sensors

MDPI

Article

A Goal-Directed Trajectory Planning Using Active Inference in UAV-Assisted Wireless Networks

Ali Krayani [1,2,*], Khalid Khan [1], Lucio Marcenaro [1,2], Mario Marchese [1,2] and Carlo Regazzoni [1,2]

[1] Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture, University of Genoa, 16145 Genoa, Italy; khalid.khan@edu.unige.it (K.K.); lucio.marcenaro@unige.it (L.M.); mario.marchese@unige.it (M.M.); carlo.regazzoni@unige.it (C.R.)

[2] Italian National Inter-University Consortium for Telecommunications (CNIT), 43124 Parma, Italy

* Correspondence: ali.krayani@ieee.org

Abstract: Deploying unmanned aerial vehicles (UAVs) as aerial base stations is an exceptional approach to reinforce terrestrial infrastructure owing to their remarkable flexibility and superior agility. However, it is essential to design their flight trajectory effectively to make the most of UAV-assisted wireless communications. This paper presents a novel method for improving wireless connectivity between UAVs and terrestrial users through effective path planning. This is achieved by developing a goal-directed trajectory planning method using active inference. First, we create a global dictionary using traveling salesman problem with profits (TSPWP) instances executed on various training examples. This dictionary represents the world model and contains letters representing available hotspots, tokens representing local paths, and words depicting complete trajectories and hotspot order. By using this world model, the UAV can understand the TSPWP's decision-making grammar and how to use the available letters to form tokens and words at various levels of abstraction and time scales. With this knowledge, the UAV can assess encountered situations and deduce optimal routes based on the belief encoded in the world model. Our proposed method outperforms traditional Q-learning by providing fast, stable, and reliable solutions with good generalization ability.

Keywords: UAVs; wireless networks; trajectory design; AI-enabled radios; active inference; world models; traveling salesman problem

check for updates

Citation: Krayani, A.; Khan, K.; Marcenaro, L.; Marchese, M.; Regazzoni, C A Goal-Directed Trajectory Planning Using Active Inference in UAV-Assisted Wireless Networks. *Sensors* **2023**, *23*, 6873. https://doi.org/10.3390/s23156873

Academic Editor: Omprakash Kaiwartya

Received: 3 July 2023
Revised: 10 July 2023
Accepted: 25 July 2023
Published: 2 August 2023

1. Introduction

In recent years, there has been a significant amount of research interest in unmanned aerial vehicles (UAVs) due to their impressive features, such as their maneuverability, ease of positioning, versatility, and the high likelihood of line-of-sight (LoS) air-to-ground connections [1,2]. UAVs are feasibly exploited to alleviate a wide range of challenges in commercial and civilian sectors [3,4]. It is expected that forthcoming wireless communication networks will need to provide exceptional service to meet the demands of users. This presents difficulties for traditional terrestrial-based communication systems, particularly in hotspot areas with high traffic [5–7]. UAVs have the potential to serve as flying base stations, providing support to the land-based communication infrastructure without the need for costly network construction [8]. In addition, their ability to be easily relocated makes them particularly highly beneficial in the aftermath of natural disasters [9,10]. UAVs can also be deployed as intermediaries between ground-based terminals, improving transmission link performance and enhancing reliability, security, coverage, and throughput [11,12]. As such, UAV-assisted communications are becoming increasingly vital in developing future wireless systems [13–17].

UAV-aided wireless communications possess a distinct advantage owing to the controllable maneuverability of UAVs, which allows for flexible trajectories. This added degree of freedom significantly boosts the system's performance. Therefore, optimizing the UAV's trajectory is an indispensable area of focus in this field, as it is paramount to exploit the potential of UAV-assisted wireless communications fully [18]. Several studies have looked into improving system performance through trajectory design. One study, for example, optimized the trajectory of a UAV to gather received signal strength measurements efficiently and improve the accuracy of spectrum cartography [19]. Another study proposed a method for planning the trajectory of a UAV to provide emergency data uploading for large-scale dynamic networks [20]. Multi-hop relay UAV trajectory planning is also crucial in UAV swarm networks [21]. Joint optimization of the UAV's trajectory and user association was suggested in [22] to maximize total throughput and energy efficiency. Another study examined joint UAV trajectory design and time allocation for aerial data collection in NOMA-IoT networks [23]. In a cluster-based IoT network, joint optimization of the UAV's hovering points and trajectory was studied to achieve minimal age-of-information data collection [24]. Autonomous trajectory planning solutions were proposed in [25] to enable UAVs to navigate complex environments without GPS while fulfilling real-time requirements. Lastly, the trajectory of a UAV was optimized in [26] to minimize propulsion energy and ensure the required sensing resolutions for cellular-aided radar sensing.

Traditional methods rely on optimization mathematical models that require precise information about the system, including the number of users in different areas and network parameters when designing a UAV trajectory. However, this approach may not be feasible in real-world situations due to the constantly changing environment and limited battery life, making it difficult to solve these problems using traditional techniques [27]. On the other hand, artificial intelligence (AI) techniques, such as machine learning (ML) and reinforcement learning (RL), have proven to be effective in addressing challenges related to sequential decision making. By equipping UAVs with AI capabilities (AI-enabled UAVs), they can attain a remarkable level of self-awareness, transforming wireless communications [28]. With AI, UAVs can effectively comprehend the radio environment by discerning and segregating the explanatory factors that are concealed in low-level sensory signals [29]. However, most ML and RL methods are not capable of adjusting to new situations that were not included in their initial training. This limitation in generalizing requires extensive retraining efforts, which can pose challenges for real-time prediction and decision making [30].

When AI-enabled agents sense and interact with their environment, they struggle with structuring the knowledge they gather and making logical decisions based on it. One way to address this is through knowledge representation and reasoning techniques inspired by human problem-solving to handle complex tasks effectively [31]. Causal probabilistic graphical models are a prime example of such techniques, which are highly effective in capturing the hidden patterns in sensory data obtained from the environment. These models also provide a seamless way to integrate sensory data from various sources [32]. By statistically structuring the data, they can describe different levels of abstraction that can be applied across different domains. For instance, when learning a language, one must learn how sounds form words, how words form sentences, and how grammar characterizes a language. At every level, the learning process requires making probabilistic inferences within a structured hypothesis space. Dealing with uncertainty is a common challenge in AI and decision making, as many real-world problems have incomplete or ambiguous information. Probabilistic representation is an effective technique that leverages probability theory to model and reason with uncertainty, enabling AI agents to make better decisions and operate more efficiently [33].

Active inference is a mathematical framework that helps us understand how living organisms interact with their environment [34]. It provides a unified approach to modeling perception, learning, and decision making, aiming to maximize Bayesian model evidence or minimize free energy [35]. Free energy is a crucial concept that empowers agents to systematically assess multiple hypotheses concerning behaviors that can effectively achieve their desired outcomes. Moreover, active inference governs our expectations of the world around us. Specifically, it posits that our brains utilize statistical models to interpret sensory information [36]. By using active inference, we can modify our sensory input to conform to our preconceived notions of the world and rectify any inconsistencies between our expectations and reality. Probabilistic graphical models are used to represent active inference models because they provide a clear visual representation of the model's computational structure and how belief updates can be achieved through message-passing algorithms [37].

