

Agentic AI-Empowered Multi-UAV Trajectory Optimization in Low-Altitude Economy Networks

Feibo Jiang^{ID}, Senior Member, IEEE, Xitao Pan, Li Dong^{ID}, Kezhi Wang^{ID}, Senior Member, IEEE, Xiaolong Li^{ID}, Ruidong Li^{ID}, Changhong Wang, and Cunhua Pan^{ID}, Senior Member, IEEE

Abstract—This study proposes a novel Agentic Retrieval-augmented generation with Mamba-Attention Integrated Transformer (ARMAIT) framework for multi-Uncrewed Aerial Vehicle (UAV) trajectory optimization. The framework is built upon Large Language Models (LLMs), incorporating Retrieval-Augmented Generation (RAG) empowered by Agentic AI and integrated with a UAV-specific knowledge base. Through the Agentic RAG, the LLM autonomously interprets high-level task requirements and identifies the key components necessary for trajectory optimization, including model inputs and outputs, network architecture, reward functions, and task constraints. To support efficient modeling across different system scales, we introduce the Mamba-Attention Integrated Transformer (MAIT), a hybrid neural architecture that combines the long-range dependency modeling capability of attention mechanisms with the efficient temporal dynamic representation of Mamba. Furthermore, a Trajectory-Group Relative Policy Optimization (T-GRPO) method is proposed to achieve unified policy gradient optimization in both discrete and continuous trajectory spaces for MAIT training. Extensive experimental results validate the feasibility and effectiveness of the proposed ARMAIT framework.

Index Terms—Agentic AI, large language model, Mamba, uncrewed aerial vehicle, GRPO, Agentic RAG.

I. INTRODUCTION

WITH the rapid integration of Information and Communication Technologies (ICT) and Uncrewed Aerial Vehicle (UAV) technologies, Low-Altitude Economy Networks (LAENets) have emerged as an important infrastructure for future societal services [1]. Operating below

3,000 meters, LAENets rely on diverse Low-Altitude Vehicles (LAVs)—including aerial taxis, airborne base stations, and charging platforms—to provide logistics, communication coverage, and aerial monitoring. Achieving these capabilities requires the tight integration of communication, computation, storage, and sensing. However, LAENets differ fundamentally from terrestrial networks in control and resource management. Their highly dynamic and uncertain environments increase system complexity and expose the network to greater security risks, necessitating robust and intelligent system designs.

In LAENets, UAVs act as core airborne nodes due to their mobility and flexible deployment. They can operate as aerial base stations or relays to extend coverage in remote or underserved regions and have demonstrated strong utility in applications such as medical delivery, daily logistics, and last-mile distribution [2]. Because UAV task performance and trajectory planning directly influence overall network efficiency, developing effective UAV scheduling and trajectory optimization mechanisms is essential for ensuring stable and reliable LAENet operations.

However, Deep Learning (DL)-based UAV trajectory planning methods lack cross-scenario transfer ability and generalization, making them insufficient for the diverse tasks, heterogeneous environments, and dynamic scale variations characteristic of LAENets. To overcome these bottlenecks, recent research has increasingly shifted toward Artificial General Intelligence (AGI), which emphasizes cross-task understanding, generalized reasoning, and autonomous planning, enabling agents to interpret natural-language intents, integrate heterogeneous knowledge sources, and adapt to unseen environments [3], [4], [5]. In UAV trajectory planning, AGI can automatically distill constraints and key variables from high-level task objectives and synthesize executable, cross-scenario trajectory policies, thereby advancing an integrated pipeline that unifies task understanding, policy generalization, and trajectory generation.

Against this backdrop, Transformer-based architectures have become central to general-purpose UAV trajectory planning, thanks to their ability to handle variable-length sequences, capture global dependencies, and exploit parallel computation. Self-attention supports stable global decision-making in complex spatial settings and naturally adapts to dynamic changes in device density, task scale, and state dimensionality in LAENets [6]. However, full self-attention incurs quadratic time and memory costs with respect to sequence length, which has spurred growing interest in Mamba—state-space-based

Received 6 July 2025; revised 15 November 2025 and 26 February 2026; accepted 14 March 2026. Date of publication 6 April 2026; date of current version 7 May 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 62572184 and 41604117, in part by Hunan Provincial Natural Science Foundation of China under Grant 2024JJ5270 and Grant 2025JJ50365, in part by Changsha Natural Science Foundation under Grant kq2402098 and Grant kq2402162. The associate editor coordinating the review of this article and approving it for publication was J. Wang. (Corresponding author: Xiaolong Li.)

Feibo Jiang and Xitao Pan are with the Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing and the Yuelushan Digital Intelligence Laboratory (Artificial Intelligence and International Communication), Hunan Normal University, Changsha 410081, China (e-mail: jiangfb@hunnu.edu.cn; 1837349760@hunnu.edu.cn).

Li Dong and Xiaolong Li are with the School of Computer Science, Hunan University of Technology and Business, Changsha 410205, China (e-mail: Dlj2017@hunnu.edu.cn; xlli@guet.edu.cn).

Kezhi Wang is with the Department of Computer Science, Brunel University of London, UB8 3PH London, U.K. (e-mail: Kezhi.Wang@brunel.ac.uk).

Ruidong Li is with Shandong Yunhai Guochuang Cloud Computing Equipment Industry Innovation Company Ltd., Jinan 250100, China (e-mail: lird@inspur.com).

Changhong Wang is with Jinan Maiwei Intelligent Technology Co., Ltd., Jinan 250100, China (wangchh01@inspur.com).

Cunhua Pan is with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: cpan@seu.edu.cn). Digital Object Identifier 10.1109/TCCN.2026.3680925

architectures that model long sequences with linear complexity while preserving temporal dynamics [7].

Moreover, Retrieval-Augmented Generation (RAG) provides explicit knowledge grounding and contextual adaptability by retrieving from external knowledge bases at inference time [8]. It can inject domain priors—such as flight regulations, aerodynamic models, energy-consumption formulations, and classical scheduling heuristics—into the model’s reasoning, mitigating the lack of prior knowledge in purely end-to-end learning and improving interpretability and situational robustness [9]. However, conventional RAG is largely passive and query-driven, with limited task-aware selection and little ability to adapt retrieval strategies during reasoning. These limitations motivate Agentic RAG, which enables models to proactively plan, retrieve, and verify evidence, yielding more reliable, domain-aligned, and deployable trajectory reasoning.

To advance the autonomous decision-making and trajectory optimization capabilities of multi-UAV systems in LAENets, this paper explores the integration of the Agentic Artificial Intelligence (AI) paradigm. By combining Large Language Models (LLMs), trajectory planning models, and Reinforcement Learning (RL) algorithms, we construct an Agentic RAG with Mamba-Attention Integrated Transformer (ARMAIT) framework that unifies task comprehension, knowledge retrieval, model formulation and training, and trajectory generation. The main contributions of this study are summarized as follows:

1) *Agentic RAG for Task Modeling*: We design an Agentic AI framework that integrates LLMs with a UAV-specific knowledge base. Leveraging an Agentic RAG mechanism, the LLM is empowered to autonomously interpret user intent from natural language task instructions, retrieve relevant UAV knowledge, and automatically generate the required model components, including input representations (e.g., state space), output objectives (e.g., action space), model architecture (e.g., the ratio of Attention to Mamba layers), reward function definitions, and task constraints. This mechanism establishes a closed-loop process from language understanding to task modeling, significantly enhancing the adaptability and generality of multi-UAV trajectory planning tasks.

2) *MAIT for Trajectory Optimization*: We propose a Mamba-Attention Integrated Transformer (MAIT) model, which combines the spatial global dependency modeling strength of multi-head attention with the efficient long-sequence modeling capability of the Mamba architecture. MAIT supports parallel interactions among UAVs while capturing temporal dynamics essential for trajectory planning. The UAV system states are encoded into structured input sequences, and the model architecture adapts the attention-to-Mamba layer ratio automatically, favoring attention layers for smaller-scale UAV swarms and Mamba layers for larger-scale ones. This design ensures strong generalization and high decision-making efficiency.

3) *T-GRPO for Model Training*: We present Trajectory-Group Relative Policy Optimization (T-GRPO), a policy-gradient Reinforcement Learning (RL) algorithm tailored for

multi-UAV trajectory generation. Building upon the original GRPO method, T-GRPO extends its applicability to both discrete (e.g., stop point sequences) and continuous (e.g., flight angles and distances) action spaces. The algorithm defines state and action spaces for UAV swarms as well as a global reward function. It introduces a group-based relative advantage function and policy divergence constraints to optimize the policy. This enables the stable training of the MAIT model for generating globally coordinated multi-UAV trajectories across diverse task scenarios.

This paper is organized as follows: Section II reviews some related work. Section III describes the principle of the ARMAIT framework in detail. Sections IV–VI illustrate the experimental results. Finally, Section VII concludes the paper and discusses future work.

