drones

MDPI

Article

Dynamic Evaluation of Learning Internalization Capability in Unmanned Ground Vehicles via Time Series Analysis

Zewei Dong [1,2,*], Jingxuan Yang [1], Guangzhen Su [3], Yaze Guo [2], Ming Lei [1], Xiaoqin Liu [2] and Yuchen Shi [1]

1 Department of Automation, Tsinghua University, Beijing 100084, China; yangjx20@mails.tsinghua.edu.cn (J.Y.); leim24@mails.tsinghua.edu.cn (M.L.); shiyuche21@mails.tsinghua.edu.cn (Y.S.)
2 Department of Avionics and Ordnance Engineering, Army Aviation Institute, Beijing 100123, China; guoyaze@hotmail.com (Y.G.); liuxiaoqin1121@163.com (X.L.)
3 Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China; gzsu@link.cuhk.edu.hk
* Correspondence: dzw22@mails.tsinghua.edu.cn

Highlights

What are the main findings?
- This study introduces a novel dualistic definition of learnability for Unmanned Ground Vehicles (UGVs), clearly distinguishing between Learning Internalization Capability (LIC) and Learning Generalization Capability (LGC), and focuses on dynamically quantifying LIC.
- A black-box evaluation framework based on time series analysis is proposed, which employs a sliding window-based slope-standard deviation collaborative analysis technique to objectively segment the learning process and extract five core metrics for comprehensive LIC characterization.

What are the implications of the main findings?
- The framework provides a standardized, quantitative tool for the intelligent selection of UGVs, the iterative optimization of algorithms, and the design of training strategies, moving beyond static performance snapshots.
- It demonstrates superior discriminative power in revealing the long-term learning potential of algorithms in complex scenarios, offering critical decision-support where traditional static evaluations would fail.

Abstract

Aiming to address the core issue that the current intelligence evaluation for Unmanned Ground Vehicles (UGVs) overly rely on static performance metrics and lack dynamic quantitative characterization of learning internalization capability (LIC), this study proposes a dynamic evaluation framework based on time series analysis. The framework begins by constructing a multidimensional test scenario parameter system and collecting externally observable performance sequence data. It then introduces a sliding window-based slope-standard deviation collaborative analysis technique to achieve unsupervised division of learning phases, from which five core evaluation metrics are extracted to comprehensively quantify the multidimensional dynamic characteristics of LIC in terms of efficiency, stability, and overall effectiveness. Simulation experiments were carried out using UGVs equipped with three types of path-planning algorithms in low-, medium-, and high-difficulty scenarios. Results demonstrate that the proposed algorithm can effectively distinguish multi-dimensional differences in LIC among different UGVs, exhibiting strong discriminative power and interpretability. This study provides a standardized evaluation

Check for updates

Academic Editor: Yushu Yu

Received: 4 December 2025
Revised: 29 December 2025
Accepted: 6 January 2026
Published: 8 January 2026

tool for UGV intelligent selection, algorithm iteration optimization, and training strategy design, and offering significant reference value for the evaluation of the learnability of autonomous driving systems.

Keywords: learning internalization capability (LIC); Unmanned Ground Vehicles (UGVs); dynamic evaluation; time series analysis; learnability

1. Introduction

Unmanned Ground Vehicles (UGVs), as a critical branch of intelligent equipment, are now widely deployed in various sectors such as logistics, urban inspection, environmental monitoring, and operations in hazardous environments [1–3]. In these unstructured settings, the task performance of UGVs no longer depends solely on hardware capabilities or the static suitability of pre-defined algorithms. Instead, it increasingly relies on the system's learnability [4–8], which is its capacity to continuously optimize decisions through ongoing learning and convert experiential knowledge into long-term performance gains [9–11]. Consequently, objectively quantifying the learnability of UGVs is not only a core metric for evaluating their intelligence but also a key enabler for advancing UGVs from specific scenario adaptation towards general task adaptation [12].

AI evaluation research has been conducted for many years, leading to the development of various technical frameworks [13–19]. Early mainstream evaluation methods [20], often remained at the qualitative level, using classification to distinguish intelligence levels. However, these approaches are inherently subjective and lack standardized quantitative criteria. Recent studies predominantly adopt a task-oriented evaluation paradigm [11], such as testing on specific benchmarks or competitive events. A key limitation of this paradigm is its task specificity. UGVs can achieve high scores via specialized training using reinforcement learning(RL), making the results highly context-limited. In response, capability-oriented evaluation models are gradually gaining attention. Ref. [21] proposed an 'Environmental Adaptability–Task Adaptability–Autonomy' framework, employing metrics like task completion rate and equipment wear to quantify the comprehensive effectiveness of UGVs. Ref. [22] systematized five categories of evaluation indicators within the STCER-H framework. Ref. [23] constructed an evaluation system covering multiple dimensions, including perception, decision-making, and behavioral strategies. Ref. [24] used instantaneous performance metrics, such as target recognition accuracy and decision-making response speed in dynamic scenes, as core indicators. Ref. [25] proposed a 'Robust Training–Multi-dimensional Quantification' approach (the RTCE framework) that was limited to 'safety performance.' However, these studies primarily focus on the statistical results of instantaneous performance [26]. Learnability has not been established as a core evaluation dimension [27]. Consequently, intelligence evaluation remains superficial, confined mainly to assessing functional implementation. While platforms for physical robotic testing, such as the remote multi-robot lab [28], are invaluable for final validation, a standardized methodology for quantitatively characterizing the learning process during development is still lacking. Therefore, research specifically targeting the learnability evaluation of UGVs is particularly necessary.

To address this inherent gap in existing evaluation systems, this paper defines learnability as a core evaluation dimension for UGVs. Learnability, however, is not monolithic. To evaluate it meaningfully, we propose a dualistic framework that distinguishes between two complementary yet distinct facets: Learning Internalization Capability (LIC) and Learning Generalization Capability (LGC). LIC refers to a system's ability to absorb and optimize experiential knowledge within a fixed task and environment, translating it into sustained, long-term performance gains. For example, consider a UGV performing repetitive sorting tasks in a static warehouse. Over multiple trials, a high-LIC UGV gradually internalizes experience to optimize routing strategies, yielding measurable, consistent improvements in completion time and success rate in the same static setting. This progression from initial performance to a stable, more efficient state epitomizes the consolidation of learning into durable capability—the core of LIC. In contrast, LGC denotes a system's ability to transfer and adapt learned knowledge to novel, unseen tasks or environments, transcending prior experience boundaries. It reflects the breadth and flexibility of learning.

While LIC and LGC together define a system's learnability, they differ fundamentally in evaluation objectives, methodologies, and application scenarios. This study focuses specifically on the dynamic evaluation of LIC, as it directly dictates a UGV's potential for rapid performance improvement under known operational conditions. Quantifying LIC requires analyzing the temporal evolution inherent in the performance sequence Y, a gap that our framework addresses. Though equally critical, LGC poses distinct evaluation challenges and is reserved for independent future research.

Previous methods for evaluating autonomous systems have included both system-level evaluations (e.g., expert qualitative [19,20], task-oriented [11] and capability-oriented [21,24]) and learnability analyses (e.g., learning curve fitting [29], RL convergence analysis [30], and adaptive performance metrics [31]), which typically focus on static performance or internal algorithm states. In contrast, we adopt an independent, reproducible third-party perspective and propose a fundamentally different, black-box dynamic evaluation framework based on time-series analysis. Our approach does not rely on any internal algorithm details; instead, it quantifies LIC solely through analysis of externally observable performance sequences. The main contributions are:

(i) Learnability is clearly defined for the first time through the dual dimensions of LIC and LGC, with a focus on quantifying the former. This provides a new theoretical perspective for evaluating the intelligence level of autonomous systems.

(ii) A multidimensional test scenario parameter system for dynamic evaluation is constructed. A sliding window-based slope-standard deviation collaborative analysis technique is proposed to achieve objective division of learning phases, overcoming the subjectivity of traditional experience-based threshold methods.

(iii) A set of core metrics is designed to evaluate LIC, enabling the dynamic characterization of learning efficiency, stability, and comprehensive effectiveness.

(iv) Simulation experiments are conducted to validate the effectiveness and discriminative power of the proposed method in comparing the LIC of different algorithms across scenarios. This provides scientific decision-making support for UGV algorithm selection and training strategy optimization.

Building on this conceptual foundation, we posit that the LIC of a UGV can be effectively and objectively quantified through a black-box analysis of the dynamic evolutionary patterns within its externally observable performance sequence Y. This hypothesis forms the cornerstone of the evaluation framework proposed in this study.

The remainder of this paper is organized as follows: Section 2 analyzes the core challenges in evaluating the LIC of UGVs. Section 3 details the proposed dynamic evaluation framework. Section 4 designs and conducts simulation experiments. Section 5 analyzes the experimental results to validate the method's effectiveness. Section 6 concludes the paper and outlines future research directions.

2. Problem Description

Although static performance evaluation systems are relatively well-established, quantifying the LIC of UGVs remains a core challenge that has not been fully explored. To precisely characterize this dynamic process and analyze its associated evaluation challenges, this section first establishes a formal mathematical model of the learning internalization process. This clarifies the fundamental distinction between dynamic and static evaluation, thereby systematically introducing the core problems faced in this evaluation.

2.1. Formal Modeling

The learning internalization process of an unmanned system is essentially a dynamic cycle of continuous interaction between its agent and the task environment, optimizing its behavioral policy based on historical experience. This process can be decomposed into four phases, as illustrated in Figure 1.