Motivated by the previous discussion, we propose a goal-directed trajectory design framework for UAV-assisted wireless networks based on active inference. The proposed approach involves two key computational units. The first unit meticulously analyzes the statistical structure of sensory signals and creates a world model to gain a comprehensive understanding of the environment. World models are a significant aspect of generative AI. They play a pivotal role in the development of intelligent systems. Like humans, AI agents acquire a world model by processing sensorimotor data through interactions with their environment, which serves as a simulator in their brains [38]. The second is the decision-making unit seeking to perform actions minimizing a cost function and generating preferred outcomes. The two components are linked by an active inference process. To create the world model, the UAV was trained to complete various flight missions with different realizations (such as the locations of hotspots and users' access requests) using the conventional traveling salesman problem with profit (TSPWP) [39] with the 2-OPT local search algorithm in an offline manner. The TSPWP instances (trajectories) were turned into graphs and used to build a global dictionary with two sub-dictionaries. The first sub-dictionary represents the hotspots the UAV needs to serve and their order of travel. By contrast, the second sub-dictionary shows the trajectories to follow between two adjacent nodes. The global dictionary consists of letters at multiple levels, tokens, and words. The world model is created by coupling the two sub-dictionaries, constructing a detailed representation of the environment at different hierarchical levels and time scales. The world model is structured in a coupled multi-scale generalized dynamic Bayesian network (C-MGDBN). This model builds upon the single-scale GDBN, which is a statistical model that explains how hidden states drive time series observations. However, unlike the conventional GDBN [40–42], which can only model single-scale data, our enhanced GDBN representation can encode the dynamic rules that generate observations at different temporal resolutions, making it far more versatile than traditional GDBNs. With this superior model, we can simultaneously model a UAV's behavior at different time scales. The decision-making unit relies on active inference to select actions based on the current state of the environment as inferred from the world model. The proposed framework explains how UAVs navigate their surroundings with a goal in mind, choosing actions that minimize unexpected or unusual observations (abnormalities), which are measured by how much they deviate from the expected goal.

The main contributions of this paper can be summarized as follows:

- We developed a global dictionary during training to discover the TSPWP's best strategy for solving different realizations. The dictionary comprises letters representing the available hotspots, tokens representing local paths, and words depicting the complete trajectories and order of hotspots. By studying the dictionary, we can comprehend the decision maker's grammar (i.e., the TSPWP strategy) and how it uses the available letters to form tokens and words.
- We have designed a novel hierarchical representation structuring the acquired knowledge (the global dictionary) in a C-MGDBN to accurately depict the properties of the TSPWP graphs at various levels of abstraction and time scales.
- We tested the proposed method on different scenarios with varying hotspots. Our method outperformed traditional Q-learning by providing fast, stable, and reliable solutions with good generalization ability.

The remainder of the paper is organized as follows. The literature review is presented in Section 2. The system model and problem formulation are presented in Section 3. The proposed goal-directed trajectory design method is explained in Section 4. Section 5 is dedicated to the numerical results and discussion, and finally Section 6 concludes this paper by highlighting future directions.

Notations: Throughout the paper, capital italic letters denote constants, lowercase bold letters denote vectors, and capital boldface letters denote matrices. The shorthand $\mathcal{N}(\mu, \Sigma)$ is used to denote a Gaussian distribution with mean μ and covariance Σ. If $\boldsymbol{X}$ represents a matrix, the element in its ith row and jth column is denoted by x_{ij}, and its ith row vector is represented by $\boldsymbol{x}_i$.

2. Literature Review

Solving the trajectory design problem is a crucial and leading research topic in AI-enabled wireless UAV networks. This problem involves determining the optimal shortest path for a UAV to cover all targeted hotspot zones (nodes) in a dynamic wireless environment while adhering to time and mission completion constraints. This section discusses various techniques proposed in the literature for UAV trajectory design to optimize communication performance efficiently in a flexible wireless environment. These techniques can be categorized as classical and modern optimization algorithms as depicted in Figure 1.

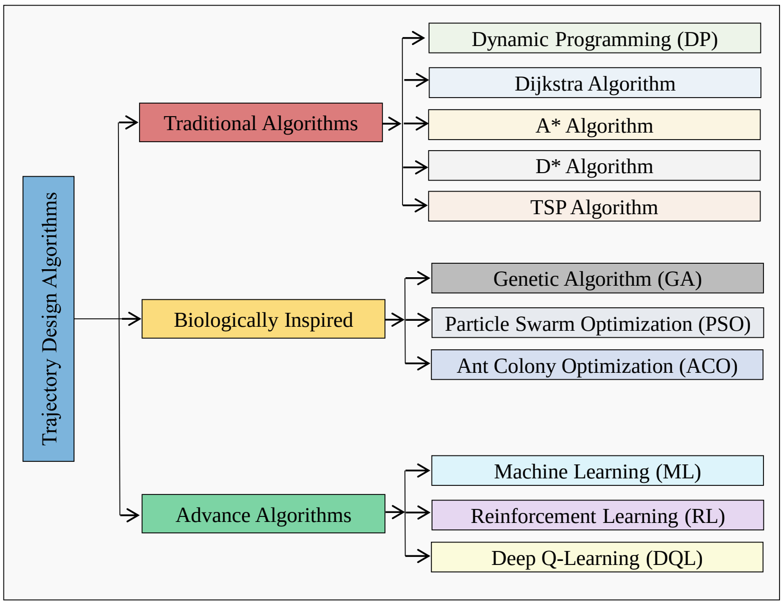


Figure 1. An overview of existing trajectory design algorithms.

In order to meet time constraints for all ground users, a feasible UAV trajectory was proposed in [43] using traditional dynamic programming (DP). However, due to an increase in hovering nodes, it may not align with time constraint criteria and may not be suitable for real-time environments. DP was also used to optimize the UAV trajectory in [44] for accessing multiple wireless sensor nodes (WSNs) and collecting data under time constraints. However, the algorithm was inefficient in recognizing and iterating through repeated grids, requiring high-order gridding for accuracy and resulting in computational complexity. In the study referenced as [45], the problem of the UAV trajectory was formulated as a mixed integer linear program (MILP). The trajectory planning is carried out in discrete time steps, where each step represents the dynamic state of the UAV in the environment. The algorithm is designed for offline planning to ensure a feasible trajectory is available before the UAV performs its tasks. However, this algorithm has limitations as it can easily become stuck due to its blind nature and cannot generate long trajectories in a complex environment. The Dijkstra algorithm proposed in [46] enables UAVs to perform environmental tasks efficiently by using the optimal battery level and reaching the target point in the shortest possible time. However, as the network scale increases, the algorithm takes a long time to provide a solution, making it unsuitable for real-time trajectory planning. The A* algorithm, as discussed in [47], selects suitable node pairs and evaluates the shortest path for UAVs based on feasible node pairs in a known static environment to address this issue. Although the A* algorithm does not provide a continuous path, it ensures that the shortest path is followed in the direction of the targeted node. However, this algorithm is not practical in a dynamic environment. To overcome this, the D* algorithm and its variants, as reviewed in [48], are efficient tools for quick re-planning in a cluttered environment. The D* algorithm updates the cost of new nodes, allowing the use of prior paths instead of re-planning the entire path. However, D* and its variants do not guarantee the quality of the solution in a large dynamic environment.