II. RELATED WORK

A. Reinforcement Learning-Enhanced UAV Systems

Wang et al. proposed a Deep Reinforcement Learning (DRL)-based trajectory control algorithm for UAV-assisted mobile edge computing, aiming to minimize system energy consumption through dynamic path adaptation [10]. Dong et al. utilized an Actor-Critic (AC)-based method to train an attention-based UAV trajectory optimizer [11]. Xue et al. introduced a data-driven trajectory imitation method grounded in inverse reinforcement learning, employing static output feedback control to enhance stability and control accuracy [12]. Additionally, Liu and Huang developed an adaptive and robust DDPG framework for aerial manipulators, focusing on quadrotor tracking control while mitigating the impact of robotic-arm motion on flight dynamics [13].

B. Transformer-Enhanced UAV Systems

In multi-UAV area coverage, Chen et al. employed a Transformer encoder within their T-MARL algorithm to adapt to variable numbers of UAVs and users [14]. Zhu et al. utilized a Transformer encoder-decoder architecture to translate the entire UAV-IoT system state into an optimized visiting order, achieving good generalization and computational efficiency [15]. Feng et al. embedded a lightweight Transformer into the critic network of each UAV base station in their EDT-MARL framework [16]. Furthermore, Wang et al. used a Vision Transformer (ViT)-based model for robust visual feature learning in the main tracker, complemented by a separate Transformer-based module for predicting target motion trajectories from sequential data [17].

C. Mamba-Enhanced UAV Systems

Xiao et al. proposed a temporally audio-driven Mamba framework that fuses spectral and temporal information to achieve accurate drone trajectory estimation and classification under complex acoustic conditions [18]. Tao et al. introduced FDMamba-Net, which incorporates feature decoupling along with multi-layer Mamba blocks, enabling efficient and highly accurate flight trajectory prediction with significantly reduced

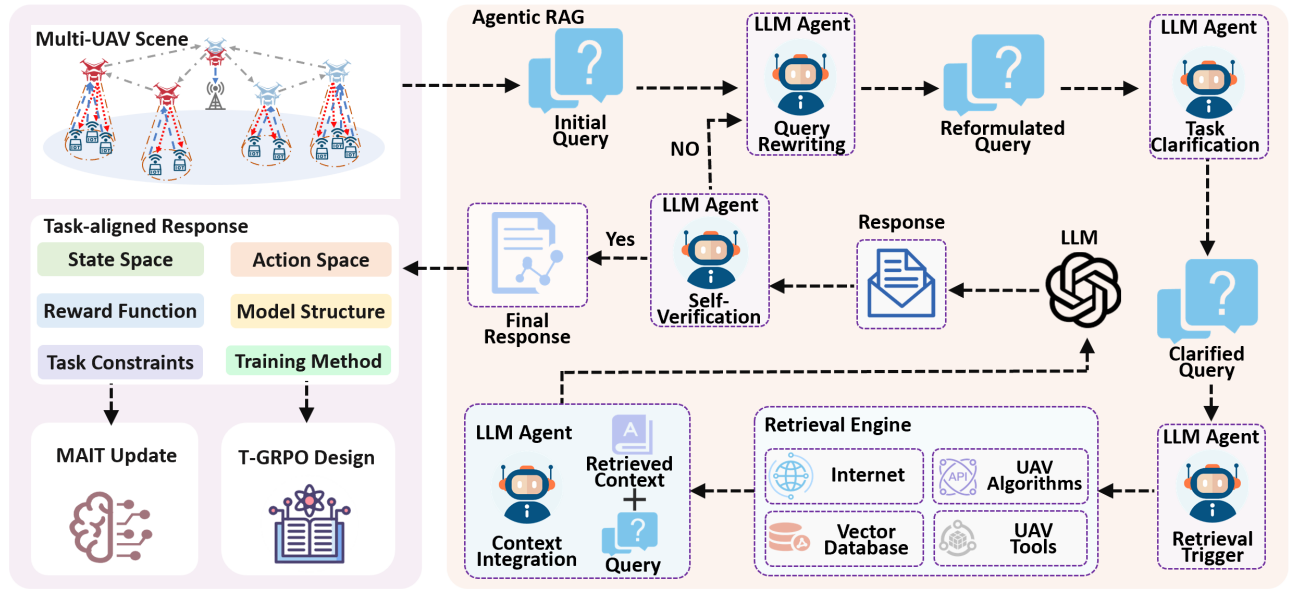


Fig. 1. Workflow of the ARMAIT framework.

model complexity [19]. Liang et al. developed a motion-aligned visual Mamba network that leverages self-supervised learning for long-horizon 3D UAV trajectory forecasting without requiring manual labels [20].

D. Agentic AI-Enhanced UAV Systems

Sapkota et al. proposed a foundational framework for Agentic UAVs, which integrates perception, reasoning, and learning to enhance autonomous aerial operation across diverse domains [21]. Zhao et al. designed an energy-efficient Agentic semantic wireless network, where UAVs leverage Agentic intelligence to optimize communication, task offloading, and resource allocation [22]. Furthermore, Koubaa and Gabr introduced an LLM-driven UAV framework with integrated tool-calling capabilities, demonstrating significant gains in cognitive reasoning, detection reliability, and autonomous action planning [23].

However, existing UAV trajectory planning methods are typically designed for specific system models and lack the flexibility to accommodate diverse scenarios and task requirements. To address this limitation, we propose the ARMAIT framework, which integrates LLMs and an Agentic RAG mechanism to interpret varying task demands and convert them into concrete design specifications.

III. ARMAIT FRAMEWORK

The ARMAIT framework comprises three key components: an Agentic RAG-based task analysis module, the MAIT trajectory generator, and the T-GRPO optimizer. The Agentic RAG module couples an LLM with a UAV-domain knowledge base to parse natural-language task descriptions, infer the required problem formulation, and automatically instantiate the model architecture and data interfaces for MAIT and T-GRPO. MAIT integrates attention-based global representation with Mamba's efficient sequence modeling to produce coordinated trajectories for multiple UAVs. T-GRPO jointly

optimizes policies over discrete and continuous action spaces and trains MAIT to generate trajectories that satisfy multiple constraints. Together, these modules form a closed-loop, end-to-end pipeline that bridges natural-language understanding and multi-UAV trajectory planning, as illustrated in Fig. 1.

A. Agentic RAG Task Analysis Module

To effectively analyze diverse UAV path-planning tasks, LLMs have been increasingly adopted for their strong natural-language understanding and reasoning capabilities. However, existing LLM-based task analysis often lacks sufficient UAV-domain knowledge, which can lead to formulation errors and hallucinated content, ultimately undermining the generation of task-aligned planning solutions in real-world deployments. RAG alleviates these issues by introducing semantic retrieval prior to generation, enriching the query with contextual evidence and thereby reducing hallucinations and knowledge staleness [24]. Nevertheless, when applied to UAV task analysis, conventional RAG still suffers from two key limitations: (1) The user query cannot dynamically adapt to the evolving task context of UAVs, resulting in inconsistencies between the retrieved content and the generation objective; (2) The lack of procedural control prevents dynamic adjustment of the retrieval strategy or refinement of the query expression based on intermediate outputs, thereby inhibiting autonomous reflection and self-correction.

To overcome the limitations of static modeling and single-round inference in traditional RAG systems, this study introduces a novel Agentic RAG architecture for UAV task modeling and reasoning. This system integrates the core principles of Agentic AI, which endow LLMs with autonomous perception, reasoning, and proactive planning capabilities. By enabling dynamic decision-making and self-directed retrieval for complex UAV tasks, Agentic RAG establishes a closed-loop task-solving pipeline with autonomous reasoning chains for trajectory planning. The following outlines the operational

workflow of Agentic RAG for analyzing UAV trajectory planning tasks:

1) *Initial Query Reception*: The system receives a natural language input from the user describing a UAV-related task (e.g., “Please dispatch 10 UAVs to collect data from 100 IoT devices in a designated area, optimizing for energy consumption”).

2) *Query Rewriting*: Upon understanding the task objectives and key variables, the LLM Agent semantically reconstructs the initial input. For example, the input may be reformulated as: “Based on the criterion of minimizing energy consumption, construct a multi-UAV trajectory planning model using reinforcement learning to coordinate the flight paths of 10 UAVs for efficient data collection from 100 IoT devices in a target area.”

3) *Task Clarification*: The Agent determines whether additional background information is required, such as scenario models, flight altitude, or battery constraints. It then prompts the user to provide these parameters to complete the necessary prior knowledge. For example, the Agent may automatically ask: “Please provide the geographical distribution of IoT devices, flight distance limitations, and the maximum payload capacity of each UAV for improved modeling accuracy.”

4) *Context Retrieval*: The clarified query is submitted to a retrieval engine, which includes a vector database constructed from UAV trajectory planning literature, a tool library composed of various UAV-related models (such as UAV dynamics models and wireless channel models), an algorithm library containing diverse UAV-related algorithms (such as convex optimization algorithms and genetic algorithms), and API interfaces with direct access to the Internet.