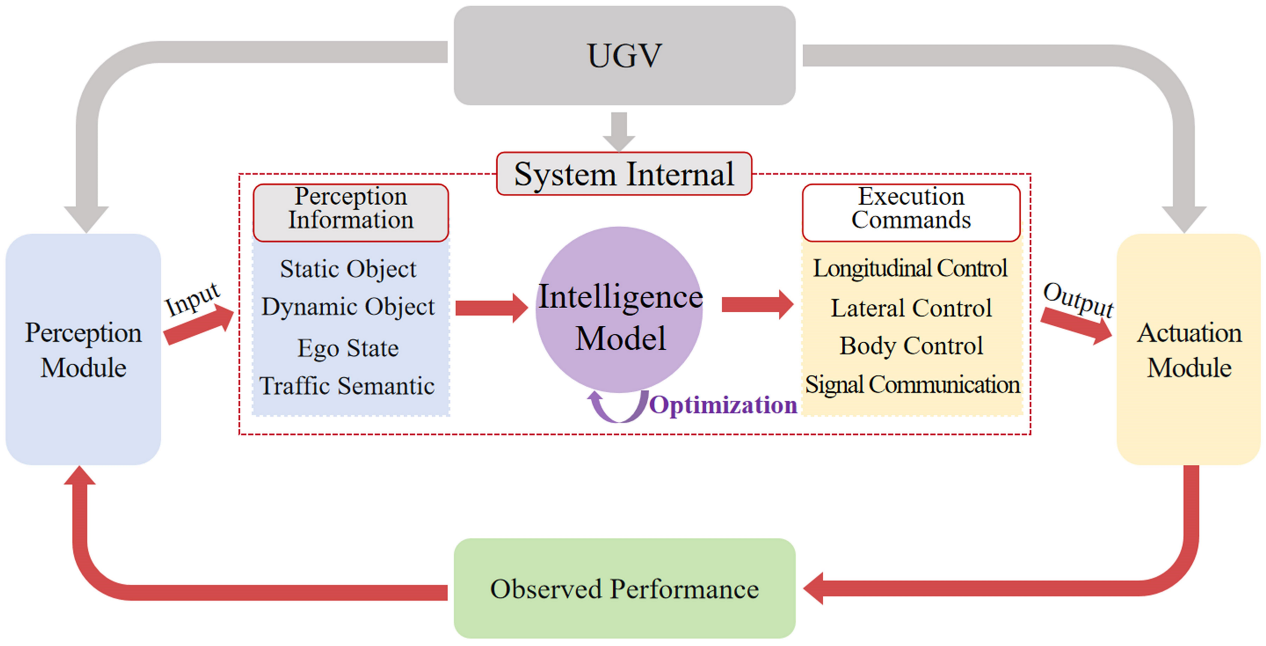


Figure 1. Schematic of the learning internalization process in a UGV.

(i) Experience Generation: In a specific test scenario $\mathcal{E}$, the system executes the task for the t-th time, generating an experience trajectory $\tau_t = (s_0, a_0, s_1, a_1, \ldots, s_T)$, where $s \in S$ represents the environmental state and $a \in A$ represents the system action. As indicated by the red arrow in Figure 1, each cycle produces one experience trajectory τ_t.

(ii) Policy Update: The system's internal learning mechanism utilizes the accumulated experience data $D_t = \{\tau_1, \tau_2, \ldots, \tau_t\}$ to update its decision-making policy, i.e., $\pi_{t+1} \leftarrow Learn(\pi_t, D_t)$. This process encapsulates the black-box operations internal to the system, such as parameter optimization and model adjustment, represented by the purple arrow in Figure 1.

(iii) Performance Manifestation: The updated policy π_{t+1} is executed in the environment. Its performance is quantified by one or more observable performance metrics y_{t+1}, i.e., $y_{t+1} = P(\tau_{t+1} | \pi_{t+1}, \mathcal{E})$. Here, P is a performance mapping function that translates the trajectory into a scalar performance value (e.g., task success rate, efficiency score), as shown in the yellow area of Figure 1.

(iv) Ability Internalization: As the iteration count t increases, the system's performance sequence $Y = \{y_1, y_2, \ldots, y_N\}$ exhibits certain evolutionary patterns. The strength of LIC is precisely reflected in the dynamic statistical characteristics and convergence properties of this performance sequence Y. As indicated by the red dashed arrow in Figure 1, LIC gradually forms as this cyclic process evolves.

In contrast, traditional static evaluation methods treat the system as a static mapping:

$$y = P(\tau \mid \pi, \mathcal{E}) \tag{1}$$

where the evaluation function H typically performs cross-sectional statistics (e.g., mean, maximum) on the outputs of multiple independent tests, expressed as $Static_{Score} = H(y_1, y_2, \ldots, y_N)$. This method focuses solely on the system's performance at a specific point or the statistical outcome of multiple tests, completely ignoring the temporal correlation and dynamic evolution patterns inherent in the performance output sequence $\{y_t\}$. Consequently, it cannot determine whether the system's performance improves with accumulated experience.

In opposition, the dynamic evaluation central to this paper focuses on quantifying the system's ability to improve performance, as revealed by the sequence $Y = \{y_1, y_2, \ldots, y_N\}$. Therefore, we define the LIC evaluation problem as follows: construct an evaluation function F that takes the observable performance sequence Y from a fixed scenario $\mathcal{E}$ as input. For LIC to be deemed present and quantifiable, Y must satisfy two core statistical conditions:

(i) Significant Positive Trend: The sequence Y must exhibit a statistically significant positive trend.

(ii) Stationary Convergence: The sequence must eventually reach a stationary convergence phase, where the fluctuations are bounded and exhibit no systematic drift.

The strength of LIC is then quantified by F through metrics that capture the efficiency of the ascent and the stability of the convergence. Crucially, this definition is falsifiable: if Y resembles a random walk, shows no significant trend, or degrades, then by definition, LIC is considered absent or negligible for the tested system in scenario $\mathcal{E}$. Thus:

$$\text{LIC} = F(Y) = F(\{y_1, y_2, \ldots, y_N\}) \tag{2}$$

where F operates under the premise that the above conditions hold.

2.2. Core Challenges

Based on the formal definition above, current evaluation systems face three core challenges when addressing the dynamic attribute of LIC [32]:

(i) Lack of Quantification for Dynamic Evolution. Existing evaluations predominantly focus on cross-sectional statistics of system performance, such as the final performance y_N or the average performance $\bar{y}$. These metrics fail to capture the temporal correlation and dynamic trends within the performance sequence Y. A system with high LIC should exhibit a rapid initial climb in its Y sequence, followed by stable convergence. However, the primary challenge is how to objectively and unsupervisedly identify the learning phases from a potentially noisy Y sequence and quantify the characteristics of each phase. This demands that the evaluation function F can effectively extract the dynamic patterns of the sequence, a requirement that static aggregation functions like $Static_{Score} = H(y_1, y_2, \ldots, y_N)$ cannot be fulfilled.

(ii) Universality Requirement for Black-Box Evaluation. As shown in the formula $\pi_{t+1} \leftarrow Learn(\pi_t, D_t)$, the system's internal learning mechanism $Learn(\cdot)$ is typically invisible to a third-party evaluator. Therefore, the evaluation function F must strictly adhere to a black-box paradigm [33]. It should make inferences based solely on the externally observable Y, without any assumptions about the internal principles of $Learn(\cdot)$. That is,

$LIC = F(Y)$, and F must be independent of the specific implementation of $Learn(\cdot)$. This necessitates that F possesses universality across heterogeneous algorithm architectures.

(iii) Gap in the Evaluation Metric System. The dynamic characteristics of the performance sequence Y are multidimensional. Currently, there is a lack of a systematic indicator vector $\vec{I} = (I_{\text{AI}}, I_{\text{CV}}, I_{\text{MP}}, I_{\text{ISR}}, I_{\text{AUC}})$ to collaboratively characterize features such as learning efficiency, convergence stability and overall learning effectiveness and to aggregate $\vec{I}$ into an overall evaluation of LIC $(\text{LIC} = \Phi(\vec{I}))$.

2.3. Critical Comparison with Existing Evaluation Methods

The challenges discussed in Section 2.2 reveal the inadequacy of existing methods for evaluating LIC. To clarify the position of our solution, we compare three methods: Expert Qualitative Evaluation, Task-Oriented or Capability-oriented Evaluation, and our Dynamic, Black-box LIC Evaluation. The comparison across key dimensions is summarized in Table 1, which highlights how our framework addresses the identified gaps.

Table 1. Comparison of evaluation methods for UGVs.

Dimension	Expert Qualitative Evaluation	Task-Oriented or Capability-Oriented Evaluation	Dynamic, Black-Box LIC Evaluation
Evaluation Object	Functional features and performance parameters.	Task or capability success metrics.	Temporal evolution of the performance sequence.
Core Methodology	Rating against defined autonomy levels.	Optimization and testing against fixed tasks or capabilities.	Time-series analysis of learning phases.
Primary Output	An intelligence or autonomy level.	A ranking or score.	Multi-dimensional metrics.
Capability for Assessing LIC	Fails to capture dynamic properties.	Limited, but fails to capture performance trends over time.	Directly quantifies learning efficiency, stability and effectiveness.
Algorithm Dependency	Dependent on known performance parameters.	Often encourages algorithm-specific optimization.	Independent (Black-box).

To further elucidate the methodological distinction of our framework, we provide a focused comparison with three related technical concepts: learning curve fitting, RL convergence analysis, and adaptive performance metrics. While these approaches also deal with learning or performance changes, their objectives and mechanisms differ fundamentally from our LIC evaluation.

(i) Learning Curve Fitting fits a global trend to the entire performance sequence to extract a single learning rate parameter. In contrast, our method performs non-parametric segmentation of the sequence into local phases (Ascent, Convergence) and extracts a multi-dimensional feature vector to characterize phase-specific dynamics.

(ii) RL Convergence Analysis is a white-box technique that examines the stability of internal algorithm states. Our framework, however, is a black-box tool that assesses convergence solely through the stability of the external performance output, using metrics.

(iii) Adaptive Performance Metrics measure a system's robustness or speed in responding to environmental changes. Our LIC evaluation, in contrast, quantifies a system's ability to optimize and internalize performance through experience within a fixed, unchanging scenario.

In summary, the key to this research lies in constructing an evaluation framework F that can automatically and robustly extract the dynamic evolutionary features from the performance sequence Y, and output a set of standardized, interpretable quantitative indicator vectors $\vec{I}$ and a final evaluation score LIC. The following sections will focus on this core problem, detailing the method design and experimental validation.

3. Methodology

To address the challenges in evaluating LIC outlined in Section 2, this study draws on the learning curve theory [34] for describing the dynamic evolution of learning processes. We propose a dynamic evaluation method based on time series analysis. This method treats the learning internalization process of an intelligent system as a dynamic system evolving over time. By analyzing its externally observable time-series performance data, it establishes a black-box evaluation framework that is independent of the system's internal implementation.

3.1. Framework

This study constructs a systematic evaluation framework, as shown in Figure 2. This framework strictly adheres to the black-box evaluation paradigm, relying solely on the system's externally observable sequential behavior data, thereby ensuring the objectivity of the evaluation process and the universality of the results.

