the cumulative *undiscounted* reward in the MDP context. The objective is to find a decision-making policy π, which minimizes the mean-std of the cumulative reward, *i.e.*, $\pi^* = \arg,\min_\pi \mathbb{E}(\sum_{k=0}^{\infty} r_k) + \zeta\sigma(\sum_{k=0}^{\infty} r_k)$. Note that in this paper, we adopt the same convention used in canonical RL by treating the rewards collected after the absorbing state as 0.

4 Methodology

This section presents the mean standard deviation proximal policy optimization (MS-PPO) methodology, which aims at finding the optimal policy with the minimal mean-std parking space search time. In the following subsections, we will present (1) the mean-variance Bellman equation, which serves as the theoretical foundation of MS-PPO; (2) mean-std temporal difference (MS-TD), which simultaneously estimates the mean and standard deviation parking space search time; (3) mean-std policy gradient (MS-PG), which updates the policy towards optimum with respect to the mean-std objective; (4) mean-std actor-critic (MS-AC), which simultaneously updates the mean-std actor network and the two critic networks; and (5) mean-std proximal policy optimization (MS-PPO), which performs policy optimization through clipped mean-std advantage function minimization. We end the section with the computational complexity analysis of the MS-PPO algorithm. Fig. 2 presents the overall framework of MS-PPO.

4.1 Mean-Variance Bellman Equation

The canonical Bellman equation, *i.e.*, the mean Bellman equation, describes the recursive relationship between the expected return of a state and its successor states' expected return. In this subsection, we extend the canonical Bellman

[1]Note that in the parking context, when the ego vehicle executes edge ij at vertex i, it either reaches vertex j as its next state or it finds an available parking space and thus reaches the absorbing state (s_o) and terminates the parking space search process.

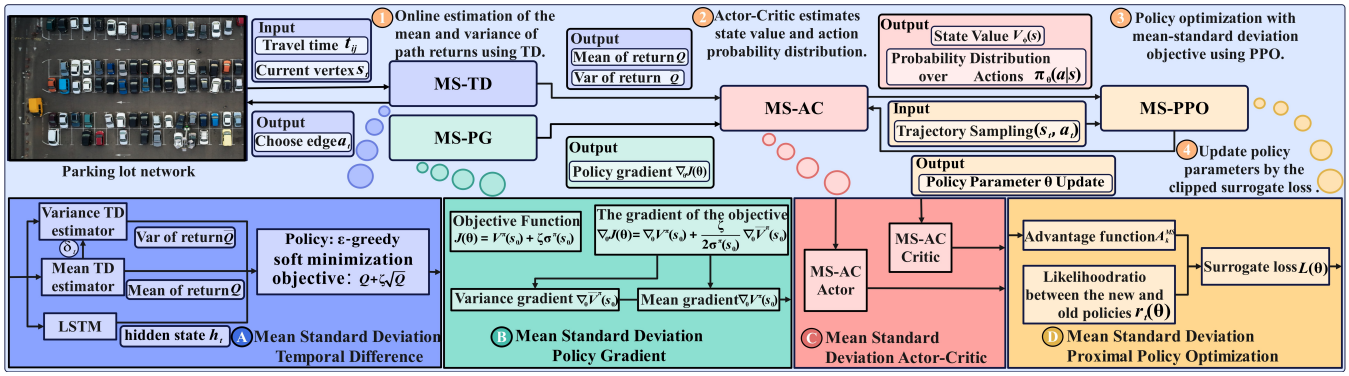


Figure 2: The MS-PPO Framework: the ego vehicle navigates in the parking lot network, takes its current residing node and past trajectory as inputs. Ⓐ MS-TD estimates the mean and variance of the policy's return; Ⓑ MS-PG derives the policy gradient with respect to the mean-std objective; Ⓒ MS-AC combines MS-TD with MS-PG, where the two critic networks estimates the mean and variance of the return and the actor network updates the policy with respect to the mean-std objective; Ⓓ MS-PPO enhances MS-AC with a clipped mean-std advantage function, preventing large policy updates and ensuring stable convergence.

equation to the variance form, which captures the recursive relationship between the variance of the current state's return, and the variance of the successor states' return.