5) *Context Integration and Generation*: The Agent integrates the retrieved context with the user’s original intent for downstream modeling. For instance, it may issue a prompt such as: “Based on the retrieved knowledge, please generate a draft design of a multi-UAV trajectory optimization system. Include the definition of state space, action space, constraint condition, reward function, and recommend a suitable MAIT architecture and T-GRPO method.” This then triggers the LLM to perform generation and reasoning, resulting in an initial output, such as:

- *State space*: Position, battery level, task load, etc.
- *Action space*: Discrete motion primitives.
- *Reward function*: The total UAV energy consumption, including flight energy, hovering energy, and wireless charging energy.
- *Model architecture*: A Transformer with 10 encoder layers, comprising 50% Mamba and 50% attention layers.
- *Training method*: Discrete trajectory training with 8 independent rollouts.
- *Task constraints*: The total energy consumption must not exceed the UAV’s maximum energy capacity.

6) *Self-Verification and Evaluation*: The Agent semantically verifies the generated output by checking whether all essential parameters are included and whether the proposed design satisfies task constraints. For example, the Agent may identify the absence of resource constraints.

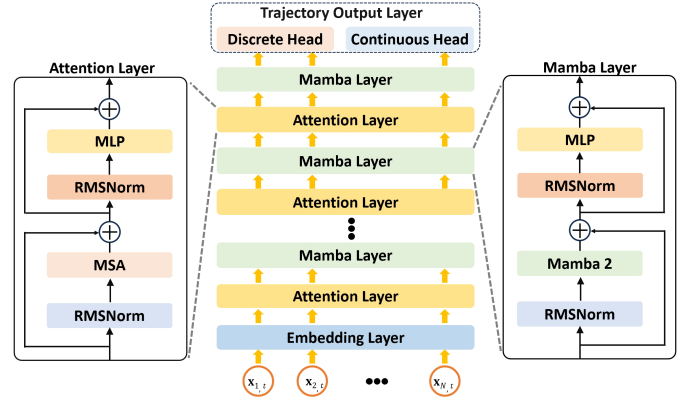


Fig. 2. Architecture of the MAIT model.

7) *Multi-Round Retrieval and Generation*: If the current output does not meet task objectives or lacks sufficient contextual grounding, the Agent restarts the query rewriting and retrieval cycle for another round of interaction. Otherwise, the finalized solution is returned to the MAIT model and the T-GRPO algorithm.

In summary, Agentic RAG overcomes the static and passive limitations of conventional RAG systems by introducing an LLM-based agent equipped with autonomous reasoning and feedback-control capabilities [25]. This enhancement substantially improves the adaptability, accuracy, and robustness of LLMs in UAV scenarios involving complex modeling, decision support, and interactive intelligent generation.

B. Mamba-Attention Integrated Transformer

To enhance both the modeling capacity and computational efficiency of multi-UAV trajectory optimization, we develop a hybrid framework that integrates Mamba with Transformer self-attention. In this design, the attention module captures global dependencies and interaction patterns, whereas the Mamba module models mobility continuity and the temporal evolution of resource constraints, reducing the time and memory complexity of long-sequence processing from quadratic to linear in sequence length. This complementary integration enables faster and more accurate trajectory-planning decisions and improves generalization and robustness in highly dynamic environments.

The MAIT model is composed of an embedding layer, attention layers, Mamba layers, and a trajectory output layer. Within the attention and Mamba layers, there are also Multi-Layer Perceptron (MLP) sublayers and Root Mean Square Normalization (RMSNorm) sublayers. The MLP sublayer enhances the representation capacity through linear transformations, while the RMSNorm sublayer normalizes the data distribution and stabilizes the training dynamics [26]. The structure of the MAIT model is illustrated in Fig. 2. The working process of the MAIT model can be described as follows:

1) *Embedding Layer*: In the MAIT model, the efficient representation of input states is critical for capturing spatiotemporal dependencies and modeling collaborative behaviors among multiple UAVs. Unlike the embedding layer in

conventional Transformers—which relies on token embeddings derived from a discrete vocabulary and fixed positional encodings for natural language inputs—the MAIT model processes continuous, heterogeneous, and physically constrained numerical states inherent to UAV trajectory planning. Such high-dimensional physical features cannot be represented through a standard lookup-table embedding, nor can their continuous temporal dynamics be effectively captured by sinusoidal positional encodings.

To address this, we design a structured embedding layer that maps the raw observation state of the UAV system at each time step into a unified high-dimensional representation, which serves as the input to the MAIT model. The input embedding layer consists of three components: (1) State embedding, (2) Temporal encoding, and (3) Identity embedding. The overall representation of the i -th input (either a ground device or a UAV) in the embedding layer is formulated as:

$$\mathbf{h}_i^0 = \mathbf{x}_{i,t} \mathbf{W}_s + \mathbf{p}_t + \mathbf{b}_i \quad (1)$$

where $\mathbf{x}_{i,t} \in \mathbb{R}^{d_{in}}$ denotes the raw state vector of the i -th input at time step t , including features such as position, energy level, task load, and environmental observations. $\mathbf{W}_s \in \mathbb{R}^{N \times d_{in}}$ is a linear projection matrix that maps the raw state into the dimensionality of the Transformer representations d . $\mathbf{p}_t \in \mathbb{R}^d$ is the temporal encoding vector for time step t , providing the necessary sequential information for sequence modeling, which is learnable. $\mathbf{b}_i \in \mathbb{R}^d$ is the identity embedding of the i -th input, encoding its uniqueness in the task, d is the dimensionality of the Transformer representations.

This embedding layer offers several advantages. First, the state embedding ensures unified alignment of input features from different attribute spaces, facilitating effective sequence modeling by the MAIT. Second, the temporal encoding equips the model with the capability to capture the sequential dynamics of trajectory evolution. Finally, the identity embedding explicitly models the heterogeneity and parallelism of ground devices and UAVs, providing a structural foundation for subsequent interaction-based attention mechanisms. Therefore, this embedding layer not only enhances the expressive power of the input representations but also provides robust structural priors to support multi-UAV trajectory planning tasks.

2) *Attention Layer*: The l -th attention layer of the MAIT maps the features of the i -th input $\mathbf{h}_i^{l-1}$ to query $\mathbf{Q}_i$, key $\mathbf{K}_i$ and value $\mathbf{V}_i$ with weights $\mathbf{W}^q, \mathbf{W}^k, \mathbf{W}^v$, respectively. The equation is given as

$$\mathbf{Q}_i = \mathbf{W}^q \mathbf{h}_i^{l-1}, \quad \mathbf{K}_i = \mathbf{W}^k \mathbf{h}_i^{l-1}, \quad \mathbf{V}_i = \mathbf{W}^v \mathbf{h}_i^{l-1} \quad (2)$$

where $\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i$ represent the query, key and value of the i -th input, respectively. $\mathbf{W}^q, \mathbf{W}^k, \mathbf{W}^v$ are learnable matrices that map from $\mathbf{h}_i^{l-1}$ to $\mathbf{Q}_i, \mathbf{K}_i$ and $\mathbf{V}_i$. Using the attention mechanism, one input node may receive information from the other input node. Hence, the compatibility of the i -th input node and the j -th input node is denoted as δ_{ij} , which is given as follows:

$$\delta_{ij} = \frac{\mathbf{Q}_i^T \mathbf{K}_j}{\sqrt{d_k}} \quad (3)$$

where d_k is the dimension of the key $\mathbf{K}_i$. Here, the purpose of dividing by $\sqrt{d_k}$ is to scale the compatibility δ_{ij} to avoid poor

performance of self-attention [26]. Then, the self-attention feature of the i -th input node is given as

$$\mathbf{Z}_i = \sum_{j=1}^N \text{Softmax}(\delta_{ij}) \mathbf{V}_j \quad (4)$$

where $\mathbf{V}_j$ represents the value of the j -th input node, and the Softmax function can normalize the output and preserve the data characteristics.

The Multi-head Self-Attention (MSA) mechanism allows the attention layers to jointly attend to information from different representation subspaces at different input nodes [26]. In MSA, we have multiple queries $[\mathbf{Q}_{i,1}, \dots, \mathbf{Q}_{i,H}]$, keys $[\mathbf{K}_{i,1}, \dots, \mathbf{K}_{i,H}]$ and values $[\mathbf{V}_{i,1}, \dots, \mathbf{V}_{i,H}]$ for H heads. Then, the H -head self-attention feature $[\mathbf{Z}_{i,1}, \dots, \mathbf{Z}_{i,H}]$ of the i -th input node is calculated by equations (2)-(4), respectively. Next, we calculate the MSA feature of the i -th input node at the l -th self-attention layer as

$$\mathbf{h}_i^l = \mathbf{W}^z [\mathbf{Z}_{i,1}, \mathbf{Z}_{i,2}, \dots, \mathbf{Z}_{i,H}] \quad (5)$$

where $\mathbf{W}^z$ is the learnable weight to merge the H -head self-attention features.

3) *Mamba Layer*: Mamba-2 is an improved SSM-based neural architecture designed to efficiently model long-range dependencies in sequential data [27]. Unlike the attention mechanism, which relies on pairwise token interactions, Mamba-2 formulates sequence modeling as continuous-time state-space dynamics, enabling linear time complexity with respect to sequence length and hardware-efficient implementation. It is particularly well-suited for long-sequence processing, low-latency inference, and memory-constrained environments.