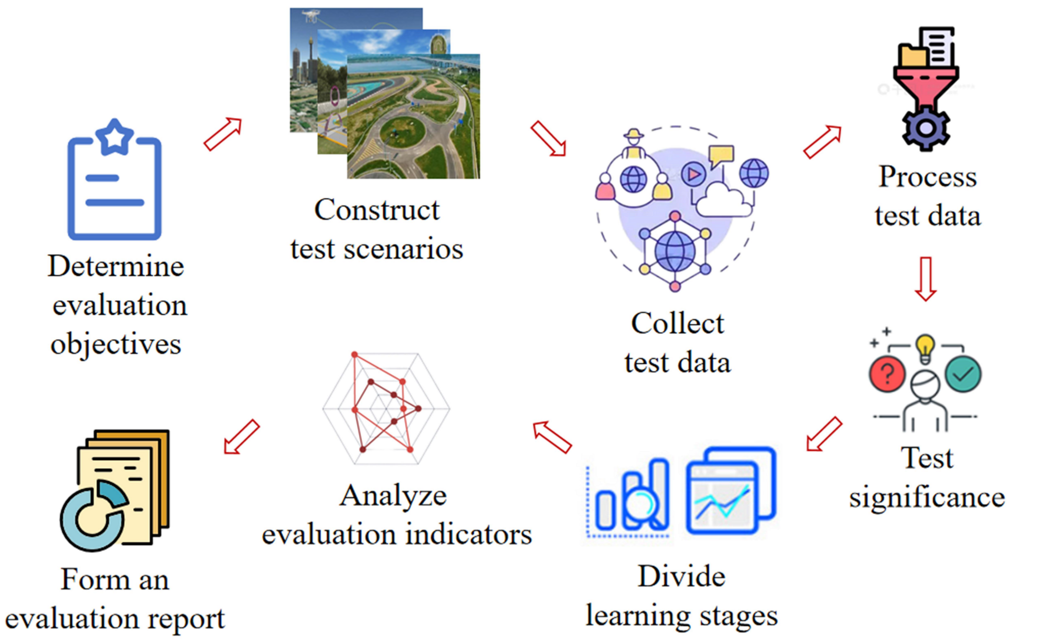


Figure 2. Framework for evaluating LIC.

(i) Test Scenario Modeling: A multidimensional parameter space is constructed to provide a controlled environment for stimulating and observing learning behaviors.

(ii) Time Series Data Collection and Preprocessing: Serialized tests are conducted under predefined scenarios to collect the raw performance sequence $Y = \{y_1, y_2, \ldots, y_N\}$, followed by data cleaning and standardization.

(iii) Dynamic Division of Learning phases: Based on a sliding window-based slope-standard deviation collaborative analysis technique, the time-series performance sequence is analyzed to identify and objectively partition it into Ascent Phase and Convergence Phase.

(iv) Quantitative Evaluation of Learning Characteristics: Multi-dimensional characteristic metrics are extracted from the partitioned learning phases to form an evaluation vector $\vec{I}$, achieving a comprehensive quantitative characterization of the LIC.

3.2. Test Scenario Modeling

To stimulate and observe learning behaviors, we first construct a multidimensional parameter space for precise environment modeling:

$$\boldsymbol{\Theta} = (\Theta_e, \Theta_o, \Theta_s) \tag{3}$$

where Θ_e represents environmental parameters, Θ_o represents task object parameters, and Θ_s represents the system's own parameters.

(1) Environmental Parameters (Θ_e)

These describe the external physical conditions of the system's operation, including illumination intensity, weather conditions, and noise level, which affect the input quality of the perception module.

(2) Task Object Parameters (Θ_o)

These define the characteristics of the target entities for task execution, described using feature vectors or parameterized trajectories. For example, a regular object can be represented as $\vec{v} = (l, w, h)$, and a dynamic obstacle trajectory as $p(t) = \left(x_0 + v_x t + \frac{1}{2}a_x t^2, y_0 + v_y t + \frac{1}{2}a_y t^2\right)$.

(3) System's Own Parameters (Θ_s)

These reflect the system's hardware configuration and algorithmic characteristics, including computational capability, sensor accuracy, exploration parameters, and learning rate.

A unified mathematical model for test scenarios is established:

$$\Gamma = \{(p_1, p_2, \ldots, p_k) | p_i \in D_i, i = 1, 2, \ldots, k\} \tag{4}$$

where p_i is the i-th variable parameter and D_i is its domain. Three types of test scenarios are generated through parameter configuration:

Specific Configuration Scenario: All parameters are fixed, $\Gamma = \{(c_1, c_2, \ldots, c_k)\}$.

Single-Variable Analysis Scenario: A single parameter varies, $\Gamma = \{(p_1, c_2, \ldots, c_k) | p_1 \in D_1\}$.

Multi-Variable Coupling Scenario: Multiple parameters change simultaneously, $\Gamma = \{(p_1, p_2, \ldots, p_k) | p_i \in D_i\}$.

3.3. Time Series Data Collection and Preprocessing

3.3.1. Time Series Data Collection

For each test scenario Γ, we conduct N independent trials. We record the performance indicator value y_t for every successful task completion, where t denotes the sequence number of successes. This establishes a quantitative mapping:

$$y_t = f(\Gamma, t), for\ t = 1, 2, \ldots, N_{success} \tag{5}$$

where N_{success} is the total number of successful tasks. The performance indicator Y can be any extrinsic metric central to characterizing intelligence levels. This framework is general-purpose, and the choice of Y is task-dependent. For instance, it can be task success rate, completion time, or energy consumption for efficiency-related tasks; detection accuracy for perception-related tasks; tracking error for control-related tasks; or a comprehensive intelligence score generated by expert weighting for overall assessment. The subsequent analysis operates on the sequence Y regardless of its specific semantic meaning.

If a task fails (according to preset criteria such as timeout, target drop, or critical action interruption), the failure event is recorded, and the total number of failures is denoted as $N_{failure}$. The failure rate will be used to quantify test scenario complexity and determine the significance of the learning effect.

The total number of tests N is determined by either reaching a preset fixed upper limit or by terminating when the performance fluctuation amplitude remains below a threshold ϵ for M consecutive tests, in which case N corresponds to the actual number of tests performed.

3.3.2. Data Preprocessing

(1) Outlier Filtering: A sliding window statistical method is used to identify and filter outliers. If the indicator value y_t from a test fall outside the range of 3 standard deviations from the mean within the sliding window, it is removed:

$$|y_t - \mu_w| > 3\sigma_w \tag{6}$$

The remaining N' data points form a new sequence for subsequent analysis.

(2) Data Normalization: The Min–Max linear normalization method [35] is used to map the processed data to the interval [0, 1].

$$\tilde{y}_t = \frac{y_t - Y_{\min}}{Y_{\max} - Y_{\min}} \quad \text{for} \quad t = 1, 2, \ldots, N' \tag{7}$$

where $Y_{\min}$ and $Y_{\max}$ denote the minimum and maximum values of the processed performance sequence Y, respectively.

3.4. Evaluation Model

To fulfill the requirements for a robust, black-box evaluation of LIC as outlined in Section 2.2, our evaluation model follows a structured workflow. First, a significance test (Section 3.4.1) determines whether a meaningful learning effect exists. If confirmed, the phase division algorithm (Section 3.4.2) objectively identifies the Ascent and Convergence phases within the performance sequence Y. The key hyperparameters of this algorithm (Section 3.4.3) are systematically analyzed and tuned to ensure robustness and generalizability. Finally, the evaluation metrics (Section 3.4.4) are specifically selected to translate the characteristics of these phases into a comprehensive assessment vector, I. Consequently, metrics tailored for analyzing stationary time series, such as autocorrelation, were excluded from our framework, as they are unsuitable for capturing the non-stationary and continuously evolving nature of the learning process inherent to LIC assessment.

3.4.1. Significance Test

Let the task success rate and the mean intelligence evaluation value in the first m tests be S_{initial} and $\bar{y}_{\text{initial}}$, respectively, and in the last m tests be S_{final} and $\bar{y}_{\text{final}}$, respectively (where N is the total number of tests, and $m \leq \frac{N}{2}$). The learning growth rate G is then:

$$G = \frac{\bar{y}_{\text{final}} - \bar{y}_{\text{initial}}}{\bar{y}_{\text{initial}}} \tag{8}$$

Determining the significance of the learning effect requires simultaneously satisfying the following two conditions:

Criterion 1: $S_{\text{final}} - S_{\text{initial}} \geq \theta_S$, where θ_S is a preset task success rate threshold set according to the test scenario difficulty.

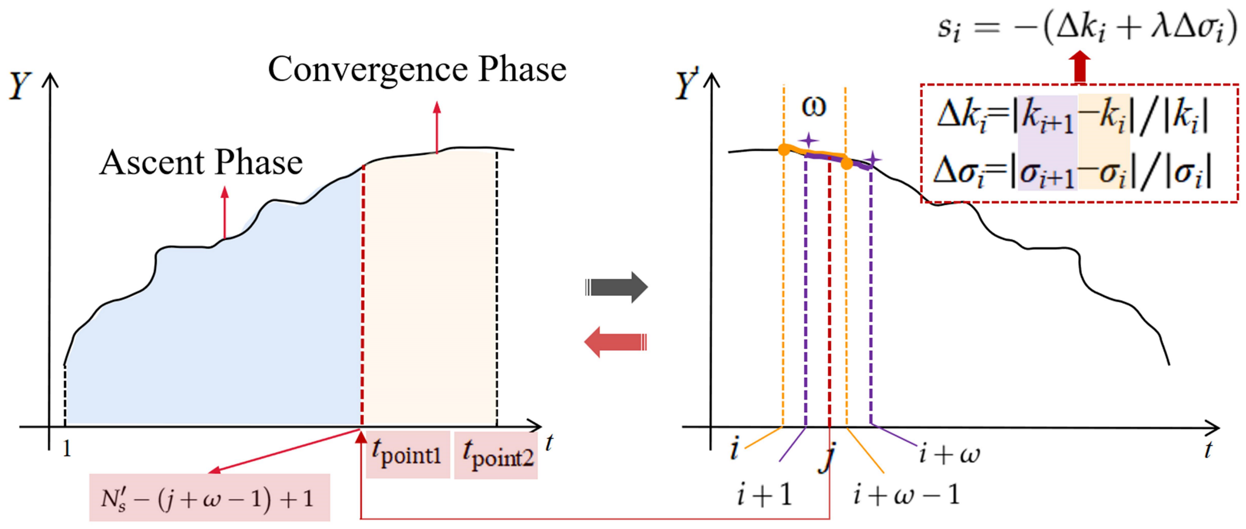
Criterion 2: $G \geq \theta_G$ (where θ_G is a preset minimum effective growth rate threshold), and the null hypothesis $H_0 : \bar{y}_{\text{initial}} = \bar{y}_{\text{final}}$ is rejected using a two-independent-samples t-test (significance level α = 0.05).

The learning effect is deemed significant only if both criteria are met; otherwise, it is considered not significant.

3.4.2. Learning Phase Division

Based on the evolutionary characteristics of the performance sequence, the learning process can be divided into two typical phases: the Ascent Phase, corresponding to the algorithm exploration phase with rapid performance improvement but significant fluctuations; and the Convergence Phase, corresponding to the mature optimization phase where performance stabilizes and fluctuations diminish [36–39]. Traditional time series analysis struggles to accurately distinguish random fluctuations from substantive progress during the volatile Ascent Phase, easily leading to misidentification of phases.