When given $s \in \mathcal{S}$, $a \in \mathcal{A}$, a proper policy $\pi(a|s)$, $\mathbb{P}(s'|s,a)$, $R(s,a)$, and γ, we have the following mean Bellman equation for state-value function, which is the same as the canonical Bellman equation[2]:

$$V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} \mathbb{P}(s'|s,a)\big[R(s,a) + V^\pi(s')\big], \quad (1)$$

where $V^\pi(s)$ refers to the expected return under π when the agent starts at s. Next, we define $\bar{V}^\pi(s)$ as the variance of the return under policy π when the agent starts at s, and present the variance Bellman equation in a theorem form as follows:

Theorem 1 (Variance Bellman Equation for State-Value Function)**.** *The variance value function satisfies:*

$$\bar{V}^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} \mathbb{P}(s'|s,a)\big[\mathbb{E}(\delta^2) + \bar{V}^\pi(s')\big], \quad (2)$$

where $\delta = r + V^\pi(s') - V^\pi(s)$ refers to the sampled temporal difference (TD) error.

The proof of Theorem 1 makes use of the law of total expectation in Probability theory, and the Markov property, and is presented in our GitHub repository. Eq. (1) and Eq. (2) together constitute the mean-variance Bellman equation set for state-value function. Now, we directly present the mean-variance Bellman equation set for action-value functions, *i.e.*, $Q^\pi(s,a)$ and $\bar{Q}^\pi(s,a)$ as follows:

$$Q^\pi(s,a) = \sum_{s'} \mathbb{P}(s'|s,a)\Big[R(s,a) + \sum_{a'} \pi(a'|s') Q^\pi(s',a')\Big] \quad (3)$$

$$\bar{Q}^\pi(s,a) = \sum_{s'} \mathbb{P}(s'|s,a)\Big[\mathbb{E}[\delta^2] + \sum_{a'} \pi(a'|s') \bar{Q}^\pi(s',a')\Big]. \quad (4)$$

[2]Note that since $\gamma = 1$ in the parking space search problem, we directly omit it in the equation expression.

4.2 Mean-Std Temporal Difference (MS-TD)

The mean-variance Bellman equation set introduced in the previous subsection provides the theoretical foundation to simultaneously estimate the mean and variance of the given policy's return. However, in many practical scenarios, the exact model dynamics, *e.g.*, $\mathbb{P}(s'|s,a)$, $\mathbb{E}(\delta^2)$, are not accessible, rendering model-based solutions inapplicable. Therefore, this subsection presents the mean-std temporal difference (MS-TD) algorithm, which estimates the mean and standard deviation of a given policy's return in a model-free manner.

First the canonical one-step temporal difference (TD) learning method, *i.e.*, SARSA (Sutton and Barto 2018), for the mean value function estimation is expressed as:

$$V^\pi(s_k) \leftarrow V^\pi(s_k) + \alpha_k \left(r_{k+1} + V^\pi(s_{k+1}) - V^\pi(s_k)\right), \quad (5)$$

where $0 < \alpha_k < 1$ is the learning rate. Next, we deliver the TD-learning method for the variance value function estimation in the following theorem, followed by the proof process.

Theorem 2 (The Variance Value Function Theorem)**.** *For a given policy π, the variance value function $\left(\bar{V}(s)\right)$ can be estimated online as:*

$$\bar{V}^\pi(s_k) \leftarrow \bar{V}^\pi(s_k) + \bar{\alpha}_k(\delta_k^2 + \bar{V}^\pi(s_{k+1}) - \bar{V}^\pi(s_k)), \quad (6)$$

where $\delta_k = r_{k+1} + V^\pi(s_{k+1}) - V^\pi(s_k)$ is the k^{th} time step TD error, and $0 < \bar{\alpha}_k < 1$ is the variance learning rate.

Proof. Let G_k (random variable) be the return from state s_k following π, and in this case, $\mathrm{Var}(G_k) = \bar{V}^\pi(s_k)$. We have:

$$\begin{aligned}
&\mathrm{Var}(G_k) \\
&= \mathrm{Var}(r_{k+1} + G_{k+1}) \\
&= \mathbb{E}\big(r_{k+1} + G_{k+1} - \mathbb{E}(G_k)\big)^2 \\
&= \mathbb{E}\big(r_{k+1} + G_{k+1} - V^\pi(s_k)\big)^2 \\
&= \mathbb{E}\big(r_{k+1} + G_{k+1} - V^\pi(s_k) + V^\pi(s_{k+1}) - V^\pi(s_{k+1})\big)^2 \\
&= \mathbb{E}\big(r_{k+1} + V^\pi(s_{k+1}) - V^\pi(s_k) + G_{k+1} - V^\pi(s_{k+1})\big)^2
\end{aligned}$$