The fundamental principle of Mamba-2 is to model the output $y(t)$ as the solution of a linear dynamical system driven by the input signal $x(t)$ via a hidden state $h(t)$. The system is governed by the following equations:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t) \quad (6)$$

$$y(t) = \mathbf{C}h(t) \quad (7)$$

where $h'(t)$ is the derivative of $h(t)$ with respect to time t , $\mathbf{A}$ is the learnable state matrix, $\mathbf{B}$ is the learnable input matrix, and $\mathbf{C}$ is the learnable output matrix.

Mamba-2 can be regarded as a discrete version of continuous SSMs. The discretization formulas are represented as follows:

$$\bar{\mathbf{A}} = \exp(\Delta \mathbf{A}) \quad (8)$$

$$\bar{\mathbf{B}} = (\Delta \mathbf{A})^{-1} (\exp(\Delta \mathbf{A}) - \mathbf{I}) \cdot \Delta \mathbf{B} \quad (9)$$

where $\bar{\mathbf{A}}$ is the discretized learnable state matrix, $\bar{\mathbf{B}}$ is the discretized learnable input matrix, Δ is the discretization time step parameter, $\exp(\cdot)$ is the matrix exponential function, and $\mathbf{I}$ is the identity matrix.

Building upon this foundation, the continuous system equations are transformed into their discrete counterparts as follows:

$$h_t = \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t \quad (10)$$

$$y_t = \mathbf{C}h_t \quad (11)$$

where h_t , h_{t-1} , x_t , and y_t are the current hidden state, previous hidden state, current input, and current output at discrete time step t , respectively.

In addition, for an input sequence with size T , a global convolution with kernel $\bar{\mathbf{K}}$ can be applied for computing the output of Eqs. (10)-(11) as follows:

$$\bar{\mathbf{K}} = (\mathbf{C}\bar{\mathbf{B}}, \mathbf{C}\bar{\mathbf{A}}\bar{\mathbf{B}}, \dots, \mathbf{C}\bar{\mathbf{A}}^{T-1}\bar{\mathbf{B}}) \quad (12)$$

$$\mathbf{y} = \mathbf{x} * \bar{\mathbf{K}}, \quad (13)$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the input and output matrices of the Mamba layer, respectively.

Mamba-2 layers capture contextual information via selective state propagation, removing the need for the explicit multi-head design commonly used in Transformers. To evaluate MAIT's scalability in multi-UAV swarm settings, we analyze its computational complexity. The multi-head self-attention in the attention branch has quadratic complexity, $O(N^2d)$, so both computation and memory costs grow rapidly with the number of UAVs N . In contrast, the Mamba branch adopts a selective state-space model with linear complexity, $O(Nd)$, enabling efficient long-sequence processing without forming attention matrices. MAIT further adapts the ratio of attention to Mamba layers to the swarm size: attention layers dominate in small-to-medium swarms to strengthen global coordination, whereas Mamba layers dominate in large-scale scenarios to maintain efficient inference. As a result, MAIT achieves strong scalability across UAV networks of varying sizes.

4) *Trajectory Output Layer*: In UAV trajectory optimization, the trajectory output layer acts as the terminal module of the Transformer, decoding the high-dimensional temporal representations produced by MAIT into executable control signals and thus bridging the model representation space and the real flight control space. Designing a structurally clear, functionally complete, and task-generalizable output mechanism that supports both discrete and continuous planning modes is therefore essential for improving adaptability and generalization across diverse UAV scenarios.

The trajectory output layer is designed to support the simultaneous modeling and parallel output for multiple UAVs, aiming to handle both discrete trajectory planning and continuous trajectory control tasks. Its fundamental function is to decode the UAV trajectories from the complete feature sequence representations generated by the Transformer with L layers, denoted as $\mathbf{H}_t = \{\mathbf{h}_{1,t}^L, \mathbf{h}_{2,t}^L, \dots, \mathbf{h}_{N,t}^L\} \in \mathbb{R}^{N \times d}$ into the corresponding action outputs $\mathbf{A}_t = \{\mathbf{a}_{1,t}, \mathbf{a}_{2,t}, \dots, \mathbf{a}_{M,t}\} \in \mathbb{R}^{M \times K}$, where N is the number of input nodes, d is the dimensionality of the Transformer representations, M is the number of UAVs, and K is the dimensionality of the task-related output space.

The discrete trajectory head is designed to handle trajectory planning tasks where the set of target nodes consists of a finite number of stop points. For example, in scenarios involving data collection from fixed IoT devices, the trajectory head enables each UAV to select the most appropriate target from a predefined set of stop points $\mathcal{V} = \{v_1, \dots, v_K\}$. The output is a probability distribution over the candidate stop points for

each UAV at the t -th time step:

$$p_j(t) = \text{Softmax}(\mathbf{W}^Q \mathbf{H}_t) \mathbf{M} \quad (14)$$

where $\mathbf{W}^Q$ is the learnable projection matrix, and $\mathbf{M}$ is the mask matrix. Initially, we set $\mathbf{M} = [1, \dots, 1]$ at the beginning of trajectory planning. The length of $\mathbf{M}$ is K . $\mathbf{M}(i) = 1$ means the UAV can fly to the i -th stop point, while $\mathbf{M}(i) = 0$ means that the stop point is currently inaccessible or restricted.

In each time step, each UAV selects the stop point with max probability in $\mathbf{P}(t) = \{p_j(t), j \in \mathcal{V}\}$ as the next stop point, so one has

$$\mathbf{A}_t = \text{argmax}(\mathbf{P}(t)) \quad (15)$$

where $\text{argmax}(\cdot)$ returns the index of the stop point with the max probability.

The continuous trajectory head is applied to scenarios that require the output of continuous action variables, such as position coordinates, orientation vectors, and similar representations. Typical applications include smooth trajectory following or dynamic navigation. Specifically, the predicted action $\mathbf{a}_{i,t}$ can take various formats depending on the control task, such as: $(\Delta x, \Delta y)$ representing position increments, (d, θ) representing distance and heading angle, or (x, y) representing the absolute target position, so one has

$$\mathbf{A}_t = \text{Tanh}(\mathbf{W}^Q \mathbf{H}_t) \mathbf{S} \quad (16)$$

where $\mathbf{W}^Q$ is the learnable projection matrix, and $\mathbf{S}$ is the scale matrix, which is a task-specific physical scaling factor used to adjust the predicted output to a physically meaningful range.

The proposed output layer provides several key functionalities that make it well suited for multi-UAV trajectory planning. First, it enables parallel modeling of multiple UAVs through a unified state representation and simultaneous output generation, thereby improving the capture of cooperative behaviors. Second, by employing separate output heads for discrete selection and continuous control, the architecture adapts to diverse task formats and supports seamless integration across heterogeneous mission requirements. Third, differentiated activation functions are adopted for distinct action spaces: Softmax maps the outputs to a probability distribution for discrete target selection, while Tanh constrains control variables within physically feasible ranges for continuous command generation. This design achieves a balance between output precision and control stability and promotes more robust policy learning in RL-based optimization. The workflow of the MAIT model is presented in Algorithm 1.

C. Trajectory-Group Relative Policy Optimization

Conventional RL methods for UAV trajectory planning primarily rely on AC-type algorithms, such as A3C and DDPG. However, these methods face significant challenges during training. On one hand, AC algorithms require the simultaneous maintenance of multiple policy networks and value networks. The value network typically has a scale comparable to the policy network, resulting in considerable computational and memory overhead. On the other hand, in highly dynamic

Algorithm 1 Mamba-Attention Integrated Transformer

Input: Raw state observations at each time step.
Output: Planned trajectories for all UAVs.

```

1: Initialize the MAIT model parameters
2: for each time step  $t$  do
3:   for each input  $i$  do
4:     Project raw state to embedding features by Eq. (1).
5:   end for
6:   for each encoder layer  $l$  do
7:     if layer type is attention layer then
8:       Compute MSA by Eqs. (2)–(5).
9:     else
10:      Compute Mamba-2-based state-space transformation by Eqs. (12)–(13).
11:    end if
12:    Apply the RMSNorm and MLP transformation.
13:  end for
14:  if the trajectory is discrete then
15:    Use Softmax function to predict probabilities over candidate waypoints and select next target by Eqs. (14)–(15).
16:  else
17:    Use Tanh function to predict continuous control variables by Eq. (16).
18:  end if
19: end for
20: return Final planned trajectories for all UAVs.

```

eliminates the need for a value network, significantly reducing computational and memory costs during RL. Moreover, T-GRPO estimates advantages based on the relative quality among multiple generated outcomes under the same state, rather than relying on predicted rewards from a value network, thereby mitigating estimation bias. In addition, T-GRPO introduces an optimized regularization strategy for Kullback–Leibler (KL) divergence, further enhancing training stability and convergence efficiency. In the following, we present a detailed formulation of the T-GRPO training procedure, as illustrated in Fig. 3, beginning with the definitions of the state, action, and reward.