To address this, this study adopts a reverse-order analysis strategy, analyzing backwards from the end of the sequence (Convergence Phase) towards the start (Ascent Phase), as shown in Figure 3. This method starts from the less volatile region, allowing clearer identification of performance state transition boundaries and effectively avoiding interference from the severe fluctuations in the Ascent Phase. The specific steps are as follows (see Figure 4):



$s_i = -(\Delta k_i + \lambda \Delta\sigma_i)$
Convergence Phase
Ascent Phase
Y
Y'
ω
$\Delta k_i = |k_{i+1} - k_i| / |k_i|$
$\Delta\sigma_i = |\sigma_{i+1} - \sigma_i| / |\sigma_i|$
1
t_{point1}
t_{point2}
t
$N_s' - (j + \omega - 1) + 1$
i
j
$i + \omega$
$i + 1$
$i + \omega - 1$
t

Figure 3. Schematic illustration of the learning phase division algorithm.

(1) Data Preparation: The learning curve consists of performance indicator values for all successful task sequences, denoted as $Y = \left\{y_1, y_2, \ldots, y_{N_s'}\right\}$, where N_s' is the total number of successful tasks.

(2) Reverse Order Processing: The original learning curve sequence Y is reversed to obtain a new sequence $Y' = \left\{y'_{N_s'}, y'_{N_s'-1}, \ldots, y'_1\right\}$. Subsequent analysis is performed on Y'.

(3) Sliding Window Definition: A sliding window of fixed size ω is defined. The selection of ω requires a trade-off between sensitivity to local changes and robustness to noise; it is typically set to 5–10% of the total number of tests.

(4) Intra-Window Statistic Calculation: For each valid starting position i in the reversed sequence Y', we extract a subsequence Y_i' of ω consecutive points. This starting point i must satisfy $i \leq N_s' - \omega + 1$.

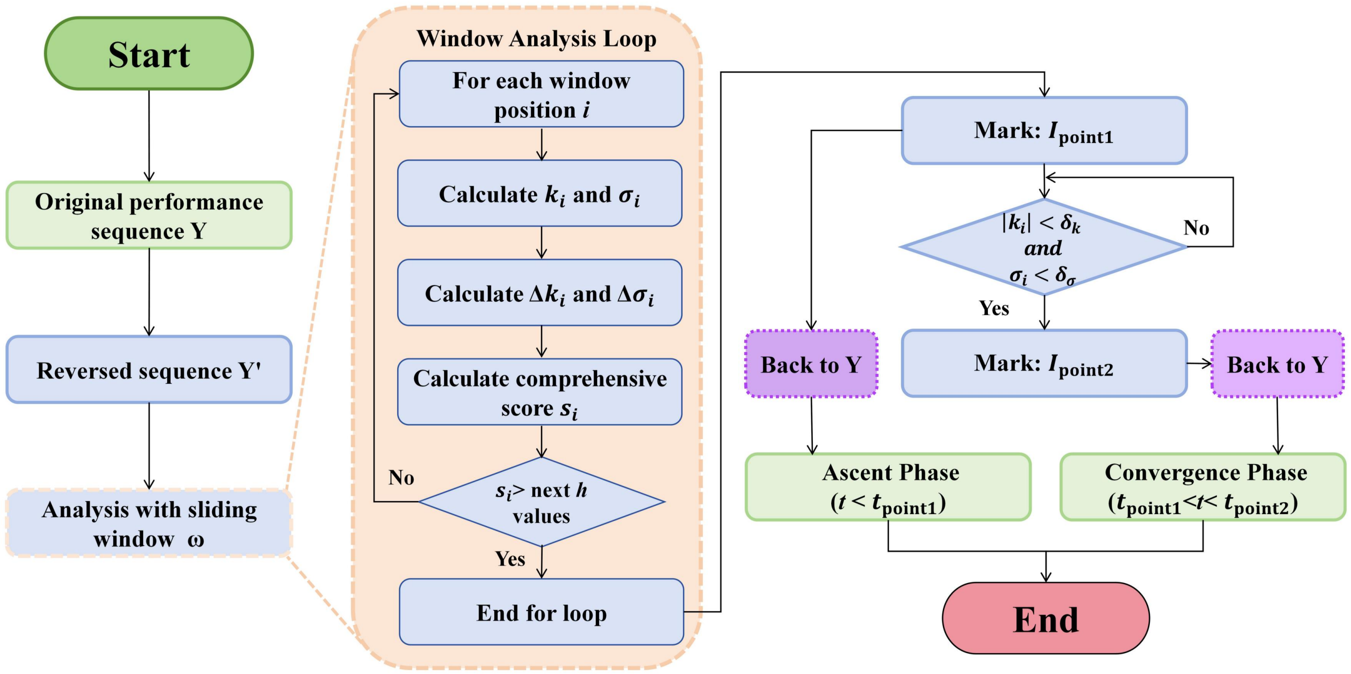

Start
Original performance sequence Y
Reversed sequence Y'
Analysis with sliding window ω
Window Analysis Loop
For each window position i
Calculate k_i and σ_i
Calculate Δk_i and $\Delta\sigma_i$
Calculate comprehensive score s_i
s_i> next h values
No
Yes
End for loop
Mark: I_{point1}
$|k_i| < \delta_k$ and $\sigma_i < \delta_\sigma$
No
Yes
Back to Y
Mark: I_{point2}
Back to Y
Ascent Phase ($t < t_{point1}$)
Convergence Phase ($t_{point1}<t< t_{point2}$)
End

Figure 4. Flowchart of the learning phase division process.

Slope Calculation: Perform linear regression on the subsequence Y_i' to calculate its slope k_i. This slope quantifies the average trend of the performance indicator within the window (in the reversed sequence, a negative slope typically indicates performance improvement in the original sequence).

Standard Deviation Calculation: Calculate the standard deviation σ_i of the subsequence Y_i'. This standard deviation quantifies the fluctuation of the performance indicator around its mean within the window.

(5) Rate of Change Calculation: For adjacent windows i and $i+1$, calculate the slope change rate Δk_i and the standard deviation change rate $\Delta\sigma_i$:

$$\Delta k_i = \frac{|k_{i+1} - k_i|}{|k_i|},\ \Delta\sigma_i = \frac{|\sigma_{i+1} - \sigma_i|}{|\sigma_i|} \tag{9}$$

where $i = 1, 2, ..., N_s' - \omega + 1$.

(6) Comprehensive Score Calculation: Combine the slope change rate and standard deviation change rate to calculate a comprehensive score:

$$s_i = -(\Delta k_i + \lambda\Delta\sigma_i) \tag{10}$$

The negative sign indicates the desire to find positions where the rate of change decreases (a larger score indicates a gentler change). λ is a tuning coefficient that balances the relative importance of the slope change rate and the standard deviation change rate. This study considers them equally important, setting $\lambda = 1$.

(7) Transition Point Detection: Transition point between Ascent and Convergence Phases: Find the position I_{point1} in the score sequence $\left\{s_1, s_2, \ldots, s_{N_s'-\omega+1}\right\}$ where its value is greater than the subsequent h consecutive values (typically an empirical value $h = 3$ is used).

End point of Convergence Phase: Within the Convergence Phase, find the first position I_{point2} that satisfies the following condition:

$$|k_i| < \delta_k \text{ and } \sigma_i < \delta_\sigma$$

where δ_k and δ_σ are preset thresholds for slope and standard deviation, indicating that the performance has sufficiently stabilized.

(8) Phase Division: Map I_{point1} and I_{point2} back to their corresponding positions in the original sequence Y. The transition point between the Ascent and Convergence Phases is $t_{\text{point1}} = N_s' - (I_{\text{point1}} + \omega - 1) + 1$, and the end point of the Convergence Phase is $t_{\text{point2}} = N_s' - (I_{\text{point2}} + \omega - 1) + 1$. In the original sequence, $t \leq t_{\text{point1}}$ is the Ascent Phase, and $t_{\text{point1}} < t \leq t_{\text{point2}}$ is the Convergence Phase.

The proposed phase division algorithm is specifically designed for the non-stationary, evolving nature of learning curves. It differs in objective from classical statistical change-point detection methods, such as PELT [40] or CUSUM [41]. Those methods are optimized to detect abrupt shifts in the parameters (e.g., mean, variance) of otherwise stationary processes. In contrast, our method targets the transition in the evolutionary pattern itself—from a phase of volatile improvement (non-stationary trend) to one of stable convergence. By tracking the simultaneous stabilization of k_i and σ_i, our approach provides a more direct and interpretable segmentation for learning process analysis.

The detailed justification for the selection of the hyperparameters (ω, λ, h, δ_k, δ_σ) used in this algorithm, along with a systematic sensitivity analysis, is provided in Section 3.4.3.

3.4.3. Hyperparameter Sensitivity Analysis and Tuning

To ensure robustness and generalizability, the key hyperparameters were selected as follows:

(i) The sliding window size ω was set to 20 (5% of the data length), which balanced trend sensitivity and noise resilience. A general guideline is 5–10% of the sequence length.

(ii) The balance factor λ was set to 1 to weight trend change (Δk_i) and stability change $(\Delta \sigma_i)$ equally, reflecting the dual focus of LIC on efficiency and stability. Evaluators may adjust λ to emphasize speed ($\lambda < 1$) or stability ($\lambda > 1$).

(iii) The minimum window count h was set to 3 to reliably detect phase transitions in our smoothed data. For noisier sequences, increasing h to 5 is recommended.

(iv) For the convergence thresholds δ_k and δ_σ, adaptive relative thresholds were adopted: δ_k was set to 5% of the maximum absolute slope in the Ascent Phase, and δ_σ to 5% of the global standard deviation of the normalized sequence. These percentages can be raised for noisier data to avoid premature convergence.

Overall, the algorithm shows strong robustness within these ranges, and the provided values offer practical starting points for new datasets.

3.4.4. Evaluation Metrics

To systematically characterize the multi-dimensional dynamic characteristics of LIC, this paper constructs an evaluation vector $\vec{I} = (I_{\text{AI}}, I_{\text{CV}}, I_{\text{MP}}, I_{\text{ISR}}, I_{\text{AUC}})$ comprising five dimensions. Specifically, I_{AI} and I_{ISR} are designed to evaluate learning efficiency and improvement magnitude, while I_{CV} and I_{MP} characterize convergence stability and performance level, respectively. I_{AUC} serves as a comprehensive metric of overall learning effectiveness. This enables both horizontal comparison of multiple algorithms within the same scenario and adaptability analysis of a single algorithm across multiple scenarios. The specific metric definitions are as follows:

(1) Average Increment (I_{AI}): This metric quantifies the learning speed during the Ascent Phase. It is defined as the ratio of the total performance gain within the phase to the number of training iterations, where a larger value indicates faster learning. I_{AI} is calculated as:

$$I_{\text{AI}} = \frac{y_{\max} - y_{\min}}{|Y_p|} \tag{11}$$

where Y_p is the performance subsequence during the Ascent Phase, and $y_{\max}$ and $y_{\min}$ are the maximum and minimum performance values within Y_p, respectively.