$$= \mathbb{E}\big(r_{k+1} + V^\pi(s_{k+1}) - V^\pi(s_k)\big)^2 + \mathbb{E}\big(G_{k+1} - V^\pi(s_{k+1})\big)^2$$
$$+2\mathbb{E}\big(r_{k+1} + V^\pi(s_{k+1}) - V^\pi(s_k)\big)\mathbb{E}\big(G_{k+1} - V^\pi(s_{k+1})\big)$$
$$= \mathbb{E}(\delta_k^2) + \mathrm{Var}(G_{k+1}),$$

where the last equality uses the fact that $\mathbb{E}[G_{k+1} - V^\pi(s_{k+1})] = 0$. The TD update approximates $\mathbb{E}[\delta_t^2]$ with the sampled one, *i.e.*, δ_t^2. □

The key insight is that we can decompose the variance of returns into the immediate variance contribution (captured by δ_t^2) and the future variance (bootstrapped from $\overline{V}^\pi(s_{t+1})$). The decomposition enables the online updates without waiting for episode completion, dramatically improving learning efficiency in the parking space search context. The extension to varince action-value functions follows naturally, enabling us to estimate both $Q^\pi(s,a)$ and $\bar{Q}^\pi(s,a)$ online:

Remark 1 (Action-Value MS-TD Update Theorem)**.** *The mean and variance of action values can be updated using:*

$$Q^\pi(s_k,a_k) \leftarrow Q^\pi(s_k,a_k) + \alpha_k\Big(r_{k+1} + Q^\pi(s_{k+1},a_{k+1}) - Q^\pi(s_k,a_k)\Big),$$
$$\bar{Q}^\pi(s_k,a_k) \leftarrow \bar{Q}^\pi(s_k,a_k) + \bar{\alpha}_k\Big(\delta_k^2 + \bar{Q}^\pi(s_{k+1},a_{k+1}) - \bar{Q}^\pi(s_k,a_k)\Big).$$

where $\delta_k = r_{k+1} + Q^\pi(s_{k+1},a_{k+1}) - Q^\pi(s_k,a_k)$.

So far, we have presented MS-TD which functions as a model-free reinforcement learning algorithm simultaneously estimating the mean and variance of a given policy's return. The complete pseudo code of MS-TD for both the mean-variance state-value and action-value function estimation is provided in our GitHub repository. Note that, when we apply the ϵ-greedy policy update rule (Sutton and Barto 2018) with respect to the mean-std objective, MS-TD is able to function independently for the mean-std parking space search time minimization, and we will perform the ablation study of MS-PPO by treating MS-TD as an independent functional module in Section 5.3.

4.3 Mean-Std Policy Gradient (MS-PG)

With the mean and variance TD estimators established in the previous subsection, we now derive the policy gradient with respect to the mean-std optimization objective. The key result is summarized in the following mean-std policy gradient theorem, with the complete derivation provided thereafter.

Theorem 3 (The Mean-Std Policy Gradient Theorem)**.** *For the objective* $J(\boldsymbol{\theta}) = V^\pi(s_0) + \zeta\sigma^\pi(s_0)$*, where* $\sigma^\pi(s_0) = \sqrt{\overline{V}^\pi(s_0)}$*, the gradient of* $J(\boldsymbol{\theta})$ *can be expressed as:*

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{s,a\sim d^{\pi_{\boldsymbol{\theta}}}}\Bigg[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\Big(Q^\pi(s,a) + \frac{\zeta}{2\sigma^\pi(s_0)} \Big(\bar{Q}^\pi(s,a) + \big(Q^\pi(s,a)\big)^2 - 2V^\pi(s_0)Q^\pi(s,a),\Big)\Big)\Bigg] \tag{7}$$

where $d^{\pi_{\boldsymbol{\theta}}}$ *refers to the undiscounted state-action distribution under policy* π.

Proof.

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$
$$= \nabla_{\boldsymbol{\theta}} V^\pi(s_0) + \frac{\zeta}{2\sigma^\pi(s_0)} \nabla_{\boldsymbol{\theta}} \overline{V}^\pi(s_0)$$
$$= \nabla_{\boldsymbol{\theta}} V^\pi(s_0) + \frac{\zeta}{2\sigma^\pi(s_0)} \left(\nabla_{\boldsymbol{\theta}} \mathbb{E}_\pi[G_0^2] - 2V^\pi(s_0)\nabla_{\boldsymbol{\theta}} V^\pi(s_0)\right)$$
$$= \mathbb{E}_{s,a\sim d^{\pi_{\boldsymbol{\theta}}}}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) Q^\pi(s,a) + \frac{\zeta}{2\sigma^\pi(s_0)} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) \big(\bar{Q}^\pi(s,a) + (Q^\pi(s,a))^2\big) - 2V^\pi(s_0)Q^\pi(s,a))\Big]$$
$$= \mathbb{E}_{s,a\sim d^{\pi_{\boldsymbol{\theta}}}}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) \big(Q^\pi(s,a) + \frac{\zeta}{2\sigma^\pi(s_0)} \big(\bar{Q}^\pi(s,a) + (Q^\pi(s,a))^2 - 2V^\pi(s_0)Q^\pi(s,a)\big)\big)\Big]$$