1) *State, Action and Reward*: To systematically formalize the multi-UAV cooperative trajectory planning task, the definitions are given as follows:

- **State space** s_t : The system state at time step t consists of the joint states of all input nodes, expressed as

$$s_t = \{s_{1,t}, s_{2,t}, \dots, s_{N,t}\}, \quad s_{i,t} \in \mathbb{R}^{d_m} \quad (17)$$

where $s_{i,t}$ denotes the state vector of the i -th input node at time t .

- **Action space** a_t : Depending on the specific task formulation, the action space is defined as follows:

$$a_t = \{a_{1,t}, a_{2,t}, \dots, a_{M,t}\}, \quad a_{i,t} \in \mathbb{R}^K \quad (18)$$

For discrete planning tasks, each UAV selects its next target from a predefined set of candidate stop points denoted by $\mathcal{V} = \{v_1, \dots, v_K\}$. In the case of continuous control, each UAV instead generates a continuous control vector to adjust its motion parameters. Hence, the joint RL trajectory of the system over a planning horizon is represented by the sequence of states and actions:

$$\tau = \{s_t, a_t\}_{t=0}^T \quad (19)$$

where T is the time horizon of the planning process.

- **Reward function** $r(s_t, a_t)$: The reward function reflects the instantaneous system benefit, such as negative energy consumption, coverage gain, path smoothness, or collision avoidance penalties. The cumulative discounted return over the entire RL trajectory is defined as

$$R(\tau) = \sum_{t=0}^T \gamma^t r(s_t, a_t) \quad (20)$$

where $\gamma \in [0, 1]$ is the discount factor, and $r(\cdot)$ is the task-defined reward function.

2) *Group Sampling*: To sufficiently explore the policy space and mitigate the bias introduced by a single trajectory, a multi-trajectory sampling strategy is employed. In each round of policy updates, the algorithm starts from a given initial state s_0 and performs G independent rollouts under the current policy π_θ , thereby generating G complete sequences of state-action pairs. Each rollout fully records the transitions and action selections from the initial state to the terminal step, which can be formalized as

$$\tau^{(j)} = \{s_0, a_0^{(j)}, s_1^{(j)}, a_1^{(j)}, \dots, s_T^{(j)}, a_T^{(j)}\}, \quad j = 1, 2, \dots, G. \quad (21)$$

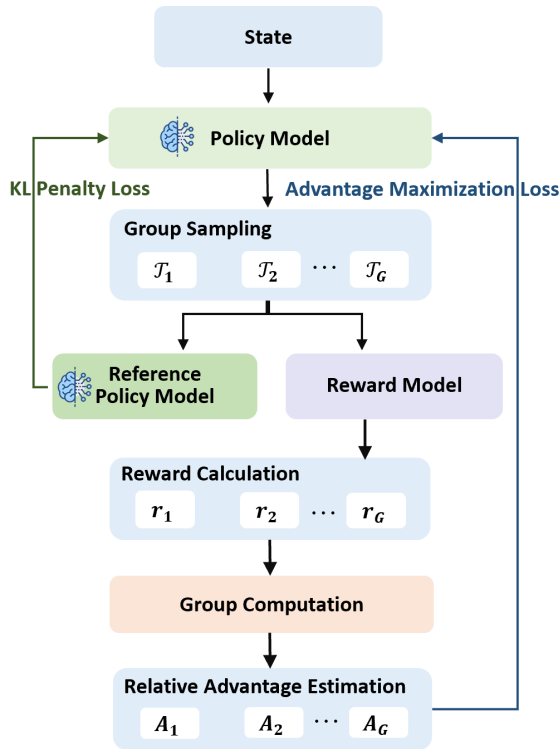


Fig. 3. Training pipeline of T-GRPO.

environments, the value of intermediate states is difficult to estimate accurately, leading to instability during training.

To address the aforementioned challenges, we propose a novel T-GRPO method [28]. This method completely

This multi-trajectory parallel sampling scheme enhances the diversity of policy behavior representation and ensures the robustness and sufficiency of subsequent advantage estimation.

3) *Reward Calculation*: The cumulative return of each trajectory is obtained by summing the discounted instantaneous rewards over time. Specifically, with a discount factor $\gamma \in [0, 1]$, the cumulative reward of the j -th trajectory is defined as

$$R^{(j)} = \sum_{t=0}^T \gamma^t r(s_t^{(j)}, a_t^{(j)}). \quad (22)$$

This cumulative reward captures the overall performance of UAVs during the planning horizon.

4) *Relative Advantage Estimation*: To avoid bias from absolute value estimation, T-GRPO adopts a normalization-based relative advantage across the group. The mean μ_R and standard deviation σ_R of the group returns are calculated as

$$\mu_R = \frac{1}{G} \sum_{j=1}^G R^{(j)} \quad (23)$$

$$\sigma_R = \sqrt{\frac{1}{G} \sum_{j=1}^G (R^{(j)} - \mu_R)^2 + \delta} \quad (24)$$

where δ is a small positive constant for numerical stability. The relative advantage of the j -th trajectory is then expressed as

$$A^{(j)} = \frac{R^{(j)} - \mu_R}{\sigma_R}. \quad (25)$$

This mechanism improves the stability of policy optimization and mitigates the impact of varying reward scales during updates.

5) *Policy Optimization*: In the T-GRPO training process, for each trajectory and each decision step t , the probability ratio between the current policy and the old policy is first computed to measure the relative change in action distributions, formulated as

$$r_t^{(j)} = \frac{\pi_\theta(a_{i,t}^{(j)} | s_t)}{\pi_{\theta_{\text{old}}}(a_{i,t}^{(j)} | s_t)} \quad (26)$$

where π_θ and $\pi_{\theta_{\text{old}}}$ denote the probability distributions of the current and previous policies, respectively.

To guarantee stable policy updates, the T-GRPO framework adopts a clipped objective inspired by PPO [29], constraining the excessive amplification of relative advantage. The GRPO loss at step t on the j -th trajectory is defined as

$$L_t^{(j)} = \min(r_t^{(j)} A^{(j)}, \text{clip}(r_t^{(j)}, 1 - \epsilon, 1 + \epsilon) A^{(j)}) \quad (27)$$

where $\text{clip}(\cdot)$ is a clipping function, which serves to constrain the policy probability ratio $r_t^{(j)}$ in a controllable range, thereby preventing overly aggressive updates that may cause instability, and ϵ is a hyperparameter controlling the clipping range.

The overall policy objective is then expressed as maximizing the average across all sampled trajectories and time steps:

$$\mathcal{L}(\theta) = -\frac{1}{G} \sum_{j=1}^G \sum_{t=0}^T L_t^{(j)}. \quad (28)$$

Algorithm 2 Trajectory-Group Relative Policy Optimization

Input: Initial policy π_θ , initial state s_0 , total rollouts G , time horizon T .
Output: Optimized policy parameters θ .

- 1: **while** not converged **do**
- 2: Sample G stochastic trajectories starting from s_0 under π_θ .
- 3: **for** each trajectory $j = 1, \dots, G$ **do**
- 4: Compute cumulative reward by Eq. (22).
- 5: **end for**
- 6: Compute group mean and standard deviation by Eqs. (23)-(24).
- 7: **for** each trajectory $j = 1, \dots, G$ **do**
- 8: Calculate relative advantage by Eq. (25).
- 9: **end for**
- 10: Compute the overall policy objective by Eqs. (26)-(28).
- 11: Compute the total loss $\mathcal{L}_{\text{total}}$ with KL regularization in Eq. (29).
- 12: Update the policy π_θ by minimizing $\mathcal{L}_{\text{total}}$.
- 13: **end while**
- 14: **return** Optimized policy parameters θ .

Furthermore, to regulate the magnitude of policy shifts and prevent excessive divergence during updates, a KL penalty term is introduced, yielding the total objective function as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}(\theta) + \lambda \frac{1}{G} \sum_{j=1}^G \sum_{t=0}^T D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot | s_t) \| \pi_\theta(\cdot | s_t)) \quad (29)$$

where λ is a regularization coefficient. This composite objective effectively balances policy improvement, update stability, and group-relative advantage, thereby enhancing convergence speed and robustness in multi-UAV trajectory planning tasks. The KL penalty term is introduced to explicitly constrain the divergence between the new and old policies, ensuring stable policy updates in the absence of a value network and preventing abrupt policy shifts in mixed discrete-continuous action spaces. Compared with PPO, which restricts updates only implicitly through ratio clipping, KL regularization provides more precise control over policy distribution changes, leading to smoother, more coordinated, and better-converging policy learning in large-scale multi-UAV scenarios. The workflow of the T-GRPO algorithm is presented in Algorithm 2.