(2) Stability Degree (I_{CV}): This metric quantifies the fluctuation amplitude of the algorithm's output performance during the Convergence Phase. It is represented by the coefficient of variation in the performance values within this phase, where a smaller value indicates more stable performance. I_{CV} is calculated as:

$$I_{\text{CV}} = \frac{\sigma_c}{\bar{y}_c} \tag{12}$$

where $\bar{y}_c$ and σ_c are the mean and standard deviation of the performance values within the Convergence Phase, respectively.

(3) Mean Performance (I_{MP}): This metric characterizes the stable output level achieved during the Convergence Phase, representing the algorithm's average performance in its mature phase. It is calculated as the arithmetic mean of all performance values within this phase, where a larger value indicates superior stable performance. I_{MP} is defined as:

$$I_{\text{MP}} = \frac{1}{n}\sum_{t=1}^{n} y_t \tag{13}$$

where n is the number of performance observations in the Convergence Phase, and y_t denotes the performance value at the t-th observation.

(4) Increase in Success Rate (I_{ISR}): This global metric quantifies the overall improvement in task success rate throughout the learning process. It is defined as the difference in success rates between the first m and last m trials, where a larger value indicates more substantial improvement. I_{ISR} is calculated as:

$$I_{\text{ISR}} = S_{\text{final}} - S_{\text{initial}} \tag{14}$$

where S_{initial} and S_{final} denote the success rates during the initial and final phases of the learning process, respectively.

(5) Area Under the Curve (I_{AUC}): This global metric reflects the overall efficiency and cumulative performance of the learning process. It is calculated as the area under the learning curve, with a larger value indicating better cumulative learning effects and higher learning efficiency. Using the trapezoidal rule for numerical integration, I_{AUC} is computed as:

$$I_{\text{AUC}} = \sum_{t=1}^{N'-1} \frac{y_t + y_{t+1}}{2} \tag{15}$$

where N' is the total number of successful trials, and y_t represents the performance value at the t-th successful trial.

4. Numerical Experiment

This experiment systematically evaluates the LIC of path-planning algorithms using three Unmanned Ground Vehicles (UGVs) as test subjects. To ensure comparability, the three vehicles differ only in their path-planning algorithms, with all other configurations being identical. Considering the high costs, low efficiency, and difficulty in replicating scenarios associated with physical vehicle tests, a simulation-based approach was adopted. All tests were performed on a platform with an Intel(R) Core(TM) i7-9750H CPU(Intel Corporation, Santa Clara, CA, USA) @ 2.60 GHz.

4.1. Test Scenario Construction

To create a representative and moderately complex testing environment, a two-dimensional simulation scenario was designed, as shown in Figure 5. This scenario incorporates static and dynamic obstacles, narrow passages, and unstructured path elements. Interferences from natural factors (e.g., lighting, weather) and road condition variations were excluded to ensure result reproducibility [42,43]. This methodology is well-supported by prior work in virtual robotic testing, such as the high-fidelity simulation environment for autonomous vehicle testing [44], which demonstrates the efficacy of virtual platforms for controlled and reproducible evaluation. The specific parameter settings are as follows:

(i) Terrain Parameters: The area was set as a 20 m $\times$ 20 m rectangle. The ground friction coefficient was set to 0.8, and the influence of terrain relief on vehicle motion was neglected.

(ii) Obstacle Configuration: Three static obstacles were placed, each with a radius of $*r*(m)$, located at coordinates (15, 15), (8, 3), and (5, 5), respectively. One dynamic obstacle was set to move reciprocally between (5, 10) and (15, 10) along the x-axis at a speed of 1 m/s. Note that the dynamic obstacle model is intentionally simplistic to establish a clear, reproducible baseline for initially validating the proposed evaluation framework. The framework itself is agnostic to environmental complexity and can be applied to scenarios with more sophisticated dynamics.

(iii) UGV Parameters: A differential drive model was used. The vehicle dimensions were 1.0 m $\times$ 0.5 m, with a maximum acceleration of ± 0.5 m/s^2 and a maximum steering angle of $\pm 30°$. The minimum safe distance was set to 0.5 m.

(iv) Task Objective: The UGV starts from the origin (0, 0) and must safely navigate around all obstacles to reach the target point (20, 20). The optimization objectives are to minimize both travel time and path length.

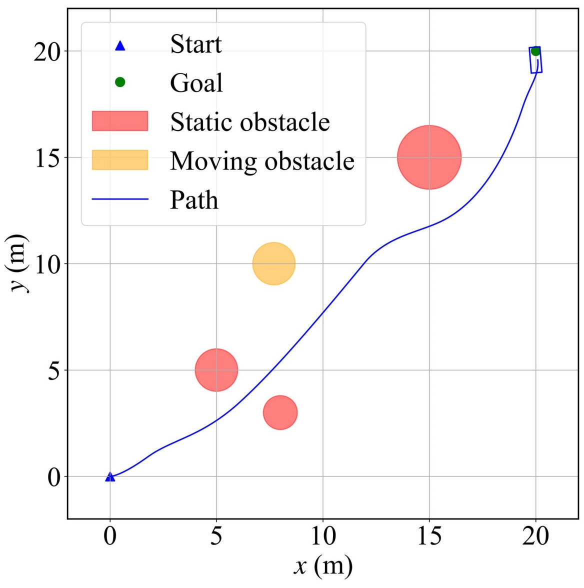


Figure 5. Test scenario.

4.2. Experimental Strategy

The obstacle radius was chosen as the task object variable to define three test scenarios: low difficulty ($\Gamma_1, R = 0.8$ m), medium difficulty ($\Gamma_2, R = 1.6$ m), and high difficulty ($\Gamma_3, R = 2.0$ m). In each scenario, 500 independent repeated trials were conducted for each algorithm. During these trials, a total of eight intelligence evaluation metrics across three categories—safety, efficiency, and stability—were fully recorded. The comprehensive intelligence score for each test was calculated using the method described in [45].

To overcome the challenge of directly quantifying learning performance in physical tests, three types of path-planning algorithms with distinct performance characteristics were selected as test objects to validate the evaluation method's effectiveness.

(i) Rule-Driven Algorithm: This method relies on a manually designed evaluation function for online optimization within the velocity space. It is characterized by strong real-time response capabilities and obstacle avoidance robustness, represented here by the Dynamic Window Approach (DWA) [46].

(ii) Probability Sampling Search-Based Algorithm: As a Monte Carlo sampling-based search method, this algorithm is suitable for path finding in high-dimensional state spaces, represented here by the Rapidly exploring Random Tree (RRT) [47].

(iii) Deep RL-Based, Offline Data-Driven Algorithm: This method optimizes policy by maximizing cumulative reward and policy entropy, capable of learning end-to-end planning strategies that combine high performance with good generalization capability, represented here by the Soft Actor–Critic (SAC) [48]. For the SAC implementation, we used the default hyperparameters from Ray RLlib 1.11.0. The learning rate for both the policy and value networks was set to 3×10^{-4}, with a discount factor $\gamma = 0.99$. Target networks were softly updated with $\tau = 0.005$. We employed a replay buffer of size 500,000, twin Q-networks, and automatic target entropy tuning. The agent was updated at every timestep using a batch size of 256. Both the policy and Q-network architectures consisted of two fully connected hidden layers with 256 units each and ReLU activation functions.

This selection of algorithms is deliberate, forming a capability spectrum for LIC evaluation: DWA serves as a non-learning, rule-driven baseline; RRT represents stochastic exploration without systematic policy improvement; and SAC is the learning-capable agent whose internalization process is the primary target of our quantitative assessment.

5. Results

5.1. Low- and Medium-Difficulty Test Scenarios

(1) Qualitative Analysis

Path trajectories are direct external manifestations of UGV decision-making behavior. To gain a preliminary qualitative understanding of their LIC, this section first compares and analyzes the path trajectories of each UGV in Γ_1 and Γ_2, as shown in Figure 6.

Figure 6 reveals distinct behavioral patterns:

(i) UGV-1's trajectory lines are highly coincident and almost indistinguishable. This indicates that as a local reactive algorithm, its decisions rely entirely on instantaneous sensor information. It lacks the ability to accumulate and utilize historical experience.

(ii) In contrast, UGV-2 shows significant trajectory divergence and randomness. This reflects the extensive exploration behavior of its sampling-based planner. Although random sampling can occasionally yield shorter paths, these sporadic improvements are stochastic and non-directed. The algorithm does not construct internal models or transfer knowledge from past successes, relying instead on random exploration rather than systematic self-improvement. Thus, its performance variations stem primarily from its random sampling nature rather than from systematic, experience-driven policy improvement.

(iii) UGV-3 clearly demonstrates a transition from exploration to optimization. Early trajectories are dispersed, elongated, and collision-prone, whereas later ones converge rapidly to a limited set of short and smooth paths. This marked improvement in trajectory quality provides strong qualitative evidence of robust learning internalization capability enabled by the SAC algorithm.

In summary, the trajectory comparison preliminarily reveals that UGV-1 employs a relatively fixed strategy, UGV-2 a random exploration strategy, and UGV-3 demonstrates a clear performance optimization trend. However, qualitative analysis is insufficient for precisely quantifying the efficiency and stability of the learning process, necessitating further quantitative analysis.