□

With Theorem 3, one can update the policy parameters towards the minimal mean-std return direction, and the pseudo code of the MS-PG algorithm is provided in the supplementary material.

4.4 Mean Standard Deviation PPO (MS-PPO)

Mean Standard Deviation Actor-Critic. To effectively combine MS-TD with MS-PG, we develop the MS-AC algorithm, which contains two critic networks estimating the mean and variance of the policy's return, respectively, and one actor network, which updates the policy's parameter vector based on MS-PG's output. The pseudo code of MS-AC is presented in our GitHub repository due to space limitation.

Inspired by the success of proximal policy optimization (PPO) in maintaining the stable policy optimization process, we further develop the MS-PPO algorithm, which builds upon MS-AC while incorporating PPO's conservative update strategy, integrating all previous components into a practical and robust solution for the reliable parking space search. Before presenting the MS-PPO algorithm, we define the mean-std advantage function as follows:

Definition 1 (Mean-Std Advantage Function)**.** *The advantage function for the mean-std objective is defined as:*

$$A_k^{MS} = Q^\pi(s_k,a_k) - V^\pi(s_k) + \zeta\Big(\sqrt{\bar{Q}^\pi(s_k,a_k)} - \sqrt{\bar{V}^\pi(s_k)}\Big) \tag{8}$$

where $Q^\pi(s_k,a_k)$ *and* $V^\pi(s_k)$ *are the mean action-value function and mean state-value function, respectively,* $\bar{Q}^\pi(s_k,a_k)$ *and* $\bar{V}^\pi(s_k)$ *are the variance action-value function and variance state-value function, respectively, and* $\zeta > 0$ *is the reliability coefficient.*

Definition 2 (MS-PPO Surrogate Objective)**.** *The clipped surrogate objective for MS-PPO that ensures monotonic improvement while optimizing the mean-std objective is:*

$$\mathbb{L}(\boldsymbol{\theta}) = \mathbb{E}\big[\min\big(r_k(\boldsymbol{\theta})A_k^{MS}, \mathrm{clip}(r_k(\theta), 1-\kappa, 1+\kappa)A_k^{MS}\big)\big] \tag{9}$$

where $r_k(\boldsymbol{\theta}) = \frac{\pi_{\boldsymbol{\theta}}(a_k|s_k)}{\pi_{\boldsymbol{\theta}_{\mathrm{old}}}(a_k|s_k)}$ *is the importance sampling ratio between the evaluation policy and the behavior policy, and* $0 < \kappa < 1$ *is the clipping parameter.*

The clipping mechanism in Eq. (9) serves a dual purpose: (1) it prevents destructively large policy updates when $A_k^{\mathrm{MS}} > 0$ and removes incentives for excessively reducing action probabilities when $A_k^{\mathrm{MS}} < 0$. The MS-PPO algorithm is then to minimize the clipped mean-std advantage function, *i.e.*, $\boldsymbol{\theta}_{k+1} = \arg,\min_{\boldsymbol{\theta}} \mathbb{L}(\boldsymbol{\theta})$. The pseudo code of MS-PPO is presented in Algorithm 1.

Algorithm 1: MS-PPO's Algorithm Flow Process

Input: (1) learning rate α, $\bar{\alpha}$;
(2) reliability coefficient ζ; (3) clipping parameter κ;
(4) number of epochs K; (5) max. iteration N;
(6) current policy $\pi_{\boldsymbol{\theta}}$;
(7) mean and variance value functions $Q(s,a)$, $\bar{Q}(s,a)$, $V(s)$, $\bar{V}(s)$;
Output: The optimized mean-std parking space search policy $\pi_{\boldsymbol{\theta}}$