In order to design an effective path planning model for a UAV, the discrete space-based traveling salesman problem (TSP) [49] is utilized to search for the optimal shortest path for the UAV to travel through a fixed number of cities, with each city only being visited once. The UAV must also return to the starting city within a fixed flight time for battery charging. However, the TSP is an offline algorithm, so when a new city appears in the UAV's path, the cost of the new city is updated from the starting point, resulting in the entire path being replanned from the start to the new end, which is a major drawback. The TSP is a challenging NP-hard problem and can be difficult to solve in polynomial time unless P = NP. Two approaches are available when dealing with the challenging NP-hard problem in TSP. The first involves using heuristics, such as 2-OPT and 3-OPT, to quickly generate near-optimal tours through local improvement algorithms [50]. The second approach is to utilize evolutionary optimization algorithms, such as genetic algorithm (GA), particle swarm optimization (PSO), and ant colony optimization (ACO), which have proven to be effective in minimizing the total distance travelled by the salesman in real-world scenarios [51]. While the GA is a good solution for obtaining an appropriate path for a UAV, it can be relatively slow, making it inefficient for modern path planning problems that require fast performance [52]. On the other hand, the PSO is good at local optimization and can be used in combination with a GA that is good at global optimization [53]. The ACO is also effective in solving the UAV path planning problem, but it requires a significant amount of data to find the optimal solution, has a slow iteration speed, and demands much more simulation time [54]. Therefore, a combination of these algorithms may be necessary to effectively solve the UAV path planning problem.

Reinforcement learning (RL) is a popular AI tool used to tackle complex problems such as trajectory design and sum-rate optimization, which are critical challenges due to the continuous environmental variation over time. Indeed, solving mathematical optimization models is only possible when a priori input data are available or requires too high complexity and computational time. Recent studies [55–57] proposed optimal trajectory design for UAVs using Q-learning to maximize the sum rate [55], increase the QoE of users [56], and enhance the number and fairness of users served [57]. However, Q-learning has a drawback in that the number of states increases exponentially with the number of input variables, and its memory usage also increases sharply. Due to the mobility of both ground and aerial users, the curse of dimensionality can cause Q-learning to fail. As a result, solving the trajectory design problem in a large and highly dynamic environment is a challenging task. A machine learning (ML) technique has been proposed in [58] to optimize the flight path of UAVs in order to meet the needs of ground users within specific zones during set time intervals. Another study in [59] explored a multi-agent Q-learning-based method to design the UAV's flight path based on predicting the movement of the user to maximize the sum rate. Additionally, a meta-learning algorithm was introduced in [60] to optimize the UAV's trajectory while meeting the uncertain and variable service demands of the GUs. However, these reinforcement learning-based solutions can only work in certain environments and are unsuitable for highly dynamic and unpredictable environments. A deep Q-learning (DQL) algorithm was introduced in [61] to enable UAVs to provide network service for ground users in rapidly changing environments autonomously. However, the user mobility model in this algorithm is simple and does not account for ground users moving to different positions multiple times, resulting in inadequate trajectory results for different paths.

In this work, we tackled the challenge of designing a UAV trajectory by treating it as a traveling salesman with profit problem (TSPWP). We leveraged the potent 2-OPT local search algorithm to attain an optimal offline solution. We then converted the resulting TSP instances from diverse examples into graphs and trained the UAV using them. This allowed the UAV to comprehend the properties of the TSP graphs and establish a world model that includes a hierarchical and multi-scale representation. This world model empowers the UAV to figure out the TSP strategy to solve the problem and implicitly discover the objective function. Our approach enables the UAV to deduce optimal routes by utilizing the beliefs encoded in the world model when confronted with a new realization. This significantly helps the UAV ascertain the best solution, even in situations where there are discrepancies between what it knows and what it sees.

3. System Model and Problem Formulation

Consider a UAV-assisted wireless network, as shown in Figure 2, with a single UAV acting as a flying base station (FBS) to serve U ground users (GUs) distributed randomly across a geographical area and requesting uplink data service. GUs that demand the data service are introduced as active users; others are so-called inactive users, as illustrated in Figure 2. It is assumed that the GUs are partitioned into N distinct groups, each of which is defined as a hotspot area. The UAV's mission is to fly from a start location, move towards hotspots with high data service requests, and then return to the initial location within a time period T for battery charging. Thus, the UAV's initial ($\boldsymbol{l_0}$) and final ($\boldsymbol{l_T}$) locations are predefined, represented by $\boldsymbol{l_0} = \boldsymbol{l_T} = [\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0]$. It is important to note that the variable T is directly proportional to the number of available hotspots (N). As N increases, T also increases and vice versa. The UAV adjusts its deployment location at each flight slot according to the users realization forming a trajectory denoted by $\boldsymbol{q}_u(t) = [\mathrm{x}_u(t), \mathrm{y}_u(t), \mathrm{z}_u(t)]$. The sequence tracing UAV's travels among the available hotspots during the flight time duration is given by $\bar{\boldsymbol{q}}_u = [h_1, \dots, h_{N'}]$, where $h_n \in \mathcal{N}$ is the nth hotspot served by the UAV and N' is the total number of the hotspots served along the trajectory. Let $\mathcal{L}$ be the set of all possible trajectories the UAV might follow and $\Pr(h_{n+1}|h_n, \tau_{h_{n+1}})$ be the probability to move toward hotspot h_{n+1} after being in h_n (visited at time $T - \tau_{h_n}$), where $\tau_{h_{n+1}}$ is the remaining time to go back to the original location after serving h_{n+1}. The set of available hotspot areas is denoted as $\mathcal{N} \triangleq \{h_n = h_1, h_2, \dots, h_N\}$ and GUs across the total geographical area are denoted as $\mathcal{K} \triangleq \{K_n = K_1, K_2, \dots, K_N\}$, where K_n is the set of users belonging to the nth hotspot and each GU belongs to a single hotspot where the coordinate of each GU is given by $\boldsymbol{p}_{k_n} = [\mathrm{x}_{k_n}, \mathrm{y}_{k_n}]$. Each hotspot n is characterized by its center $\boldsymbol{p}_n = [\mathrm{x}_n, \mathrm{y}_n]$ and radius r_n representing the coverage range and the average data rate R_n that depends on the number of active users in hotspot n where $\mathrm{R}_n \in \mathcal{R}$, such that $\mathcal{R} \triangleq \{R_n = R_1, R_2, \dots, R_N\}$.

To capture the dynamic nature of the network, the UAV flight time (T) is discretized into a set $\mathcal{T}$ of M equal time slots where the length of each time slot is $t = (\frac{T}{M})$. Due to its short duration, the UAV's location, uplink data requests and channel conditions are considered fixed in each t. Furthermore, in the considered network, the UAV assigns a set of uplink resource blocks (RBs) to serve the active GUs in a specific hotspot (one RB for each active GU) who transmit their data over the allocated RBs using the orthogonal frequency division multiple access (OFDMA) scheme.