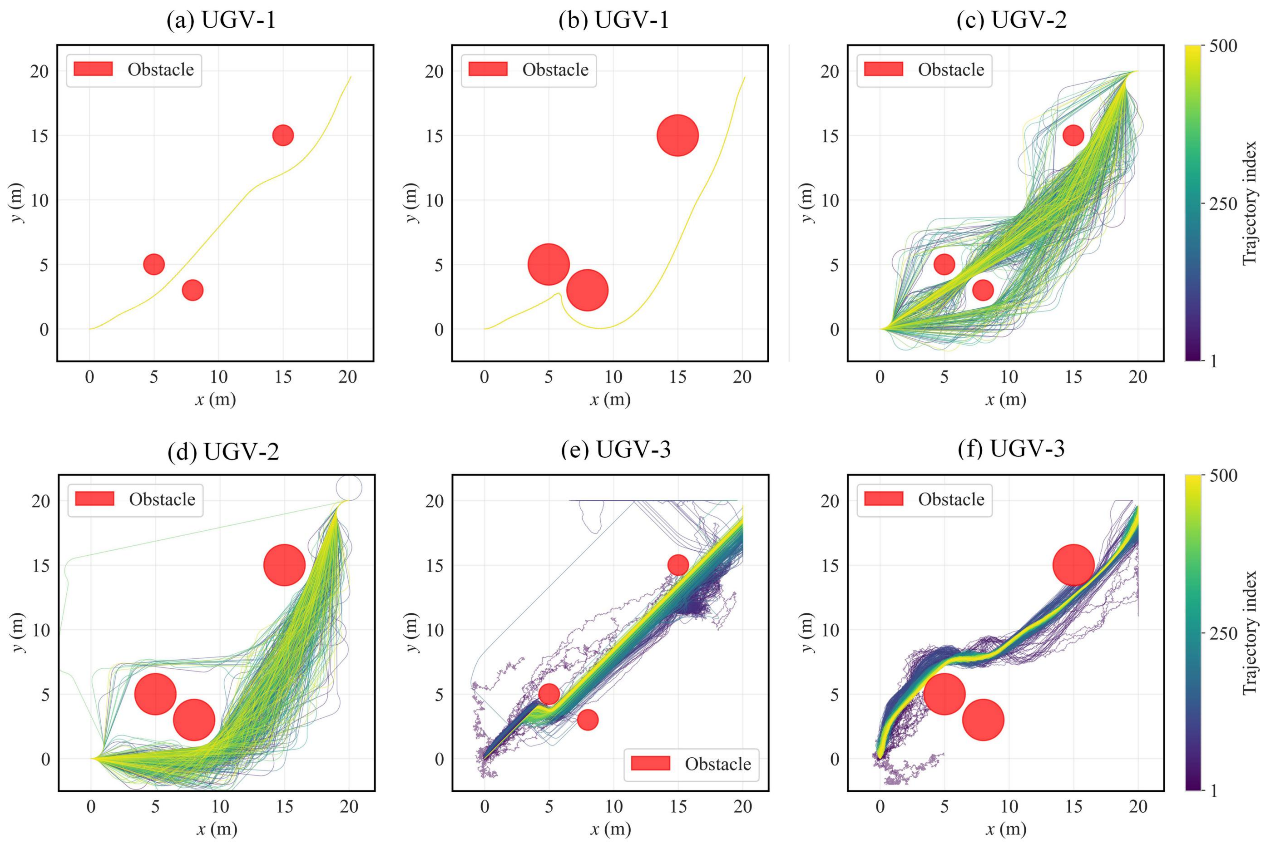


Figure 6. Comparative path planning trajectories of each UGV under different test scenarios: (**a**,**c**,**e**) Γ_1; (**b**,**d**,**f**) Γ_2.

(2) Quantitative Analysis

This section presents performance curves with the number of tests on the horizontal axis and the normalized intelligence evaluation value on the vertical axis. The phase transition points in learning were identified using the sliding window-based slope-standard deviation collaborative analysis method described in Section 3.4.2.

The dynamic evolution features of the learning curves further confirm and refine the conclusions from the trajectory analysis:

The learning curve for UGV-1 consistently fluctuates near a horizontal line. This quantitatively confirms that its performance lacks an improving trend over time. Its intelligence level is entirely predetermined by the initial algorithm parameters, indicating an absence of LIC.

The learning curve for UGV-2 shows narrow fluctuations overall, without distinct Ascent or Convergence Phases. This indicates that its random sampling behavior leads to performance volatility, and the learning effect is not significant.

UGV-3's learning curve clearly shows the typical two-phase pattern. In the Ascent Phase (before t_{point1}), performance rises rapidly with large fluctuations, indicating active exploration. In the Convergence Phase (after t_{point1}), performance stabilizes with minimal fluctuation, showing that a stable policy has been learned. This clear dynamic evolution pattern fully validates the superior LIC of UGV-3.

The contrasting patterns between UGV-2 and UGV-3 objectively separate stochastic exploration from systematic learning internalization. UGV-2's flat, noisy learning curves

(Figure 7) are indicative of random exploration without performance consolidation. In stark contrast, UGV-3's distinct two-phase curves (Figure 7) demonstrate experience-driven policy improvement and stabilization. Therefore, the proposed framework and its metrics naturally discriminate between mere exploration and genuine internalization, addressing a key challenge in evaluating learning algorithms.

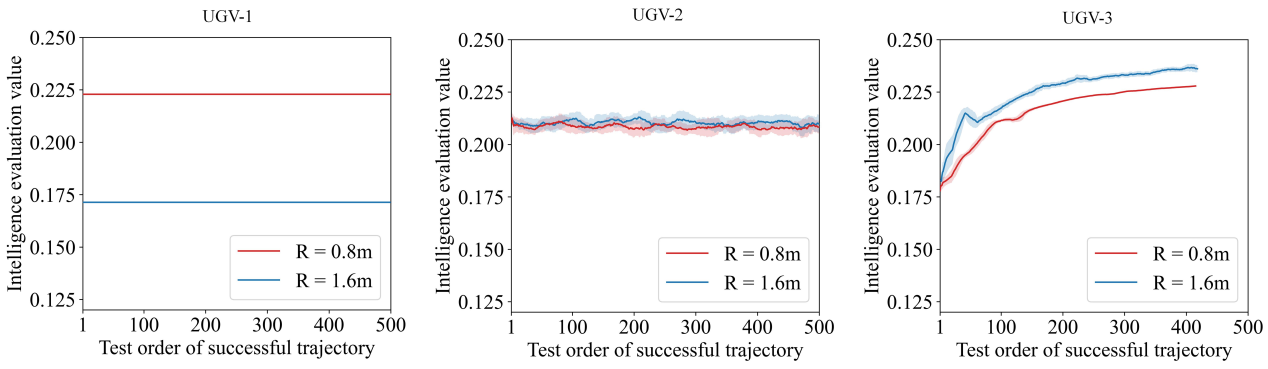


Figure 7. Learning curves with sliding-window standard deviation bands for each UGV.

The proposed sliding-window-based analysis differs in objective from classic learning curve fitting methods (e.g., exponential or power-law models). Those methods aim to approximate the global trend of the entire sequence Y with a smooth parametric function. In contrast, our method is designed for *unsupervised segmentation*; it identifies the local transition point between learning phases by analyzing changes in the local statistics (k_i and σ_i). This approach is more suitable for the non-stationary, fluctuating sequences typical of learning processes. To illustrate this process, Table 2 provides a segment of the raw intermediate metrics (Δk_i, $\Delta\sigma_i$, and S_i) generated by the algorithm for UGV-3 in Γ_2.

Table 2. Sliding window analysis for phase division of UGV-3 in Γ_2 (R = 1.6 m). Only a segment of the data is shown for illustration.

Test Order	Reversed Test Order	Δk_i	$\Delta\sigma_i$	S_i
⋮	⋮	⋮	⋮	⋮
320	80	1.636237	0.028435	−1.664672
321	79	1.753966	0.022396	−1.776362
322	78	1.365370	0.010992	−1.376362
323	77	1.787613	0.021973	−1.809586
324	76	1.264837	0.013216	−1.278053
325 *	75	0.655003	0.007394	−0.662397
326	74	0.751055	0.014326	−0.765381
327	73	1.012440	0.066446	−1.078886
328	72	0.854451	0.025945	−0.880396
329	71	1.161479	0.015769	−1.177248
330	70	0.872914	0.015037	−0.887951
⋮	⋮	⋮	⋮	⋮

* Indicates t_{point1}.

Table 2 presents the key intermediate metrics (Δk_i, $\Delta\sigma_i$, and S_i) from the sliding-window analysis for UGV-3 in Γ_2. The critical transition point t_{point1} (original test order 325) is identified where the composite score S_i drops sharply to -0.6624, signaling a simultaneous stabilization in both trend and fluctuation. This objectively marks the shift from the volatile Ascent Phase to the stable Convergence Phase, validating the phase-division algorithm.

Based on the phase division results from the curves, the LIC was further quantified using Equations (11)–(15). The results are shown in Table 3.

Table 3. Quantitative results of LIC for UGV-3.

Testing Scenarios	t_{point1}	I_{AI} ↑	I_{CV} ↓	I_{MP} ↑	I_{ISR} ↑	I_{AUC} ↑
Γ_1 (R = 0.8 m)	315	0.000119	0.000293	0.225687	0.76	89.908414
Γ_2 (R = 1.6 m)	325	0.000127	0.000420	0.233934	0.54	94.364434
$\Gamma_1 \rightarrow \Gamma_2$		6.62%	43.58%	3.65%	−28.95%	4.96%

↓ is cost-type indicator; ↑ is benefit-type indicator.

When the test scenario changes from $\Gamma_1 \rightarrow \Gamma_2$, the spatial constraints become significantly tighter. This change is clearly reflected in the metric variations. The 43.58% increase in I_{CV} indicates reduced convergence stability in the more complex environment. Meanwhile, the 28.95% decrease in I_{ISR} confirms the greater challenge in achieving success rate improvements under tighter spatial constraints. In contrast, the minor improvements in I_{AI} and I_{MP} suggest the algorithm maintains learning efficiency despite the increased difficulty. These variations demonstrate our metrics' sensitivity in capturing different aspects of LIC under evolving environmental challenges.

5.2. High-Difficulty Test Scenario

In scenario Γ_3, the task success rates for both UGV-1 and UGV-2 were 100%, while for UGV-3 it was 0%, as shown in Figure 8.

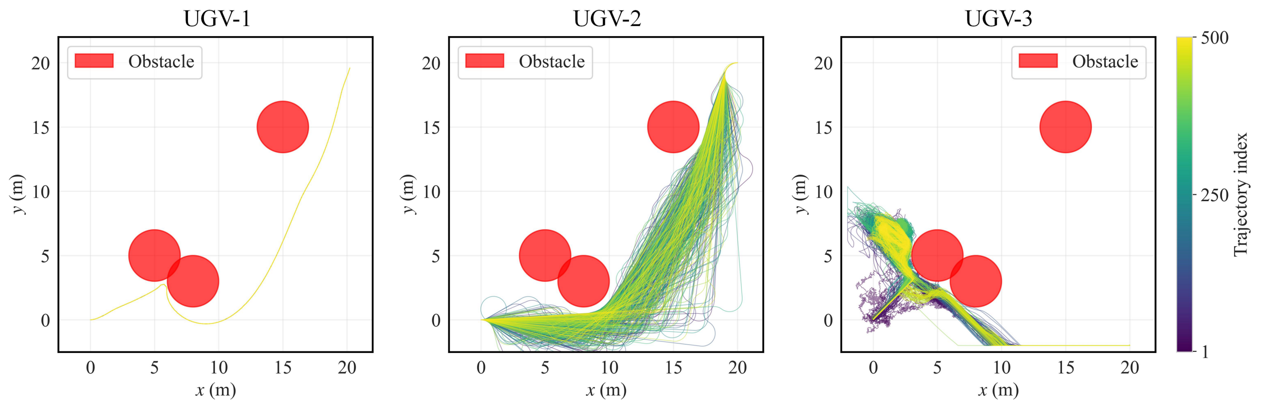


Figure 8. Learning curves and phase transition points of each UGV in Γ_3.

Because UGV-3 possesses learning capability, testing was continued. It found the target point at the 631st test and its performance gradually converged by the 846th test, as shown in Figure 9a. This initial failure phase is primarily due to reward sparsity and policy initialization. In Γ_3, the narrow passage created by the large obstacles made successful exploration and the consequent positive reward rare. Furthermore, the policy pre-trained in simpler scenarios was ineffective, requiring extensive re-exploration. This combination led to the high initial failure rate, which was ultimately overcome, demonstrating UGV-3's strong LIC. Further testing reveals the distinct advantages of the proposed method.