IV. CASE STUDY 1. CONTINUOUS TRAJECTORY PLANNING

A. Simulation Parameter Setting

In this case study, we investigate the trajectory planning problem in a UAV-assisted Mobile Edge Computing (MEC) system, adopting a system framework similar to that proposed in [10]. The simulation area covers 400×400 meters, with 100 UEs and four UAVs. Each UE generates one task in each time slot, where the communication data size of each task is selected between 10 and 50 KB, and the computational resource demand is chosen within the range of 2×10^9 to

TABLE I
PARAMETER SETTINGS FOR CASE STUDY 1

Parameter	Value
Number of time slots T	60
Number of UEs N	100
Number of UAVs M	4
Maximal number of tasks V^{max}	30
Maximal distance d^{max}	30 m
Maximal time duration T^{max}	1 s
Maximal azimuth angle θ^{max}	$\pi/4$
Altitude value of the UAV $Z_j(0)$	75 m
Channel power gain g_0	1.42×10^{-4}
Transmitting power P^{Tr}	0.1 W
Channel bandwidth B	10 MHz
Noise power σ^2	-90 dBm
Maximal energy of each UAV e^{max}	10^6 J
Effective switched capacitance k_i	10^{-28}
Maximal computation resource f^{max}	100 GHz

2×10^{10} CPU cycles. The remaining parameters of this case study are detailed in Table I [10].

B. Agentic RAG Analysis

We designed an Agentic RAG system based on DeepSeek R1, harnessing its powerful reasoning capabilities to better support agent-driven optimization of the RAG framework. We adopted Qanything as the underlying RAG system [30] and enriched its knowledge base with a substantial amount of literature related to UAV trajectory planning, to enhance the LLM's understanding of UAV mission requirements [10].

To systematically evaluate the quality of task formulations generated by different RAG systems, we design an assessment scheme consisting of four key metrics. (1) State Correctness: This metric examines whether the generated state space includes all essential variables and whether each variable is assigned an appropriate type. (2) Action Correctness: This metric evaluates whether the constructed action space contains all required elements and whether their types are correctly specified. (3) Constraint Coverage: This metric assesses whether the system correctly identifies and incorporates the primary task constraints. (4) Reward Consistency: This metric measures the degree to which the designed reward function aligns with the underlying task objectives. Model hallucination is used to assess whether the generated content introduces fictitious variables, incorrect constraints, or invalid reward definitions that are inconsistent with the knowledge base or the underlying system model. Accordingly, we define an Anti-Hallucination Rate to evaluate the impact of hallucination on the final task formulation. This metric measures the proportion of task elements that remain factually correct, constraint-compliant, and semantically valid, with each of the four evaluation dimensions contributing 25% to the final score. A higher Anti-Hallucination Rate indicates greater reliability and correctness in the generated task modeling results.

Fig. 4 compares the Anti-Hallucination Rate of conventional RAG and Agentic RAG across three representative task cases. As shown in the figure, Agentic RAG consistently achieves

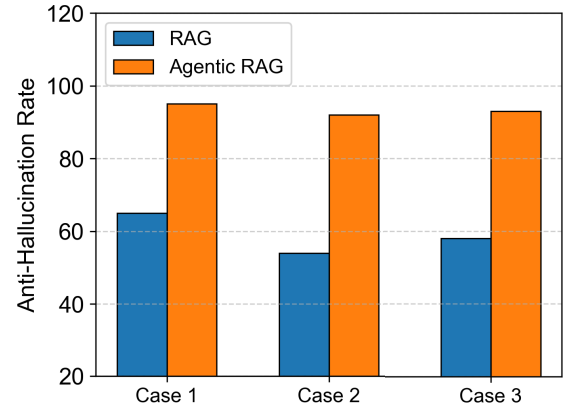


Fig. 4. Anti-hallucination rate comparison between RAG and Agentic RAG across three case studies.

substantially higher scores than the conventional RAG baseline in all cases. In Case study 1 (Section IV), Agentic RAG improves the Anti-Hallucination Rate from approximately 65% to over 95%. Similar improvements are observed in Case study 2 (Section V) and Case study 3 (Section VI), where Agentic RAG achieves around 93%–94%, significantly surpassing the 54%–58% range of conventional RAG. These results clearly demonstrate that the Agentic RAG framework substantially reduces hallucination, enhances factual and constraint consistency, and provides more reliable task modeling performance across diverse UAV trajectory planning scenarios.

After multiple rounds of retrieval, the LLM provided the following conclusions:

1) *State Space*: The state space consists of the positions of the UEs, along with the communication data size and computational resource demand of each task.

2) *Action Space*: The action space is defined as the flight distance and angle of each UAV. It is assumed here that the task offloading matrix and the resource allocation matrix can be addressed using existing convex optimization or matching algorithms [10], which are available in the tool library of the Agentic RAG framework.

3) *Reward Function*: The reward function is designed to minimize the total energy consumption of all UEs, including both task offloading energy and local execution energy.

4) *Model Architecture*: The recommended model architecture consists of 10 encoder layers, comprising 70% attention layers and 30% Mamba layers.

5) *Training Method*: The training method employs a continuous trajectory optimization strategy with 4 independent rollouts to train the model.

6) *Task Constraints*: The task constraints include binary offloading decisions for each UE's tasks, the maximum number of offloaded tasks that can be executed by a single UAV, UAV flight range limitations, UAV coverage constraints, quality-of-service requirements for task execution, and so on.

C. Comparison of Different Transformer Models

This experiment is presented to compare the performance and efficiency of different network architectures. The MAIT

TABLE II
COMPARISON OF DIFFERENT TRANSFORMER MODELS

Method	Energy Consumption (J)	Inference Time (s)
Transformer [26]	376.38	2.25
Linformer [31]	398.65	1.54
Performer [32]	338.43	1.66
MAIT 1	317.64	1.42
MAIT 2	340.57	1.15

TABLE III
COMPARISON OF DIFFERENT TRAINING METHODS

Method	Energy Consumption (J)	Training Time (s)
AC [35]	493.21	13104.46
DDPG [36]	457.79	10778.73
SAC [37]	349.04	9905.53
T-GRPO	314.53	10404.27

model is compared with the following attention mechanisms: Classic Transformer [26], Linear Transformer (Linformer) [31], and Performer [32]. We tested two different MAIT models: the MAIT model (MAIT 1) recommended by the Agentic RAG system incorporated 70% attention layers, while another MAIT model (MAIT 2) incorporated 30% attention layers. The energy consumption and inference time of different Transformer models are shown in Table II.

As shown in Table II, it is evident that the MAIT 1 model recommended by the Agentic RAG system achieves the lowest energy consumption. The MAIT 2 model, which incorporates more Mamba layers, achieves the shortest inference time but at the cost of slightly increased energy consumption. The Performer model demonstrates lower energy consumption than MAIT 2, but with a longer inference time. Other Transformer-based models do not exhibit advantages in either performance or inference time compared to the MAIT models.

The low energy consumption can be explained by two factors: (1) the Agentic RAG framework adaptively determines the MAIT architecture, including the ratio of attention and Mamba layers, according to the user scale, thus enabling efficient and accurate feature extraction; (2) the hybrid attention-Mamba structure captures both global dependencies and local patterns. The attention layers model contextual relationships, while the Mamba layers preserve sequential consistency and reduce redundant computation.

D. Comparison of Different Training Methods

This experiment evaluates the feasibility of the proposed T-GRPO algorithm by comparing it with AC [35], DDPG [36], and SAC [37]. The energy consumption and training time of these RL methods are reported in Table III. As shown in the table, T-GRPO achieves the lowest energy cost among all methods. This advantage mainly comes from its normalized advantage design, which reduces policy gradient variance and enables more stable optimization of the energy objective. Moreover, T-GRPO adaptively balances exploration and exploitation, allowing it to converge to a more effective

TABLE IV
PARAMETER SETTINGS FOR CASE STUDY 2

Parameter	Value
Number of IoTs N	500
Number of UAVs M	4
Attenuation parameter η^L	0.9
Fly speed of UAV v	10 m/s
Flight Height H^F	20 m
Flight Power P^F	75 W
Bandwidth B	2M Hz
Transmitting power P^T	0.5 W
Data collection power P^C	0.5 W
Hover power P^H	50 W
Noise power σ^2	-110 dBm

energy control policy across diverse state spaces. Therefore, it exhibits more robust and superior training performance.

V. CASE STUDY 2. DISCRETE TRAJECTORY PLANNING

A. Simulation Parameter Setting

In this case study, we examine the trajectory planning of UAVs in a Wireless Power Transfer (WPT)-assisted Internet of Things (IoT) system, adopting a framework similar to that presented in [11]. The study area encompasses a region measuring 1000×1000 meters, in which 500 IoT Devices (IoTDs) are randomly deployed. The amount of data collected by each IoTD is randomly drawn from the interval between 0.2 and 1.5 MB. Then, the detailed parameter settings of the case study are listed in Table IV [11].

B. Agentic RAG Analysis

We incorporated the reference work [11] into the knowledge base and constructed queries for the LLM concerning a similar system. After multiple rounds of retrieval, the LLM provided the following conclusions:

1) *State Space*: The state space consists of the locations of all IoTDs along with their remaining data to be collected.