1 **for** *iteration* $= 1$ **to** N **do**
2 **for** *each episode* **do**
3 Initialize s_0;
4 **for** $t = 0, 1, \ldots, T$ **do**
5 Sample $a_k \sim \pi_\theta(a_k|s_k)$;
6 Execute a_k, observe r_{k+1}, s_{k+1};
7 Store $(s_k, a_k, r_{k+1}, s_{k+1})$ into trajectory buffer $\mathcal{D}$;
8 **if** s_{k+1} *is terminal* **then**
9 break;
10 **for** *each* (s_k, a_k) *in* $\mathcal{D}$ **do**
11 Estimate $Q(s_k, a_k)$ and $\bar{Q}(s_k, a_k)$ using MS-TD;
12 Estimate $V(s_k)$ and $\bar{V}(s_k)$ using MS-TD;
13 **for** *each* (s_k, a_k) *in* $\mathcal{D}$ **do**
14 Compute A_k^{MS} using Eq. (8);
15 **for** *epoch* $= 1$ **to** K **do**
16 **for** *each mini-batch* $\mathcal{B}$ *from* $\mathcal{D}$ **do**
17 Compute $r_k(\boldsymbol{\theta})$ Compute surrogate loss $\mathbb{L}(\theta)$ using Eq. (9);
18 Update policy parameters $\boldsymbol{\theta}$ by minimizing $\mathbb{L}(\boldsymbol{\theta})$;
19 **Final**: Return the trained policy $\pi_{\boldsymbol{\theta}}$.

4.5 Computational Complexity Analysis

In this subsection, we use the big $\mathcal{O}$ notation (Knuth 1976) to analyze the computational complexity of MS-PPO's training process as depicted in Algorithm 1. Line 2-9 collect M episodes with average trajectory length $\bar{T}$, where each action sampling from π_θ requires $\mathcal{O}(d^2)$ operations for a neural network with hidden dimension d. Lines 10-12 update four value functions using MS-TD, each requiring $\mathcal{O}(d^2)$ per update for $M\bar{T}$ samples. Line 13-14 compute the mean-std advantages, requiring $\mathcal{O}(d^2)$ forward passes per sample. Lines 15-19 perform K epochs of PPO updates on mini-batches, with complexity $\mathcal{O}(d^2)$ for computing importance ratios and gradient updates.

Therefore, the total computational complexity of MS-PPO's training process can be expressed as: $\mathcal{C}_{\text{MS-PPO}} = \mathcal{O}(M\bar{T}d^2) + \mathcal{O}(4M\bar{T}d^2) + \mathcal{O}(4M\bar{T}d^2) + \mathcal{O}(KM\bar{T}d^2) = \mathcal{O}(KM\bar{T}d^2)$, after absorbing the non-leading terms. We can see that MS-PPO achieves the polynomial computational complexity with respect to related parameters.

5 Simulation Results and Analysis

In this section, we evaluate and compare the mean-std performance of MS-PPO with baseline parking space search solutions as well as canonical reinforcement learning algorithms a range of parking lot networks. Specifically, we select (1) asynchronous advantage actor-critic (A3C) (Jang, Huang, and Chiu 2020), (2) deep Q-Learning (DQN) (Boulares, Fehri, and Jemni 2024), (3) proximal policy optimization with asynchronous advantage actor-critic (A3C-PPO) (Tiong et al. 2022), (4) improved sparrow search algorithm (ISSA) (Si et al. 2025), (5) improved ACO (Geng and Sun 2023), and (6) the Dijkstra-ant colony algorithm with binary search tree (DABST) (Ata et al. 2021). Note that A3C, DQN, and A3C-PPO are recently proposed RL-based methods for parking space search problems; ISSA and ACO are recently proposed heuristic-based algorithms for parking space search; and DABST is a recently proposed graph theory-based algorithm. Additionally, since MS-PPO is essentially a coverage path planning algorithm, we also implement the Chinese postman problem algorithm (CPP) (Sun et al. 2022) as a comparative baseline. Note that CPP aims at finding the shortest closed path, which corresponds to the minimal search time in the parking space search problem. All algorithms' meta-parameter configurations are displayed in our GitHub repository.

In the following subsections, we will (1) evaluate the functional modules within MS-PPO, namely MS-TD, MS-PG, MS-AC, and MS-PPO, using the simple yet illustrative network as shown in Fig. 1a; (2) benchmark MS-PPO's mean-std performance and computation efficiency in terms of decision-making time baseline algorithms in the three real parking lot networks, namely Shenzhen, Singapore, and Chicago; and (3) perform the ablation study of MS-PPO across the three parking lot networks. The source code of MS-PPO and dataset for the three parking lot networks are publicly available[3].