It fully uncovers evolutionary learning potential. As shown in Figure 9b, the comprehensive performance of UGV-3 gradually improves with the number of tests, eventually surpassing that of the other vehicles. This contrast highlights a critical limitation of static evaluation methods. A static assessment based solely on initial performance would have prematurely dismissed UGV-3 as ineffective, while our dynamic method correctly identifies its strong long-term potential by capturing the complete evolutionary trajectory of the performance sequence Y. This demonstrates how dynamic evaluation provides superior decision-support for selecting algorithms with learning capability.

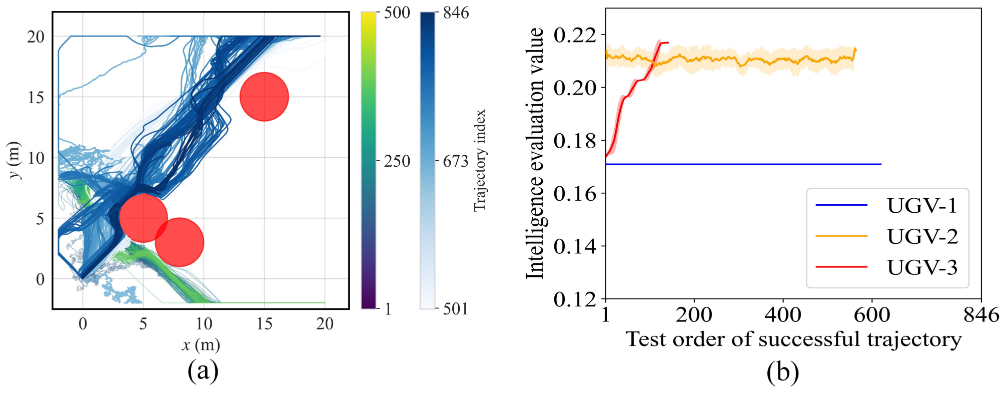


Figure 9. Performance evolution and trajectory of UGV-3 in Γ_3 with sliding-window standard deviation bands. (**a**) Trajectory of UGV-3 from test 1 to 846; (**b**) Learning curve of UGV-3, showing gradual performance improvement and eventual convergence.

To clearly compare the characteristics of the different algorithms, Table 4 summarizes the LIC profiles of each UGV across the three scenarios. It should be noted that the LIC metrics in our framework are specifically designed to quantify systems with learning capability. As non-learning algorithms, UGV-1 and UGV-2 (RRT) do not exhibit systematic improvement trends in their performance sequences. Therefore, the relevant LIC metrics are not applicable. Consequently, the table only presents the general state of their learning curves.

Table 4. Comparative analysis of learning behavior for all UGVs.

UGV Version	Testing Scenario	t_{point1}	I_{AI} ↑	I_{CV} ↓	I_{MP} ↑	I_{ISR} ↑	I_{AUC} ↑
UGV-1	Γ_1 Γ_2 Γ_3	Learning curve is horizontal; no learning state.					
UGV-2	Γ_1 Γ_2 Γ_3	Learning curve fluctuates slightly; no obvious learning state.					
UGV-3	Γ_1	315	0.000119	0.000293	0.225687	0.76	89.908414
	Γ_2	325	0.000127	0.000420	0.233934	0.54	94.364434
	Γ_3	846	0.000049	0.004399	0.216292	0.61	4.975813

↓ is cost-type indicator; ↑ is benefit-type indicator.

Combining the information in Table 4 with the learning curves (Figures 7–9), we observe the following: UGV-1 shows a flat performance curve, relying solely on preset rules with no iterative improvement; UGV-2 exhibits fluctuations stemming from random sampling, also lacking any systematic learning trend. Neither demonstrates measurable LIC. Only UGV-3 displays a clear two-phase learning pattern (efficiency gain followed by stable convergence) in Γ_1/Γ_2, and although its learning is challenged in Γ_3, it eventually converges. The LIC metrics effectively quantify this dynamic process and the differences in its scenario adaptability. For UGV-3 in Γ_3, only I_{ISR} is slightly higher than in Γ_2, while other metrics are significantly lower than in the low-difficulty scenarios. This indicates the vehicle's weaker adaptability in high-difficulty environments. This trend aligns with that shown in Figure 10, further validating the sensitivity of the LIC metrics to changes in scenario difficulty.

While UGV-3 converges in Γ_3, its slower learning, reduced stability, and lower overall performance indicate a limited learning capability in high-difficulty settings. This confirms its shortcomings in robust path planning and obstacle avoidance, aligning with the analysis in Section 5.1.

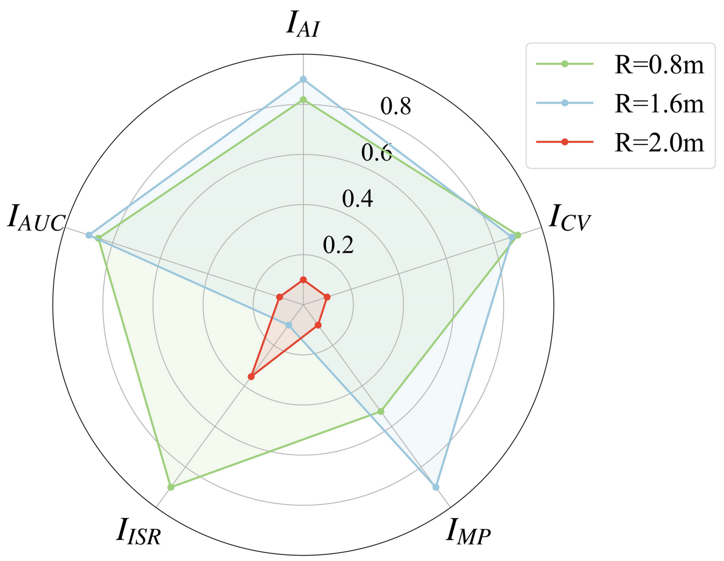


Figure 10. Multi-dimensional performance of LIC in the three test scenarios.

5.3. Scenario Suitability Recommendations

Based on the qualitative and quantitative evaluation of the functional characteristics of the UGVs, and considering the practical requirements of different task scenarios, the following recommendations are proposed:

UGV-1: Its core capability is limited to executing preset instructions, lacking mechanisms for experience accumulation and autonomous iteration. It is suitable for fixed, enclosed scenarios with singular tasks that do not require performance optimization. Typical applications include material transfer between fixed production lines in factory workshops and guided vehicle routing in enclosed ports.

UGV-2: It possesses environmental exploration capability but cannot achieve experience transfer, and its performance is volatile. It is suitable for low-priority, simple exploration tasks that tolerate trial and error. Example scenarios include preliminary terrain surveying in undeveloped mining areas and temporary equipment handling in large venues.

UGV-3: It possesses experience-driven performance iteration capability. It is suitable for core scenarios characterized by high dynamics, multiple variables, and a reliance on adaptive capability. Typical applications include dynamic loading/unloading scheduling in container ports and real-time security dispatch during large-scale events.

6. Conclusions

This study proposed and validated a dynamic evaluation framework for LIC in UGVs. The simulation results quantitatively demonstrate its efficacy: for the learning-capable UGV-3, our metrics captured a 6.62% increase in I_{AI} and a 43.58% increase in I_{CV} when environmental difficulty escalated from Γ_1 to Γ_2, while its capability to I_{ISR} decreased by 28.95%. These data concretely illustrate the framework's sensitivity in dissecting the multi-dimensional impact of complexity on learning internalization.

Through theoretical modeling, method design, and simulation validation, the following core conclusions are drawn:

(i) This study provides a clear dualistic definition of learnability, dividing it into LIC and LGC. This addresses a gap in UGV intelligence evaluation by introducing the learnability dimension, moving beyond the limitations of traditional static evaluations that focus solely on instantaneous performance statistics.

(ii) A multidimensional scenario parameter system encompassing environment, task object, and system self-parameters was constructed. The sliding window-based slope-standard deviation collaborative analysis technique was proposed, enabling the objective division of learning phases.

(iii) Five evaluation metrics were designed, covering dimensions such as learning efficiency, stability, and comprehensive effectiveness. Experimental results confirm that these metrics possess good discriminative power and interpretability.

(iv) This framework offers practical value for research requiring quantitative comparison of learning algorithms, as well as for applied scenarios involving UGV selection and training optimization.

The limitations of this study lie in its focus on the path planning task, without covering other intelligence modules such as perception and decision-making. Furthermore, the scenario difficulty was primarily varied through obstacle size, which, while effective for initial validation, represents a single dimension of complexity. Other pertinent dimensions such as sensor noise, dynamic disturbances, partial observability, and obstacle speed/density were not included, excluding real-world factors like lighting and dynamic disturbances. Notably, as the current validation is simulation-based, the transfer of the proposed framework to physical platforms would introduce specific challenges, including sensor noise and hardware degradation, which could perturb the performance sequence and impair the accuracy of phase segmentation. Future work will therefore focus on methodological adaptations, such as incorporating robust filtering techniques during data preprocessing and adjusting key hyperparameters, to enhance the framework's applicability and reliability in real-world robotic deployments. Furthermore, expanding the test scenario space along these additional dimensions will be a critical step in future research to fully assess the robustness and generalizability of the proposed LIC evaluation method.

Author Contributions: Conceptualization, Z.D., J.Y. and X.L.; methodology, Z.D., G.S. and X.L.; software, Z.D. and G.S.; validation, Z.D., J.Y. and M.L.; formal analysis, G.S., Y.G., M.L. and Y.S.; investigation, J.Y., Y.G. and M.L.; data curation, Z.D., J.Y., M.L. and Y.S.; writing—original draft preparation, Z.D., G.S. and Y.G.; writing—review and editing, Z.D., G.S., Y.G., X.L. and Y.S.; supervision, Z.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wu, S.; Li, S.; Gong, J.; Yan, Z. Modeling and quantitative evaluation method of environmental complexity for measuring autonomous capabilities of military unmanned ground vehicles. *Unmanned Syst.* **2023**, *11*, 367–382. [CrossRef]
2. Almeida, J.; Rufino, J.; Cardoso, F.; Gomes, M.; Ferreira, J. Trust: Transportation and road monitoring system for ubiquitous real-time information services. In Proceedings of the 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium, 25–28 May 2020; pp. 1–7.