In our network, the air-to-ground signal propagation is adopted and a probabilistic path loss model subject to random line-of-sight (LoS) and non-line-of-sight (NLOS) conditions is considered [62]. The channel gain between a GU ($k_n \in K_n$) and a UAV (u) can be expressed as:

$$g_{k_n,u}(t) = \frac{1}{\mathbb{K}_0 d^{\alpha}_{k_n,u}(t)} [\mathrm{Pr}_{\mathrm{LoS}} \mu_{\mathrm{LoS}} + \mathrm{Pr}_{\mathrm{NLoS}} \mu_{\mathrm{NLoS}}]^{-1}, \tag{1}$$

where $\mathbb{K}_0 = \left(\frac{4\pi f_c}{c}\right)^2$, f_c is the carrier frequency, c is the speed of light, α is the path loss exponent, and $\mathrm{Pr}_{\mathrm{LoS}}$ and $\mathrm{Pr}_{\mathrm{NLoS}}$ are the LoS and NLoS probabilities, respectively. μ_{LoS} and

μ_{NLoS} are additional attenuation factors to the free-space propagation for LoS and NLoS links, respectively. The distance between a GU (k_n) and the UAV at time slot t is given by:

$$d_{k_n,u}(t) = \sqrt{h_u(t)^2 + \left(\mathrm{x}_{k_n}(t) - \mathrm{x}_u(t)\right)^2 + \left(\mathrm{y}_{k_n}(t) - \mathrm{y}_u(t)\right)^2}. \tag{2}$$

The average achievable data rate of the set of users in hotspot n is calculated as:

$$r_{K_n} = \sum_{k_n=1}^{K_n} r_{k_n} = \sum_{k_n=1}^{K_n} B_{k_n} \log_2\left(1 + \frac{p_{k_n} g_{k_n,u}(t)}{\sigma^2}\right), \tag{3}$$

where B_{k_n} is the bandwidth of the RB allocated to GU (k_n), p_{k_n} is the transmit power of GU (k_n), and $\sigma^2 = B_{k_n} N_0$ is the power spectral density of the additive white Gaussian noise (AWGN).

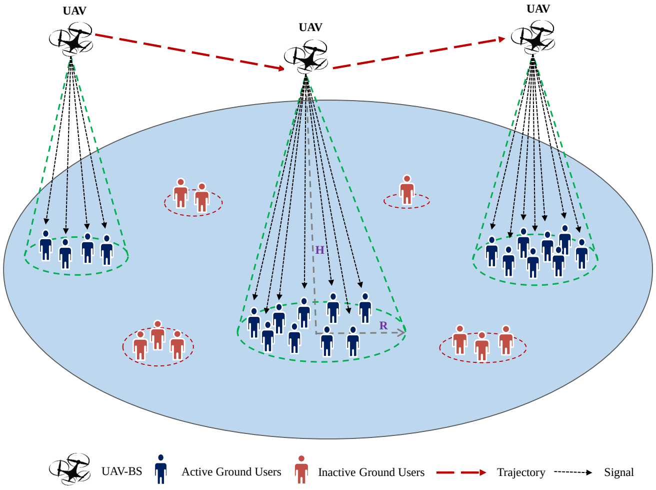


Figure 2. Illustration of the system model.

In this work, we focus on UAV trajectory design that can maximize the total sum-rate in the cell. Therefore, our optimization objective can be formulated as:

$$\max_{q_u \in \mathcal{L}} \quad r_{sum} = \sum_{h_n=1}^{N'} \sum_{k_n=1}^{K_n} \mathrm{r}_{k_n} \prod_{h_n=1}^{N'-1} \Pr(h_{n+1}|h_n, \tau_{h_{n+1}}) \tag{4a}$$

$$\text{s.t.} \quad k_i \cap k_j = \phi,\ i \neq j,\ \forall i,j \in \mathcal{N}, \tag{4b}$$

$$t(\boldsymbol{q}_u) \leq T,\ \boldsymbol{q}_u \in \mathcal{L}, \tag{4c}$$

$$0 \leq \Pr(h_{n+1}|h_n, \tau_{h_{n+1}}) \leq 1,\ 1 \leq h_n \leq N' - 1, \tag{4d}$$

$$\mathrm{r}_{k_n} \geq r_0,\ \forall k_n, \tag{4e}$$

$$0 \leq p_{k_n} \leq p_{max},\ \forall k_n. \tag{4f}$$

Constraint (4b) indicates that each GU belongs to a specific hotspot. (4c) implies that the UAV must go back to the initial location before T, where T is directly proportional to N. If N increases, T will also increase; if N decreases, T will also decrease. Furthermore, (4e) represents the sum-rate requirement for each GU and (4f) depicts the power allocation constraint. It is worth noting that in this paper, the number of hotspots remains constant in a certain mission (realization). No new hotspots emerge nor do any existing hotspots disappear while the UAV is solving a specific realization.

The symbols used in the article and their meanings are summarized in Table 1.

Table 1. Variables Description.

Symbol	Meaning
$\mathcal{U}$	Ground users (GUs)
N	Number of hotspots
T	Battery life time
$\boldsymbol{l_0}$	UAV’s initial location
$\boldsymbol{l_T}$	UAV’s final location
$\boldsymbol{q}_u(t)$	UAV’s trajectory
$\bar{\boldsymbol{q}}_u$	Sequence of hotspots served by the UAV
h_n	nth hotspot served by the UAV
N'	Total number of hotspots served along the trajectory
$\mathcal{L}$	Set of possible trajectories to follow by the UAV
$\Pr(h_{n+1}\|h_n, \tau_{n+1})$	Probability to move toward hotspot h_{n+1} after visiting h_n at time $T - \tau_{h_n}$
τ_{h_n}	Remaining time to go back to the original location after serving h_n
$\mathcal{N}$	The set of available hotspot areas
$\mathcal{K}$	The set of GUs distributed across the total geographical area
K_n	The set of GUs belonging to the nth hotspot
$\boldsymbol{p}_{k_n} = [\mathrm{x}_{k_n}, \mathrm{y}_{k_n}]$	The coordinate of GU k_n belonging to the K_n
$\boldsymbol{p}_n = [\mathrm{x}_n, \mathrm{y}_n]$	Center of nth hotspot
r_n	Radius of the nth hotspot
$\mathcal{R}$	The set of the average data rate of all the available hotspots
R_n	Data rate of the nth hotspot
t	Time slot
u	UAV
$g_{k_n,u}(t)$	Channel gain between GU (k_n) and UAV (u)
$\mathbb{K}$	Channel factor
f_c	Carrier frequency
c	Speed of light
α	Path loss exponent
$\mathrm{Pr}_{\mathrm{LoS}}$	Probability of line-of-sight
$\mathrm{Pr}_{\mathrm{NLoS}}$	Probability of non-line-of-sight
μ_{LoS}	Additional attenuation for line-of-sight links
μ_{NLoS}	Additional attenuation for non-line-of-sight links
$d_{k_n,u}(t)$	Distance between GU k_n and UAV u at time t
r_{K_n}	Achievable data rate in hotspot n
B_{k_n}	The bandwidth of the resource block (RB) allocated to user k_n
p_{k_n}	Transmit power of user k_n
σ^2	Power spectral density of the additive white Gaussian noise
$\mathcal{D}$	Training set of realizations representing M examples
$\mathcal{L}^{\dagger}$	Set of the sequences of hotspots selected by TSPWP to solve M examples
$\mathcal{Q}^{\dagger}$	Set of trajectory instances generated by TSPWP
$\mathcal{S}$	Set of clusters generated by GNG
$\tilde{l}_m$	Generalized letter
$\mathbf{A}_{\tilde{\boldsymbol{l}}_{\boldsymbol{m}}}$	Adjacency matrix
$\mathbb{A}_{\tilde{\boldsymbol{l}}}$	Global adjacency matrix
$\boldsymbol{\Pi}_{\tilde{\boldsymbol{l}}}$	Global transition matrix
$\mathbf{D}$	Degree matrix
Θ_{em}	Tokens
$\boldsymbol{\Pi}_{\boldsymbol{\Theta}}$	Tokens transition matrix
$w^{o}_{T,em}$	Words on order
$w^{p}_{T,em}$	Words on motion
$w^{c}_{T,em}$	Coupling word