5.1 Performance Evaluation in a Simple Network

In this subsection, we conduct the functional testing of each module within MS-PPO on the simple network shown in Fig. 1a. We assume that the edge distribution model follows the normal distribution, which allows us to precisely calculate the mean-std values for any given policy's resulting path, thereby enabling evaluation of the correctness and optimality of all modules within MS-PPO.

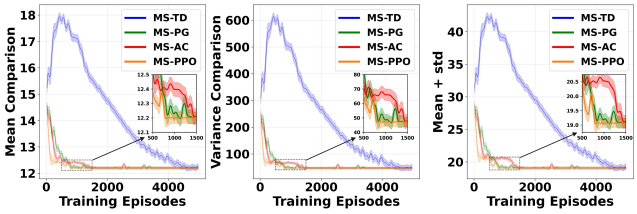


Figure 3: Learning curve comparison of MS-PPO modules.

[3]https://github.com/Charming1920/Mean-Standard-Deviation-Proximal-Policy-Optimization

TABLE II: Performance and efficiency comparison between MS-PPO and baseline algorithms.

Testing Network	ζ	Mean + ζ × Std Value								Total decision-making/planning time (seconds)							
		MS-PPO	DQN	A3C	A3C-PPO	ACO	ISSA	DABST	CPP	MS-PPO	DQN	A3C	A3C-PPO	ACO	ISSA	DABST	CPP
Shenzhen	0.1	**53.39**	77.78	74.15	76.86	77.91	59.81	83.97	59.85	1.6e-2	1.7e-2	2.0e-2	1.7e-2	8.9e-4	6.8e-1	**3.1e-4**	3.0e-3
	1	**58.71**	83.96	79.32	83.46	83.88	64.08	77.78	65.33	1.2e-2	2.6e-2	2.5e-2	1.2e-2	1.0e-3	4.6e-1	**3.1e-4**	2.3e-3
	10	**111.96**	138.81	131.03	119.58	143.58	116.81	138.71	120.07	1.2e-2	1.9e-2	2.0e-2	1.7e-2	8.9e-4	6.8e-1	**4.9e-4**	3.0e-3
Singapore	0.1	**43.41**	44.22	109.71	44.29	112.41	53.98	43.49	124.85	1.8e-2	1.6e-2	2.3e-2	3.3e-2	1.8e-3	9.8e-1	**6.4e-4**	5.5e-3
	1	**49.78**	51.25	116.79	50.59	119.94	60.65	50.80	131.15	1.2e-2	1.7e-2	3.2e-2	2.8e-2	1.3e-3	6.6e-1	**5.4e-4**	5.5e-3
	10	**113.42**	121.54	187.66	113.59	195.24	118.78	123.92	194.15	1.4e-2	2.0e-2	4.6e-2	3.2e-2	8.5e-4	6.3e-1	**4.4e-4**	5.5e-3
Chicago	0.1	**55.25**	60.07	161.83	60.36	56.53	62.87	60.06	156.59	2.0e-2	2.3e-2	4.3e-2	3.1e-2	1.8e-3	7.1e-1	**6.9e-4**	6.3e-3
	1	**61.99**	67.43	168.32	67.89	74.73	67.59	67.37	163.33	1.9e-2	2.3e-2	4.1e-2	3.2e-2	1.8e-3	7.3e-1	**7.3e-4**	7.0e-3
	10	129.34	141.10	233.22	143.19	256.74	**127.38**	140.49	230.68	1.8e-2	2.1e-2	4.1e-2	3.1e-2	1.8e-3	7.2e-1	**5.8e-4**	6.4e-3

Fig. 3 sequentially demonstrates the learning process of mean, variance, and mean + $\zeta\times$ std for each MS-PPO module when $\zeta = 1$. From the figure, we can observe that: (1) All modules eventually converge to the correct theoretical values shown in Fig. 1b. (2) MS-TD exhibits slightly slower convergence and larger oscillation when compared to other modules. We attribute this to the fact that MS-TD uses the ϵ-greedy strategy to update the policy which may incur huge policy update during the learning process. (3) Among the three policy gradient modules, MS-PPO converges faster than the other two modules, due to MS-PPO's conservative yet more stable policy optimization procedure.

5.2 Performance Comparison with Baseline Algorithms in Three Parking Lot Networks

This subsection compares the performance and efficiency of MS-PPO with baseline algorithms across three typical real-world parking lot networks: Shenzhen, Singapore, and Chicago. The parking lot network details are provided in the appendix. For performance evaluation, we randomly generate 50 points as starting locations for the parking space search process. For each starting point, we evaluate three use cases with $\zeta = 0.1$, $\zeta = 1$, and $\zeta = 10$. We use mean $+ \zeta \times$ std computed across all 50 starting points as the performance metric for comparison.