2) *Action Space*: The action space is defined as selecting the location of the next IoTD to be visited, under the constraint that each IoTD can be visited only once and exclusively by a single UAV.

3) *Reward Function*: The reward function accounts for the total UAV energy consumption, including flight energy, hovering energy, and wireless charging energy.

4) *Model Architecture*: The recommended model architecture consists of 15 encoder layers, comprising 30% attention layers and 70% Mamba layers, with a suggested discrete output head incorporating a masking matrix.

5) *Training Method*: The training method employs a discrete trajectory optimization strategy with 8 independent rollouts to train the model.

6) *Task Constraints*: The task constraints stipulate that, due to the limited data storage and energy resources of UAVs, the total energy consumption must not exceed the UAV's maximum energy capacity, and the total amount of collected data must remain within its maximum storage capacity.

TABLE V
COMPARISON OF DIFFERENT TRAJECTORY PLANNERS

Methods	Energy Consumption (Wh)	Inference Time (s)
GNN [38]	292.861	2.42
GPN [39]	278.88	1.67
ATOM [11]	260.64	2.86
MAIT 1	249.57	2.53
MAIT 2	258.34	1.21

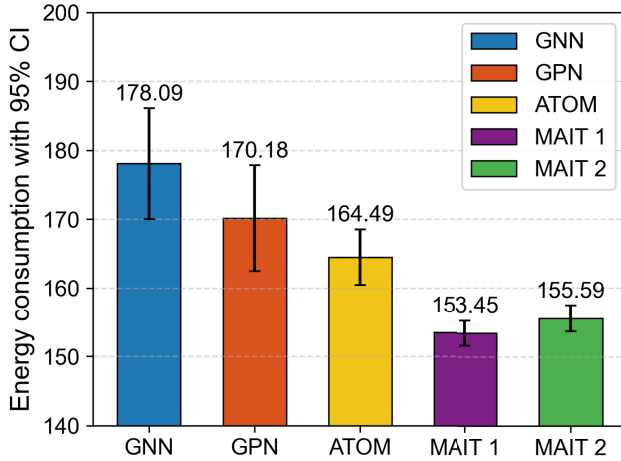


Fig. 5. Average energy consumption of different trajectory planners.

C. Comparison of Different Trajectory Planners

In our experiments, we evaluated two MAIT variants: MAIT 1, a baseline configuration with 70% attention layers, and MAIT 2, a model recommended by the Agentic RAG system that incorporates 30% attention layers. We then compared their overall performance with several discrete trajectory planning algorithms, including Graph Neural Network (GNN) [38], Graph Pointer Network (GPN) [39], and the ATOM network [11], using energy consumption and inference time as evaluation metrics, as summarized in Table V.

As shown in Table V, MAIT 1 achieves the lowest energy consumption, while MAIT 2 attains the shortest inference time among all compared algorithms. This is because the MAIT framework employs a hybrid attention-Mamba mechanism, enabling the encoder to capture both local and global IoT features for more effective representation, while the Mamba structure significantly reduces inference time.

To further evaluate the robustness and statistical reliability of the proposed MAIT models, we conducted an additional experiment involving 200 IoTs. In this setting, each algorithm—MAIT 1, MAIT 2, GNN, GPN, and ATOM—was executed 10 times with different random seeds to account for the stochasticity inherent in learning-based trajectory planning methods. We report the average energy consumption along with the 95% Confidence Interval (CI) for each method in Fig. 5. The results show that both MAIT variants consistently outperform the baseline algorithms, exhibiting significantly lower energy consumption and smaller variance across repeated runs.

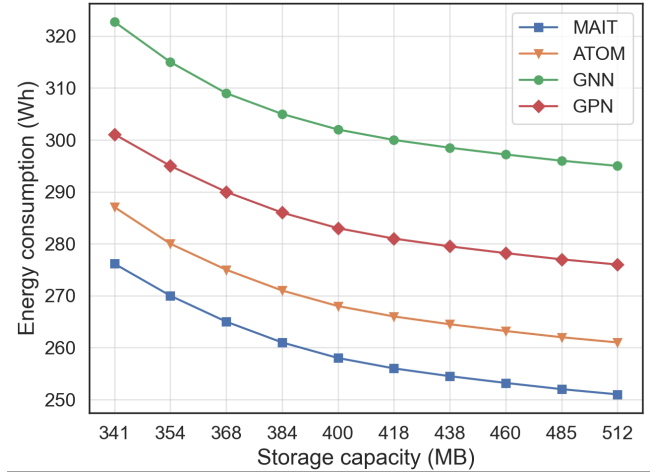


Fig. 6. Energy cost under different UAV data storage capacities.

D. Comparison of Different Constraints

This experiment aims to evaluate the generalization of the MAIT model, and we apply all trajectory designers to scenarios with different UAV data storage capacities. The range of UAV data storage capacity is set from 340 to 512 MB. The simulation results of the energy costs are shown in Fig. 6.

As shown in Fig. 6, GNN yields the highest energy cost, followed by GPN, ATOM, and MAIT. Among all neural network-based methods, MAIT achieves the lowest energy cost. This superiority can be attributed to two main factors: (1) its hybrid self-attention and Mamba mechanisms provide stronger feature extraction capability than competing methods; (2) the proposed T-GRPO better captures and adapts to dynamic factors in UAV-assisted data collection, such as remaining storage and battery capacity.

VI. CASE STUDY 3. FIELD CASE EVALUATION

This section presents a field case design to evaluate the trajectory optimization performance of the ARMAIT framework. The simulation scenario is set in the Dawang Mountain area of Changsha, Hunan Province, China, where 50 IoTs have been deployed for air quality monitoring. Each IoT operates at an independent monitoring frequency, resulting in varying amounts of collected sensory data. In this case, we consider multiple quadrotor UAVs performing fixed-point cruising and data collection. Each UAV is equipped with a limited battery capacity of 2,250 mAh for data collection tasks and utilizes 500 MB of storage space to accommodate the sensory data gathered from the IoTs.

Fig. 7 shows the UAV trajectories generated by the MAIT model, where each triangle denotes an IoT. All UAVs depart from and finally return to the white-marked region. The MAIT model produces six distinct, non-overlapping trajectories. Each UAV sequentially visits its assigned IoTs without crossing any other paths. This property mainly results from the model's global dependency modeling and spatiotemporal sequence constraints. Specifically, the attention module captures global relationships among UAVs, avoiding conflicting path assignments, while the Mamba layers model temporal evolution

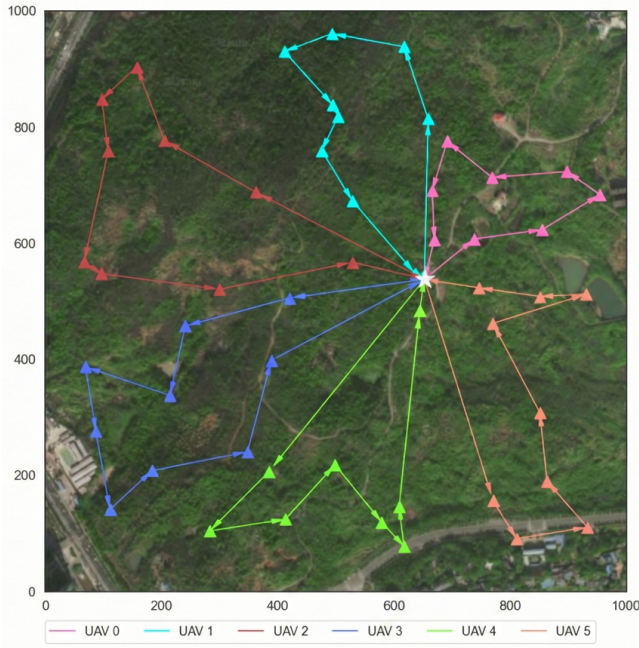


Fig. 7. The generated trajectories of UAVs in the field case.

to maintain trajectory coherence and mutual exclusiveness throughout planning. In addition, the trajectory output head incorporates feasibility masks and constraint-based projection during decoding, filtering out actions that may cause path intersections.

VII. CONCLUSION

This paper proposed the ARMAIT framework, which integrates Agentic RAG for autonomous task understanding, the MAIT model for efficient trajectory feature modeling, and T-GRPO for stable RL optimization across mixed action spaces. ARMAIT enables end-to-end reasoning and trajectory generation for multi-UAV systems. Experimental results show that ARMAIT achieves superior decision-making accuracy and faster convergence than existing approaches and demonstrates strong generalization across multiple case studies. However, the current framework still relies on global system information and focuses only on UAV trajectory optimization, without jointly considering resource scheduling. In the future, we will extend ARMAIT to decentralized and partially observable settings and explore joint optimization of trajectory and resource allocation to enhance overall system performance.