4. Proposed Goal-Directed Trajectory Design Method

In this section, we propose a goal-directed method for UAV trajectory design based on active inference. The latter is a model-based data-driven approach that rests upon the idea of using an internal generative model (world model) to cast the surrounding environment and planning actions allowing realization goals to be targeted by the agent. First, we present the perceptual learning of desired observation based on a classical traveling salesman problem (TSP) with 2-OPT [63]. Then, we show how to build the world model representing the surrounding environment by encoding the dynamic rules behind the optimal TSP trajectories.

4.1. TSP with Profits Instances

The traditional TSP is a classic algorithm problem in computer science and operation research describing how a salesman travels to several vertices (cities) and returns to the terminal (initial location), aiming to minimize the travel cost (i.e., the travel distance) while ensuring visiting each city only once [63]. In this work, we adopt the TSP with profits (TSPWP) with the 2-OPT local search algorithm [39], which is a generalization of the traditional TSP where the overall goal is the simultaneous optimization of the collected profit and the travel cost, knowing that each vertex (city) is associated with a profit. Thus, TSPWP is used to generate optimal trajectory instances *offline* that the UAV might follow to serve more users within a predefined time. Given a list of hotspots where the active users are distributed, as shown in Figure 2, and the cost (c_{ij}) of transiting between each pair of hotspots, the problem is to find the optimal route that visits each hotspot once and returns to the origin, providing a maximum sum-rate and a minimum completion time.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is a set of N vertices and $\mathcal{E}$ is a set of edges. Let $\boldsymbol{p}_n$ be the center of v_n and r_{K_n} the profit associated with v_n and a cost c_{ij} be associated with each edge $(v_i, v_j) \in \mathcal{E}$, such that:

$$c_{ij} = d(\boldsymbol{p}_i, \boldsymbol{p}_j) = \sqrt{(\mathrm{x}_i - \mathrm{x}_j)^2 + (\mathrm{y}_i - \mathrm{y}_j)^2}. \tag{5}$$

The objective function of the TSPWP with N hotspots can be defined as:

$$\min \quad \alpha \sum_{(v_i,v_j)\in\mathcal{E}} c_{ij}x_{ij} - \beta \sum_{v_j\in\mathcal{V}} r_{K_j} y_j, \tag{6a}$$

$$\text{s.t.} \quad \sum_{\substack{v_i\in\mathcal{V} \\ v_j\in\mathcal{V}\setminus\{v_i\}}} x_{ij} = y_i, \tag{6b}$$

$$\sum_{\substack{v_j\in\mathcal{V} \\ v_i\in\mathcal{V}\setminus\{v_j\}}} x_{ij} = y_j, \tag{6c}$$

$$x_{ij} \in \{0,1\},\ (v_i, v_j) \in \mathcal{E}, \tag{6d}$$

$$y_{ij} \in \{0,1\},\ (v_i \in \mathcal{V}), \tag{6e}$$

$$\alpha + \beta = 1. \tag{6f}$$

Constraints (6b) and (6c) are the assignment constraints where x_{ij} is a binary variable associated with edge (v_i, v_j), equal to 1 if and only if (v_i, v_j) is used in the solution, and y_i is a binary variable associated with vertex $v_i \in V$, equal to 1 if and only if v_i is visited.

4.2. World Model

The proposed approach consists of two computational units. The first unit aims to learn the surrounding environment by representing the statistical structure of the sensory signals (world model). The second is the decision-making unit seeking to perform actions minimizing (or maximizing) a cost function describing preferred outcomes (similar to rewards in RL). The world model is an internal generative model representing the surrounding environment (both physical and wireless environment) utilized by the UAV to make predictions about incoming sensory signals. In this subsection, given the TSPWP instances generated previously from several experiences (i.e., realizations of users distribution and users requests), our objective is to encode the dynamic rules generating those instances in a probabilistic graphical model capable of reflecting the graph structure of the TSPWP instances at multiple hierarchical levels and different time scales.

4.2.1. Dictionary Learning

Each TSPWP instance comprises the trajectory the UAV follows to reach the targeted hotspots in a particular order. Hence, the objective is to form a dictionary capturing the TSPWP graph structure, allowing one to predict the most probable hotspot to target conditioned on a specific location and the most probable path to follow to reach that targeted hotspot. Thus, the dictionary consists of two sub-dictionaries. The first encodes the rules generated the sequence order of the hotspots that the UAV intends to serve. By contrast, the second sub-dictionary encodes the rules generated the motion to travel among to neighboring hotspots. Figure 3 illustrates the process of forming the global dictionary.