Table II presents the performance and efficiency comparison between MS-PPO and baseline algorithms across the three real parking networks. From the results, we can observe that: (1) the performance patterns of almost all algorithms are consistent, with mean $+ \zeta \times$ std values increasing as we increase the reliability coefficient ζ; (2) on average, MS-PPO achieves the best overall performance across all three parking networks; (3) when the reliability coefficient ζ is low, MS-PPO produces very similar performance metrics to other algorithms, as MS-PPO also tends to find paths with the smallest mean search time when the ζ value is small, aligning with the objectives of other algorithms.

For a well-trained MS-PPO network, the decision-making time depends solely on the MS-Actor network, which involves simple algebraic calculations. The right half part of Table II shows the decision-making time comparison across all algorithms, showing that DABST's decision-making time is the smallest, but MS-PPO already achieves real-time performance with latency under 0.1 seconds.

5.3 Ablation Study of MS-PPO

This subsection conducts the ablation study by comparing the performance of MS-TD, MS-PG, MS-AC, and MS-PPO. Table III presents the results of our ablation study, revealing several noteworthy findings: (1) For large-scale parking networks such as Singapore and Chicago, when $\zeta = 0.1$, all four algorithms exhibit remarkably good and nearly indistinguishable performance. (2) For the case of $\zeta = 10$, MS-PPO demonstrates significantly superior performance compared to other variants across all three testing parking lot networks. We attribute it to the fact that MS-PPO performs the policy optimization procedure in a more conservative yet stable manner, while MS-TD, MS-PG and MS-AC are prone to be affected by the sampled TD error.

TABLE III: Ablation study of MS-PPO

Testing Network	ζ	Mean + ζ × Std Value			
		MS-PPO	MS-TD	MS-AC	MS-PG
Shenzhen	0.1	**53.39**	59.27	59.65	61.58
	1	**58.71**	66.79	65.41	64.62
	10	**111.96**	128.81	123.04	125.17
Singapore	0.1	**43.41**	44.78	44.41	44.43
	1	**49.78**	52.31	51.67	51.58
	10	**113.42**	127.61	124.23	123.01
Chicago	0.1	**55.25**	57.99	57.56	56.99
	1	**61.99**	65.35	64.98	64.42
	10	**129.34**	139.02	139.20	138.63

6 Conclusion and Future Work

This paper proposes a new parking space search problem, whose objective is to minimize the mean and standard deviation of parking space search time, and then introduced the mean standard deviation proximal policy optimization (MS-PPO) algorithm as a model-free RL-based solution. We benchmark the mean-std performance of MS-PPO with canonical parking space search algorithms in a range of parking lot networks, and perform MS-PPO's ablation study to dissect the functional contribution of each component. In the future, we are keen to scaling up MS-PPO to multiple vehicles for multi-vehicle parking space search, and also integrating MS-PPO with autonomous vehicles for reliable parking space search and autonomous parking.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 62576229).

References

Ata, K.; Che Soh, A.; Ishak, A.; and Jaafar, H. 2021. Guidance system based on Dijkstra-ant colony algorithm with binary search tree for indoor parking system. *Indonesian Journal of Electrical Engineering and Computer Science*, 24: 1173.

Bertsekas, D. P. 2018. *Abstract Dynamic Programming*. Athena Scientific.

Boulares, M.; Fehri, A.; and Jemni, M. 2024. UAV path planning algorithm based on Deep Q-Learning to search for a floating lost target in the ocean. *Robotics Auton. Syst.*, 179: 104730.

Chen, M. 2021. Urban Parking Scheme in Hangzhou Based on Reinforcement Learning. *IOP Conference Series: Earth and Environmental Science*, 638(1): 012002.

Chen, P.; Tong, R.; Yu, B.; and Wang, Y. 2020. Reliable shortest path finding in stochastic time-dependent road network with spatial-temporal link correlations: A case study from Beijing. *Expert Systems with Applications*, 147: 113192.

Dou, N.; Lian, Z.; and Guo, C. 2024. Parking Space Matching and Path Planning Based on Wolf Feeding Decision Algorithm in Large Underground Garage. In Li, J.; Zhang, B.; and Ying, Y., eds., *6GN for Future Wireless Networks*, 91–104. Cham: Springer Nature Switzerland. ISBN 978-3-031-53401-0.