REFERENCES

- [1] J. Wang et al., "Security-aware joint sensing, communication, and computing optimization in low altitude wireless networks," 2025, *arXiv:2511.01451*.
- [2] F. B. Sorbelli, "UAV-based delivery systems: A systematic review, current trends, and research challenges," *ACM J. Auto. Transp. Syst.*, vol. 1, no. 3, pp. 1–40, Sep. 2024.
- [3] J. Wang et al., "Generative AI based secure wireless sensing for ISAC networks," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 5195–5210, 2025.
- [4] Z. Ni, Z. Hua, X. Yang, R. Zhang, S. Wang, and G. Pan, "LLM-aided spectrum-sharing LEO satellite communications," *IEEE J. Sel. Areas Commun.*, vol. 44, pp. 2591–2605, 2026.
- [5] J. Wang et al., "Generative AI enabled robust data augmentation for wireless sensing in ISAC networks," *IEEE J. Sel. Areas Commun.*, vol. 44, pp. 975–990, 2026.
- [6] L. Dong et al., "LAMBO: Large AI model empowered edge intelligence," *IEEE Commun. Mag.*, vol. 63, no. 4, pp. 88–94, Apr. 2025.
- [7] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," 2023, *arXiv:2312.00752*.
- [8] F. Jiang et al., "A comprehensive survey of large AI models for future communications: Foundations, applications, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 28, pp. 4731–4764, 2026.
- [9] F. Jiang et al., "Large language model enhanced multi-agent systems for 6G communications," *IEEE Wireless Commun.*, vol. 31, no. 6, pp. 48–55, Dec. 2024.
- [10] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 10, pp. 3536–3550, Oct. 2022.
- [11] L. Dong, F. Jiang, and Y. Peng, "Attention-based UAV trajectory optimization for wireless power transfer-assisted IoT systems," *IEEE Trans. Ind. Electron.*, vol. 72, no. 8, pp. 8463–8471, Aug. 2025.
- [12] W. Xue, B. Lian, J. Fan, T. Chai, and F. L. Lewis, "Inverse reinforcement learning for trajectory imitation using static output feedback control," *IEEE Trans. Cybern.*, vol. 54, no. 3, pp. 1–13, Mar. 2024.
- [13] Y.-C. Liu and C.-Y. Huang, "DDPG-based adaptive robust tracking control for aerial manipulators with decoupling approach," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 8258–8271, Aug. 2022.
- [14] D. Chen, Q. Qi, Q. Fu, J. Wang, J. Liao, and Z. Han, "Transformer-based reinforcement learning for scalable multi-UAV area coverage," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 8, pp. 10062–10077, Aug. 2024.
- [15] B. Zhu, E. Bedeer, H. H. Nguyen, R. Barton, and Z. Gao, "UAV trajectory planning for AoI-minimal data collection in UAV-aided IoT networks by transformer," *IEEE Trans. Wireless Commun.*, vol. 22, no. 2, pp. 1343–1358, Feb. 2023.
- [16] Z. Feng, D. Wu, M. Huang, and C. Yuen, "Event-driven transformer-based reinforcement learning for trajectory design and channel assignment in multi-AAV assisted communication," *IEEE Trans. Cognit. Commun. Netw.*, vol. 11, no. 6, pp. 4254–4266, Dec. 2025.
- [17] W. Wang, R. Gao, C. Du, and Y. Li, "Enhanced transformer-based UAV-to-UAV tracking using distractor elimination and trajectory prediction," in *Proc. IEEE 17th Int. Conf. Signal Process. (ICSP)*, Oct. 2024, pp. 352–357.
- [18] Z. Xiao, H. Hu, G. Xu, and J. He, "TAME: Temporal audio-based mamba for enhanced drone trajectory estimation and classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2025, pp. 1–5.
- [19] J. Tao, T. Xu, J. Fang, J. Xu, Y. Liu, and Y. Liu, "FDMamba-Net: Feature-decoupled mamba network for efficient flight trajectory prediction," in *Proc. Int. Conf. Intell. Comput.*, 2025, pp. 460–471.
- [20] H. Liang et al., "Label-free long-horizon 3D UAV trajectory prediction via motion-aligned RGB and event cues," 2025, *arXiv:2507.03365*.
- [21] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "UAVs meet agentic AI: A multidomain survey of autonomous aerial intelligence and agentic UAVs," 2025, *arXiv:2506.08045*.
- [22] Z. Zhao et al., "Agentic AI for low-altitude semantic wireless networks: An energy efficient design," 2025, *arXiv:2509.19791*.
- [23] A. Koubaa and K. Gabr, "Agentic UAVs: LLM-driven autonomy with integrated tool-calling and cognitive reasoning," 2025, *arXiv:2509.13352*.
- [24] F. Jiang et al., "Large AI model empowered multimodal semantic communications," *IEEE Commun. Mag.*, vol. 63, no. 1, pp. 76–82, Jan. 2025.
- [25] F. Jiang, C. Pan, K. Wang, P. Michiardi, O. A. Dobre, and M. Debbah, "From large AI models to agentic AI: A tutorial on future intelligent communications," *IEEE J. Sel. Areas Commun.*, vol. 44, pp. 3507–3540, 2026.
- [26] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inform. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.
- [27] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality," 2024, *arXiv:2405.21060*.
- [28] Z. Deng et al., "Exploring DeepSeek: A survey on advances, applications, challenges and future directions," *IEEE/CAA J. Autom. Sinica*, vol. 12, no. 5, pp. 872–893, May 2025.
- [29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

- [30] F. Jiang et al., "CommGPT: A graph and retrieval-augmented multimodal communication foundation model," *IEEE Commun. Mag.*, vol. 64, no. 3, pp. 46–52, Mar. 2026.
- [31] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, "Linformer: Self-attention with linear complexity," 2020, *arXiv:2006.04768*.
- [32] K. Choromanski et al., "Rethinking attention with performers," 2020, *arXiv:2009.14794*.
- [33] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, *arXiv:1508.04025*.
- [34] Y. Li, L. Kaiser, S. Bengio, and S. Si, "Area attention," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3846–3855.
- [35] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, 2002, pp. 1–7.
- [36] H. Tan, "Reinforcement learning with deep deterministic policy gradient," in *Proc. Int. Conf. Artif. Intell., Big Data Algorithms (CAIBDA)*, May 2021, pp. 82–85.
- [37] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
- [38] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
- [39] Q. Ma, S. Ge, D. He, D. Thaker, and I. Drori, "Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning," 2019, *arXiv:1911.04936*.



Kezhi Wang (Senior Member, IEEE) received the Ph.D. degree from the University of Warwick, U.K. He is currently a Professor with the Department of Computer Science, Brunel University of London, U.K. His research interests include wireless communications, mobile edge computing, and machine learning.



Xiaolong Li received the B.E. degree from Harbin Institute of Technology, Harbin, China, in 2003, and the Ph.D. degree from Hunan University, Changsha, China, in 2008. From September 2008 to December 2016, he was with the School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin, China, where he was a Full Professor. Since January 2017, he has been a Full Professor and the Dean of the School of Computer Science, Hunan University of Technology and Business, Changsha. His research interests include

the Industrial Internet of Things (IIoT), edge/fog computing, and artificial intelligence.



Feibo Jiang (Senior Member, IEEE) received the B.S. and M.S. degrees from the School of Physics and Electronics, Hunan Normal University, China, in 2004 and 2007, respectively, and the Ph.D. degree from the School of Geosciences and Info-Physics, Central South University, China, in 2014. He is currently an Associate Professor with Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing, Hunan Normal University. His research interests include large AI model-assisted wireless communications, Agentic

AI-enabled wireless communications, semantic communications, the Internet of Things, and UAV networks. He serves as a (Leading) Guest Editor for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE WIRELESS COMMUNICATIONS.



Ruidong Li received the M.S. degree from Shandong University. He is currently a Senior Engineer with Shandong Yunhai Guochuang Cloud Computing Equipment Industry Innovation Company Ltd., China. He has been selected into Taishan Industrial Leading Talent Program of Shandong Province. His main research interests include computer architecture and information engineering.



Xitao Pan is currently pursuing the M.S. degree with the School of Computer Science and Technology, Hunan Normal University. His main research interests include semantic communication and Agentic AI.



Changhong Wang received the B.S. degree in communication engineering from Zhengzhou University in 2013 and the Ph.D. degree in information and communication engineering from Beijing Institute of Technology in 2020. He is currently a Senior Engineer with Jinan Maiwei Intelligent Technology Company Ltd. His current research interests include VLSI, intelligent system-on-chip, and computer architecture.



Li Dong received the B.S. and M.S. degrees from the School of Physics and Electronics, Hunan Normal University, China, in 2004 and 2007, respectively, and the Ph.D. degree from the School of Geosciences and Info-Physics, Central South University, China, in 2018. She is currently a Professor with Hunan University of Technology and Business, China. Her research interests include machine learning, the Internet of Things, and mobile edge computing.



Cunhua Pan (Senior Member, IEEE) received the Ph.D. degree from Southeast University, China, in 2015. He is currently a Full Professor with Southeast University. He has published more than 200 IEEE journal articles. His main research interests include reconfigurable intelligent surfaces (RIS), AI for wireless, near-field communications and sensing, and integrated sensing and communications. He is a Clarivate Highly Cited Researcher. His articles got more than 19 000 Google Scholar citations with an H-index of 70.