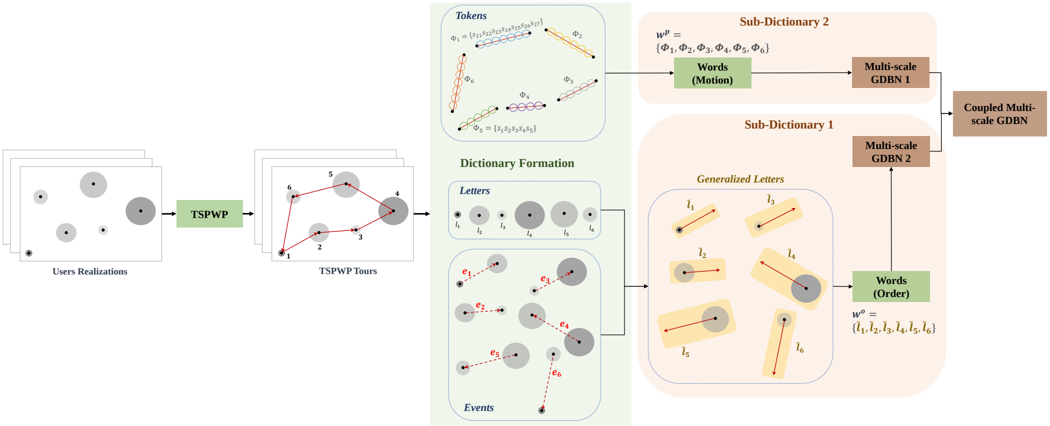

Users Realizations
TSPWP
1
2
3
4
5
6
TSPWP Tours
Tokens
$\Phi_1 = \{s_{21}s_{22}s_{23}s_{24}s_{25}s_{26}s_{27}\}$
Φ_2
Φ_3
Φ_4
$\Phi_5 = \{s_1s_2s_3s_4s_5\}$
Φ_6
Dictionary Formation
Letters
l_1 l_2 l_3 l_4 l_5 l_6
e_1 e_2 e_3 e_4 e_5 e_6
Events
Sub-Dictionary 2
$w^p = \{\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5, \Phi_6\}$
Words (Motion)
Multi-scale GDBN 1
Coupled Multi-scale GDBN
Sub-Dictionary 1
Multi-scale GDBN 2
Generalized Letters
$\tilde{l}_1$ $\tilde{l}_2$ $\tilde{l}_3$ $\tilde{l}_4$ $\tilde{l}_5$ $\tilde{l}_6$
Words (Order)
$w^o = \{\tilde{l}_1, \tilde{l}_2, \tilde{l}_3, \tilde{l}_4, \tilde{l}_5, \tilde{l}_6\}$

Figure 3. The procedure to form the global dictionary.

(1) TSPWP offline execution:

Let $\mathcal{D} \triangleq \{D_m = D_1, D_2, \ldots, D_M\}$ be a training set of realizations representing M examples of users' distribution in the cell, where D_m is the m-th realization and M is the total number of realizations. Each realization consists of the number of hotspots and their locations, the number of users inside each hotspot as well as the users' access request and users' locations. The TSPWP algorithm will be employed offline to solve all the examples in $\mathcal{D}$. Consequently, let $\mathcal{L}^\dagger \triangleq \{L_m = L_1, L_2, \ldots, L_M\}$ be a set of the sequences of hotspots selected by the UAV using TSPWP to solve the M examples, where $L_m = \{h_1, \ldots, h_{N'}\}$ is the m-th sequence of hotspots selected by the UAV to solve the m-th example and let $\mathcal{Q}^\dagger \triangleq \{\boldsymbol{q}_u^m = \boldsymbol{q}_u^1, \boldsymbol{q}_u^2, \ldots, \boldsymbol{q}_u^M\}$ be the set of trajectory instances generated by the TSPWP, where $\boldsymbol{q}_u^m$ is the m-th TSPWP trajectory generated to solve the m-th example.

(2) Unsupervised Clustering:

For each of the generated trajectories in $\mathcal{Q}^\dagger$, a growing neural gas (GNG) is employed on the generalized errors (GEs) provided by the unmotivated Kalman filter (UKF) [64] to discover the dynamic rules driving the different trajectories. Let $\mathcal{S}$ be the set of clusters generated by GNG and defined as:

$$\mathcal{S} \triangleq \{s_f = s_1, s_2, \ldots, s_F\}, \tag{7}$$

where s_f is the f-th cluster following a Gaussian distribution such that $s_f \sim \mathcal{N}(\mu_{s_f}, \Sigma_{s_f})$, and F is the total number of clusters. Clustering the trajectory data allows obtaining knowledge that reveals the latent characteristics of the UAV's motion.

(3) Sub-Dictionary 1:

Accordingly, from $\mathcal{L}^\dagger$ we form a sub-dictionary encoding the decisions made by the UAV consisting of the sequences of targeted hotspots. We define a letter $l_m = h_m$ representing a starting hotspot h_m at a given time and a generalized letter defined as:

$$\tilde{l}_m = [h_m, E(h_m, h_{m'})], \tag{8}$$

consisting of the letter itself and its derivative illustrating the event of traveling from hotspot h_m to hotspot $h_{m'}$. It is of note that a generalized letter $\tilde{l}_m$ can be seen as a pair of one node $n_i = h_m$ and one outgoing arc (n_i, n_j) from node n_i to node n_j, as shown in Figure 3. Then, for each element L_m in $\mathcal{L}^\dagger$, we transform the sequence of generalized letters expressing that experience into the following sequence: $\{\tilde{l}_{m,\tau_1}, \tilde{l}_{m,\tau_2}, \ldots, \tilde{l}_{m,\tau_T}\}$ describing the transitions between adjacent event-steps. As mentioned before, the generalized letters of a certain experience m can be seen as an unweighted graph $\mathcal{G}_m = (\mathcal{V}_m, \mathcal{E}_m)$, where $\mathcal{V}_m = \{l_{m,\tau_1}, \ldots, l_{m,\tau_T}\}$ is a set of vertices represented by the letters and $\mathcal{E}_m = \{\dot{l}_{m,\tau_1}, \ldots, \dot{l}_{m,\tau_T}\}$ is the set of edges represented by the letters' derivatives. The adjacency matrix $\mathbf{A}_{\tilde{l}_m}$ that captures the pattern of co-occurrences in the generalized letters sequence is an $\tau_T \times \tau_T$ zero–one matrix defined as $\mathbf{A}_{\tilde{l}_m} = [\mathrm{a}_{ij}]$, where:

$$\mathrm{a}_{ij} = \begin{cases} 1 & \text{if} \quad (i,j) \in \mathcal{E}, \\ 0 & \text{Otherwise.} \end{cases} \tag{9}$$

After executing the M examples, we can form the global adjacency matrix $\mathbb{A}_{\tilde{l}} = [\mathrm{a}_{i^{'},j^{'}}]$ comprising all the generalized letters (forming a global graph $\mathcal{G}_{global} = (\mathcal{V}_{global}, \mathcal{E}_{global})$) that occurred while solving the M examples, such that:

$$\mathrm{a}_{i^{'},j^{'}} = \begin{cases} 1 & \text{if} \quad (i^{'},j^{'}) \in \mathcal{E}_{global}, \\ 0 & \text{Otherwise.} \end{cases} \tag{10}$$

Where element $\mathrm{a}_{i^{'},j^{'}}$ denotes the number of times that a generalized letter $\tilde{l}_{i^{'}}$ is followed by generalized letter $\tilde{l}_{j^{'}}$ during two consecutive events in the global graph $\mathcal{G}_{global}$.