Geng, Z.; and Sun, Y. 2023. An improved ant colony optimization for UAV minimum time search path planning. *Proceedings of the 2023 4th International Conference on Artificial Intelligence in Electronics Engineering*.

Guo, H.; Hou, X.; Cao, Z.; and Zhang, J. 2022. GP3: Gaussian Process Path Planning for Reliable Shortest Path in Transportation Networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(8): 11575–11590.

Guo, H.; Hou, X.; and Peng, Q. 2022. CTD: Cascaded Temporal Difference Learning for the Mean-Standard Deviation Shortest Path Problem. *IEEE Transactions on Intelligent Transportation Systems*, 23(8): 10868–10886.

Han, Z.; Sun, H.; Huang, J.; Xu, J.; Tang, Y.; and Liu, X. 2024. Path Planning Algorithms for Smart Parking: Review and Prospects. *World Electric Vehicle Journal*, 15(7).

Hao, J.; Wang, C.; Yang, M.; and Wang, B. 2020. Hybrid Genetic Algorithm Based Dispatch and Conflict-free Routing Method of AGV Systems in Unmanned Underground Parking Lots. In *2020 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 475–480.

Hassan, M. S.; Islam, N. N.; Fahim, A. M. A.; Turja, T. H.; and Chowdhury, S. 2020. Automated Parking System using Graph Algorithm. In *Proceedings of the International Conference on Computing Advancements*, 1–6. New York, NY, USA: Association for Computing Machinery. ISBN 9781450377782.

Hou, X.; Guo, H.; and Zhang, Y. 2020. Adaptive Reliable Shortest Path in Gaussian Process Regulated Environments. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6819–6825.

Jang, H.-C.; Huang, Y.-C.; and Chiu, H.-A. 2020. A Study on the Effectiveness of A2C and A3C Reinforcement Learning in Parking Space Search in Urban Areas Problem. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, 567–571.

Jang, H.-C.; and Lee, C.-Y. 2022. Study on Q-Learning and Deep Q-Learning in Urban Roadside Parking Space Search. In *2022 International Conference on Computational Science and Computational Intelligence (CSCI)*, 1248–1253.

Knuth, D. E. 1976. Big Omicron and big Omega and big Theta. *SIGACT News*, 8(2): 18–24.

Kocsány, L.; and Szádeczky-Kardoss, E. 2022. Application of mixed graph traversal optimization for the vehicle routing problem. In *European Control Conference (ECC)*, 2149–2154.

Li, L.; Shi, D.; Jin, S.; Yang, S.; Zhou, C.; Lian, Y.; and Liu, H. 2023. Exact and Heuristic Multi-Robot Dubins Coverage Path Planning for Known Environments. *Sensors*, 23(5).

Si, L.; Ma, D.; Wang, Z.; Shao, H.; and Li, X. 2025. Path planning for mobile robots based on improved sparrow search algorithm under various maps. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*.

Song, M.; and Cheng, L. 2022. A generalized Benders decomposition approach for the mean-standard deviation shortest path problem. *Transportation Letters*, 15: 823 – 833.

Sun, L.; Wei, Y.; Li, M.; Chen, C.; and Yang, M. 2022. Smart Parking Management and Planning Based on Chinese Postman Problem. *Open Journal of Social Sciences*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. MIT press.

Tiong, T.; Saad, I.; Teo, K. T. K.; and Bin Lago, H. 2022. Autonomous Valet Parking with Asynchronous Advantage Actor-Critic Proximal Policy Optimization. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, 0334–0340.

Wang, X.; Shi, H.; and Zhang, C. 2020. Path Planning for Intelligent Parking System Based on Improved Ant Colony Optimization. *IEEE Access*, 8: 65267–65273.

Xing, T.; and Zhou, X. 2011. Finding the most reliable path with and without link travel time correlation: A Lagrangian substitution based approach. *Transportation Research Part B: Methodological*, 45(10): 1660–1679.

Zhang, Y.; and Khani, A. 2019. An algorithm for reliable shortest path problem with travel time correlations. *Transportation Research Part B: Methodological*, 121: 92–113.

Zhang, Y.; Liu, Y.; Chen, Y.; and Yang, Z. 2025. ARE-QL: an enhanced Q-learning algorithm with optimized search for mobile robot path planning. *Physica Scripta*, 100.

Ádám, A.; Kocsány, L.; and Szádeczky-Kardoss, E. 2021. Using coverage path planning methods for car park exploration. *ACTA IMEKO*, 10: 15.