3. Dong, X.; Hua, Y.; Zhou, Y.; Ren, Z.; Zhong, Y. Theory and experiment on formation-containment control of multiple multirotor unmanned aerial vehicle systems. *IEEE Trans. Autom. Sci. Eng.* **2018**, *16*, 229–240. [CrossRef]
4. Wies, N.; Levine, Y.; Shashua, A. The learnability of in-context learning. *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 36637–36651.
5. Brukhim, N.; Carmon, D.; Dinur, I.; Moran, S.; Yehudayoff, A. A characterization of multiclass learnability. In Proceedings of the 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS), Denver, CO, USA, 31 October–3 November 2022; pp. 943–955.
6. Schapire, R.E. The strength of weak learnability. *Mach. Learn.* **1990**, *5*, 197–227. [CrossRef]
7. Tesar, B.; Smolensky, P. Learnability in optimality theory. *Linguist. Inq.* **1998**, *29*, 229–268. [CrossRef]
8. Li, Y.; Zhang, Y.; Li, X.; Sun, C. Regional multi-agent cooperative reinforcement learning for city-level traffic grid signal control. *IEEE/CAA J. Autom. Sin.* **2024**, *11*, 1987–1998. [CrossRef]
9. Yu, Y.; Li, J.; Wu, T. A Group Target Tracking Method for Unmanned Ground Vehicles Based on Multi-Ellipse Shape Modeling. *Drones* **2025**, *9*, 620. [CrossRef]
10. Dong, Z. Dynamic assessment of threats to cluster targets inlowaltitude multi-domain battlefields. *J. Tsinghua Univ. (Sci. Technol.* **2024**, *64*, 1380–1390.
11. Hernández-Orallo, J. Evaluation in artificial intelligence: From task-oriented to ability-oriented measurement. *Artif. Intell. Rev.* **2017**, *48*, 397–447. [CrossRef]
12. Yu, X.; He, M.; Chang, S. Research on methods for evaluating the autonomous combat effectiveness of unmanned ground equipment. *Mil. Oper. Res. Syst. Eng.* **2020**, *34*, 26–32.
13. Guerrero-Sevilla, D.; Gonzalez-de Soto, M.; Del Pozo, S.; Martín-Jiménez, J.A.; Rodríguez-Gonzálvez, P.; González-Aguilera, D. Enhancing Overtaking Safety with Mobile LiDAR Systems: Dynamic Analysis of Road Visibility. *Remote Sens.* **2025**, *17*, 2948. [CrossRef]
14. Dong, Z.; Yang, J.; Yuan, R.; Su, G.; Lei, M. A Game-Theoretic Kendall's Coefficient Weighting Framework for Evaluating Autonomous Path Planning Intelligence. *Automation* **2025**, *6*, 85. [CrossRef]
15. Deng, Y.; Chow, A.H.; Yan, Y.; Su, Z.; Zhou, Z.; Kuo, Y.H. Hierarchical production control and distribution planning under retail uncertainty with reinforcement learning. *Int. J. Prod. Res.* **2025**, *63*, 4504–4522. [CrossRef]
16. Dodig-Crnkovic, G.; Burgin, M. A systematic approach to autonomous agents. *Philosophies* **2024**, *9*, 44. [CrossRef]
17. Ali, Y.; Hussain, F.; Haque, M.M. Advances, challenges, and future research needs in machine learning-based crash prediction models: A systematic review. *Accid. Anal. Prev.* **2024**, *194*, 107378. [CrossRef]
18. Ali, Y.; Haque, M.M.; Mannering, F. Assessing traffic conflict/crash relationships with extreme value theory: Recent developments and future directions for connected and autonomous vehicle and highway safety research. *Anal. Methods Accid. Res.* **2023**, *39*, 100276. [CrossRef]
19. Almeida, J.; Jooriah, M.; Ferreira, J.; Dias, T.; Silva, A.V.; Moura, L. 5G connected vehicle and roadside infrastructure for advanced driving maneuvers in a cross-border scenario. *Transp. Res. Procedia* **2023**, *72*, 2808–2815. [CrossRef]
20. Sun, Y.; Yang, H.; Meng, F. Research on an intelligent behavior evaluation system for unmanned ground vehicles. *Energies* **2018**, *11*, 1764. [CrossRef]
21. Guo, M.; Zheng, W.; Xu, C. Research and Effectiveness Analysis of Intelligent Unmanned Equipment Group Collaboration in Combat Simulation System. In Proceedings of the 2024 IEEE 2nd International Conference on Sensors, Electronics and Computer Engineering (ICSECE), Jinzhou, China, 29–31 August 2024; pp. 1049–1055.
22. Li, S.; Chen, C.; Fang, X. Review of Indicator Systems for Driving Behavioral Ability Evaluation of Autonomous Vehicles. *China J. Highw. Transp* **2025**, *38*, 304–323.
23. Zhu, B.; Zhang, P.; Liu, B.; Sun, Y. Safety Evaluation Method of Automated Vehicle Based on Naturalistic Driving Data. *China J. Highw. Transp* **2022**, *35*, 283–291.
24. Yang, L.; Zhao, Y.; Zhang, X. Review on testing and evaluation of cognitive abilities for autonomous vehicles. *J. Automot. Saf. Energy* **2025**, *16*, 345–358.
25. Zhou, J.; Wang, L.; Meng, Q.; Wang, X. Intelligence Evaluation Methods for Autonomous Vehicles. In Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA), Atlanta, GA, USA, 19–23 May 2025; pp. 10600–10606.
26. Langford, M.A.; Chan, K.H.; Fleck, J.E.; McKinley, P.K.; Cheng, B.H. MoDALAS: Addressing assurance for learning-enabled autonomous systems in the face of uncertainty. *Softw. Syst. Model.* **2023**, *22*, 1543–1563. [CrossRef] [PubMed]
27. Mbelekani, N.Y.; Bengler, K. Learnability in Automated Driving (LiAD): Concepts for applying learnability engineering (CALE) based on long-term learning effects. *Information* **2023**, *14*, 519. [CrossRef]
28. Casini, M.; Garulli, A.; Giannitrapani, A.; Vicino, A. A remote lab for experiments with a team of mobile robots. *Sensors* **2014**, *14*, 16486–16507. [CrossRef] [PubMed]
29. Viering, T.; Loog, M. The shape of learning curves: A review. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 7799–7819. [CrossRef]

30. Singh, S.; Jaakkola, T.; Littman, M.L.; Szepesvári, C. Convergence results for single-step on-policy reinforcement-learning algorithms. *Mach. Learn.* **2000**, *38*, 287–308. [CrossRef]
31. Stokes, C.K.; Schneider, T.R.; Lyons, J.B. Adaptive performance: A criterion problem. *Team Perform. Manag. Int. J.* **2010**, *16*, 212–230. [CrossRef]
32. Francis, A.; Pérez-D'Arpino, C.; Li, C.; Xia, F.; Alahi, A.; Alami, R.; Bera, A.; Biswas, A.; Biswas, J.; Chandra, R.; et al. Principles and Guidelines for Evaluating Social Robot Navigation Algorithms. *J. Hum. Robot Interact.* **2025**, *14*, 1–65. [CrossRef]
33. Barzamini, H.; Rahimi, M. B-AIS: An Automated Process for Black-box Evaluation of Visual Perception in AI-enabled Software against Domain Semantics. In Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering, Rochester, MI, USA, 10–14 October 2022; pp. 1–13.
34. Mazur, J.E.; Hastie, R. Learning as accumulation: A reexamination of the learning curve. *Psychol. Bull.* **1978**, *85*, 1256. [CrossRef]
35. Amelio, A.; Pizzuti, C. Is normalized mutual information a fair measure for comparing community detection methods? In Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, Paris, France, 25–28 August 2015; pp. 1584–1585.
36. Casolaro, A.; Capone, V.; Iannuzzo, G.; Camastra, F. Deep learning for time series forecasting: Advances and open problems. *Information* **2023**, *14*, 598. [CrossRef]
37. Hu, Y.; Fu, J.; Wen, G.; Lv, Y.; Ren, W. Distributed entropy-regularized multi-agent reinforcement learning with policy consensus. *Automatica* **2024**, *164*, 111652. [CrossRef]
38. Mendes, P.; Batista, P.; Oliveira, P.; Silvestre, C. Cooperative decentralized navigation algorithms based on bearing measurements for arbitrary measurement topologies. *Ocean. Eng.* **2023**, *270*, 113564. [CrossRef]
39. Camastra, F.; Capone, V.; Ciaramella, A.; Riccio, A.; Staiano, A. Prediction of environmental missing data time series by Support Vector Machine Regression and Correlation Dimension estimation. *Environ. Model. Softw.* **2022**, *150*, 105343. [CrossRef]
40. Wambui, G.D.; Waititu, G.A.; Wanjoya, A. The power of the pruned exact linear time (PELT) test in multiple changepoint detection. *Am. J. Theor. Appl. Stat.* **2015**, *4*, 581. [CrossRef]
41. Healy, J.D. A note on multivariate CUSUM procedures. *Technometrics* **1987**, *29*, 409–412. [CrossRef]
42. Xiao, X.; Xu, Z.; Datar, A.; Warnell, G.; Stone, P.; Damanik, J.J.; Jung, J.; Deresa, C.A.; Huy, T.D.; Jinyu, C.; et al. Autonomous Ground Navigation in Highly Constrained Spaces: Lessons Learned From the Third BARN Challenge at ICRA 2024 [Competitions]. *IEEE Robot. Autom. Mag.* **2024**, *31*, 197–204. [CrossRef]
43. Xiao, X.; Xu, Z.; Warnell, G.; Stone, P.; Guinjoan, F.G.; Rodrigues, R.T.; Bruyninckx, H.; Mandala, H.; Christmann, G.; Blanco-Claraco, J.L.; et al. Autonomous ground navigation in highly constrained spaces: Lessons learned from the second barn challenge at icra 2023 [competitions]. *IEEE Robot. Autom. Mag.* **2023**, *30*, 91–97. [CrossRef]
44. Bakirci, M. Simulation of autonomous driving for a line-following robotic vehicle: Determining the optimal manoeuvring mode. *Elektron. Elektrotechnika* **2023**, *29*, 4–11. [CrossRef]
45. Pang, N.; Luo, W.; Wu, R.; Lan, H.; Qin, Y.; Su, Q. Safety evaluation of commercial vehicle driving behavior using the AHP—CRITIC algorithm. *J. Shanghai Jiaotong Univ. (Sci.)* **2023**, *28*, 126–135. [CrossRef]
46. Hu, Y.; Long, H.; Chen, M. The analysis of pedestrian flow in the smart city by improved DWA with robot assistance. *Sci. Rep.* **2024**, *14*, 11456. [CrossRef]
47. Wu, Z.; Meng, Z.; Zhao, W.; Wu, Z. Fast-RRT: A RRT-based optimal path finding method. *Appl. Sci.* **2021**, *11*, 11777. [CrossRef]
48. Song, L.; Xu, C.; Hao, L.; Yao, J.; Guo, R. Research on PID parameter tuning and optimization based on SAC-auto for USV path following. *J. Mar. Sci. Eng.* **2022**, *10*, 1847. [CrossRef]