The degree of each letter $i = l_{m,\tau_i}$ is the number of its adjacent letters (or the number of outgoing edges at that letter) calculated as: $d_i = \sum_{j=1}^{|\mathcal{V}_m|} \mathrm{a}_{i^{'} j^{'}}$. Considering the degrees of all letters, we can construct the degree matrix $\mathbf{D}$, which is an $|\mathcal{V}_m| \times |\mathcal{V}_m|$ diagonal matrix defined as:

$$\mathrm{D}_{i^{'} j^{'}} = \begin{cases} d_{i^{'}} & \text{if} \quad i^{'} = j^{'}, \\ 0 & \text{Otherwise.} \end{cases} \tag{11}$$

Consequently, the global transition matrix can be constructed in the following way:

$$\mathbf{\Pi}_{\tilde{l}} = \mathbf{D}^{-1} \mathbb{A}_{\tilde{l}} = \begin{bmatrix} \Pr(\tilde{l}_1|\tilde{l}_1) & \Pr(\tilde{l}_1|\tilde{l}_2) & \dots & \Pr(\tilde{l}_1|\tilde{l}_{M'}) \\ \Pr(\tilde{l}_2|\tilde{l}_1) & \Pr(\tilde{l}_2|\tilde{l}_2) & \dots & \Pr(\tilde{l}_2|\tilde{l}_{M'}) \\ \vdots & \vdots & \vdots & \vdots \\ \Pr(\tilde{l}_{M'}|\tilde{l}_1) & \Pr(\tilde{l}_{M'}|\tilde{l}_2) & \dots & \Pr(\tilde{l}_{M'}|\tilde{l}_{M'}) \end{bmatrix}, \tag{12}$$

where $0 \leq \Pr(\tilde{l}_{\tilde{i}}|\tilde{l}_{\tilde{j}}) \leq 1$ and $\sum_{\tilde{j}=1}^{\tilde{J}} \Pr(\tilde{l}_{\tilde{i}}|\tilde{l}_{\tilde{j}}) = 1, \forall \tilde{j}$. During a flight mission that lasts for a time period T, the order of visited hotspots is recorded in a word called $w_T^o = \{\tilde{l}_{m,\tau_1}, \tilde{l}_{m,\tau_2}, \dots, \tilde{l}_{m,\tau_T}\}$.

(4) Sub-Dictionary 2:

Each event $e_m = E(h_m, h_{m'})$ can be associated with a local trajectory followed by the UAV to pass from h_m to $h_{m'}$, which can be represented by a sequence of discrete clusters. This is possible after associating the local trajectory with $\mathcal{S}$ defined in (7) to form a token comprising a sequence of letters depicting the firing sequence of clusters (neurons) from $\mathcal{S}$ during a certain event, i.e, e_m. Hence, we define a token consisting of a set of clusters and representing a local path between two adjacent hotspots as follows:

$$\Theta_{e_m} = \{s_{e_m,t_1}, s_{e_m,t_2}, \dots, s_{e_m,t_\tau}\}, \tag{13}$$

where $s_{e_m,t_i} \in \mathcal{S}$, and t_τ is the duration of event e_m specified in the number of time slots. The stochastic process decomposing the interdependent nature of the tokens that make up the local trajectories can be illustrated in a transition matrix defined as:

$$\mathbf{\Pi}_{\mathbf{\Theta}} = \begin{bmatrix} \Pr(\Theta_{e_1}|\Theta_{e_1}) & \Pr(\Theta_{e_1}|\Theta_{e_2}) & \dots & \Pr(\Theta_{e_1}|\Theta_{e_M}) \\ \Pr(\Theta_{e_2}|\Theta_{e_1}) & \Pr(\Theta_{e_2}|\Theta_{e_2}) & \dots & \Pr(\Theta_{e_2}|\Theta_{e_M}) \\ \vdots & \vdots & \vdots & \vdots \\ \Pr(\Theta_{e_M}|\Theta_{e_1}) & \Pr(\Theta_{e_M}|\Theta_{e_2}) & \dots & \Pr(\Theta_{e_M}|\Theta_{e_M}) \end{bmatrix}, \tag{14}$$

where $\Pr(\Theta_{e_i}|\Theta_{e_j})$ depicts the transition probability from token i to token j, such that $0 \leq \Pr(\Theta_{e_i}|\Theta_{e_j}) \leq 1$ and $\sum_{j=1}^{J} \Pr(\Theta_{e_i}|\Theta_{e_j}) = 1, \forall j$. During a flight mission of duration

T, the tokens that represent the entire trajectory are recorded in a word called $w_T^p = \{\Theta_{e_j}, \Theta_{e_{j+1}}, \ldots, \Theta_{e_J}\}$.

4.2.2. The Proposed Hierarchical Graphical Representation

Introducing Multi-Scale GDBN: We can see that the UAV's dynamic behavior manifests at multiple time scales, namely slot scale and event scale. It is essential to have an efficient representation that can model this dynamic behavior, including a hierarchical structure and incorporating Markov chains at various time scales. To achieve this, we propose to learn two separated dynamic models representing the dynamic behavior of the UAV when selecting the targeted hotspots (i.e., the sequence of hotspots to serve during the flight time) and when moving between two consecutive hotspots (i.e., the UAV's motion path). The proposed representation considers observations stemming from two different behavioral processes with different temporal resolutions. The first process determines the decisions made by the UAV at the event scale, while the second process determines the UAV's motion at the finer time scale (slot scale), which is nested within the event scale.

The first dynamic model entails arranging particular elements of the dictionary (sub-dictionary 1), particularly the generalized letters referenced in (8), into a multi-scale generalized dynamic Bayesian network (M-GDBN) displayed in Figure 4. The M-GDBN is a hierarchical probabilistic graphical model that consists of four levels, two of which are continuous and two of which are discrete. Each level corresponds to a distinct hierarchy and time scale. Furthermore, M-GDBN explains how the latent state variables and the observation are probabilistically linked. The explanation for the evolution of hidden variables at multiple levels is provided based on the following dynamic models:

$$w_T^o = \mathrm{f}^{(1)}(w_{T-1}^o) + \eta_T, \tag{15a}$$

$$\tilde{l}_{T,e_m} = \mathrm{f}^{(2)}(\tilde{l}_{T,e_{m-1}}, w_T^o) + \eta_{T,e_m}, \tag{15b}$$

$$\tilde{x}_{T,e_m}^l = \mathrm{g}^{(1)}(\tilde{x}_{T,e_{m-1}}^l, \tilde{l}_{T,e_m}) + \eta_{T,e_m}, \tag{15c}$$

$$\tilde{z}_{T,e_m}^l = \mathrm{g}^{(2)}(\tilde{x}_{T,e_m}^l) + \nu_{T,e_m}. \tag{15d}$$

The discrete state equations in (15a) and (15b) illustrate how words and generalized letters change over time at various temporal scales. $\mathrm{f}^{(1)}$ and $\mathrm{f}^{(2)}$ are nonlinear functions that experience random fluctuations in the states influenced by higher levels and characterized by $\eta_T \sim \mathcal{N}(0, Q)$ and $\eta_{T,e_m} \sim \mathcal{N}(0, Q)$. Going down the hierarchy, Equations (15c) and (15d) stand for the continuous state equation and the observation model, explaining the continuous state dynamic evolution and the mapping from the continuous state space to the measurement space, respectively. Observations are subject to random fluctuations playing the role of observation noise characterized by $\nu_{T,e_m} \sim \mathcal{N}(0, \sigma^2_{\tilde{z}_{T,e_m}})$. Equations (15a), (15b), (15c), and (15d) can be expressed in probabilistic form as $\mathrm{Pr}(w_T^o | w_{T-1}^o)$, $\mathrm{Pr}(\tilde{l}_{T,e_m} | \tilde{l}_{T,e_{m-1}}, w_T^o)$, $\mathrm{Pr}(\tilde{x}_{T,e_m}^l | \tilde{x}_{T,e_{m-1}}^l, \tilde{l}_{T,e_m})$, and $\mathrm{Pr}(\tilde{z}_{T,e_m}^l | \tilde{x}_{T,e_{m-1}}^l)$, respectively. Thus, the consistent global model (i.e., the joint distribution function) corresponding to the network in Figure 4 is given by:

$$\mathrm{Pr}\big(\mathcal{W}^o, \tilde{\mathcal{L}}, \tilde{\mathcal{X}}^l, \tilde{\mathcal{Z}}^l\big) = \prod_T \mathrm{Pr}\big(w_T^o\big) \prod_{T,e_m} \mathrm{Pr}\big(\tilde{l}_{T,e_m} | w_T^o\big) \mathrm{Pr}\big(\tilde{x}_{T,e_m}^l | \tilde{l}_{T,e_m}\big) \mathrm{Pr}\big(\tilde{z}_{T,e_m}^l | \tilde{x}_{T,e_m}^l\big). \tag{16}$$

M-GDBN is a directed acyclic graph where every node represents a random variable or uncertain quantity that can have multiple values. The arcs indicate a direct causal influence between linked variables, and the strength of these influences is measured by conditional probabilities. To determine the structure of M-GDBN, a node is assigned to each variable, and arrows are drawn towards it from nodes that are perceived to be its direct cause. To determine the strength of direct influences, each variable is assigned a link matrix. This matrix represents the estimated conditional probabilities of the event based on the parent set's value combination.

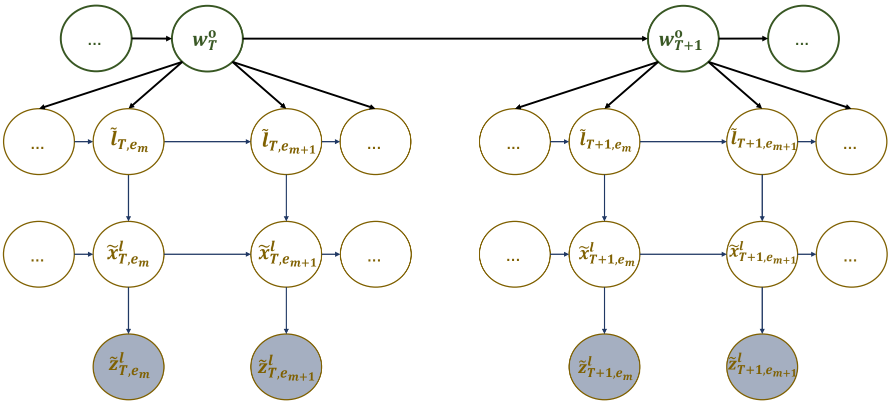


Figure 4. A multi-scale GDBN representing sub-dictionary 1 that encodes the dynamic rules generating UAV's hotspots sequence in different experiences.

In Figure 5, there is another multi-scale GDBN that deals with the dictionary components concerning the UAV's dynamic motion (sub-dictionary 2). This second network has three discrete levels and two continuous levels. The variables at the various levels explain how the observations (i.e., the UAV's trajectory) were generated. For instance, at the word scale, each word is made up of tokens that were realized at different events (event scale). Each token, in turn, is composed of discrete and continuous letters that generate observations at different slots.

In order to comprehend the generative process forming the UAV's global trajectory, we can refer to the dynamic models below:

$$w^p_T = \mathrm{f}^{(1)}(w^p_{T-1}) + \eta_T, \tag{17a}$$

$$\Theta_{T,e_m} = \mathrm{f}^{(2)}(\Theta_{T,e_{m-1}}, w^p_T) + \eta_T, \tag{17b}$$

$$\tilde{s}_{e_m,t_i} = \mathrm{f}^{(3)}(\tilde{s}_{e_m,t_{i-1}}, \Theta_{T,e_m}) + \eta_{T,e_m}, \tag{17c}$$

$$\tilde{x}_{e_m,t_i} = \mathrm{g}^{(1)}(\tilde{x}_{e_{m-1},t_{i-1}}, \tilde{s}_{e_m,t_i}) + \eta_{e_m,t_i}, \tag{17d}$$

$$\tilde{z}_{e_m,t_i} = \mathrm{g}^{(2)}(\tilde{x}_{e_m,t_i}) + \nu_{e_m,t_i}. \tag{17e}$$

The discrete state equations in (17a), (17b), and (17c) show how the trajectory words, tokens, and trajectory clusters change over time at various temporal scales. These equations use non-linear functions $\mathrm{f}^{(1)}$, $\mathrm{f}^{(2)}$, and $\mathrm{f}^{(3)}$ subject to process noise $\eta_T \sim \mathcal{N}(0, Q)$. The continuous state equation in (17d) explains how the trajectory states evolve over time, while (17e) links observations to these states. The equations mentioned earlier can be expressed probabilistically as follows: $\mathrm{Pr}(w^p_T | w^p_{T-1})$, $\mathrm{Pr}(\Theta_{T,e_m} | \Theta_{T,e_{m-1}}, w^p_T)$, $\mathrm{Pr}(\tilde{s}_{e_m,t_i} | \tilde{s}_{e_m,t_{i-1}}, \Theta_{T,e_{m-1}})$, $\mathrm{Pr}(\tilde{x}_{e_m,t_i} | \tilde{x}_{e_m,t_{i-1}}, \tilde{s}_{e_m,t_i})$, and $\mathrm{Pr}(\tilde{z}_{e_m,t_i} | \tilde{x}_{e_m,t_i})$, respectively. The network in Figure 5 has a compatible global model, represented by a joint distribution function that can be expressed as:

$$\mathrm{Pr}(\mathcal{W}^p, \Phi, \tilde{\mathcal{S}}, \tilde{\mathcal{X}}, \tilde{\mathcal{Z}}) = \prod_{T} \mathrm{Pr}(w^p_T) \prod_{T,e_m} \mathrm{Pr}(\Theta_{T,e_m} | w^p_T) \mathrm{Pr}(\tilde{s}_{T,e_m} | \tilde{x}_{T,e_m}) \mathrm{Pr}(\tilde{z}_{T,e_m} | \tilde{x}_{T,e_m}). \tag{18}$$

Coupled-MGDBN: We have organized the dictionaries we obtained into a coupled multi-scale generalized dynamic Bayesian network (C-MGDBN), which includes the two dynamic models. The first model represents the sequence of hotspots the UAV selects to solve the realizations encountered during training, which is structured in sub-dictionary 1. Meanwhile, the second model represents the UAV's path to travel between consecutive hotspots, which is structured in sub-dictionary 2. By coupling these two models stochastically in the C-MGDBN, we can incorporate more complex and sophisticated dynamics and model stochastic representations of multiple behaviors. Additionally, we have added an efficient mechanism to the C-MGDBN that captures multiple event and state transitions, which help explain how the UAV approached a particular task (such as trajectory design) in different examples.

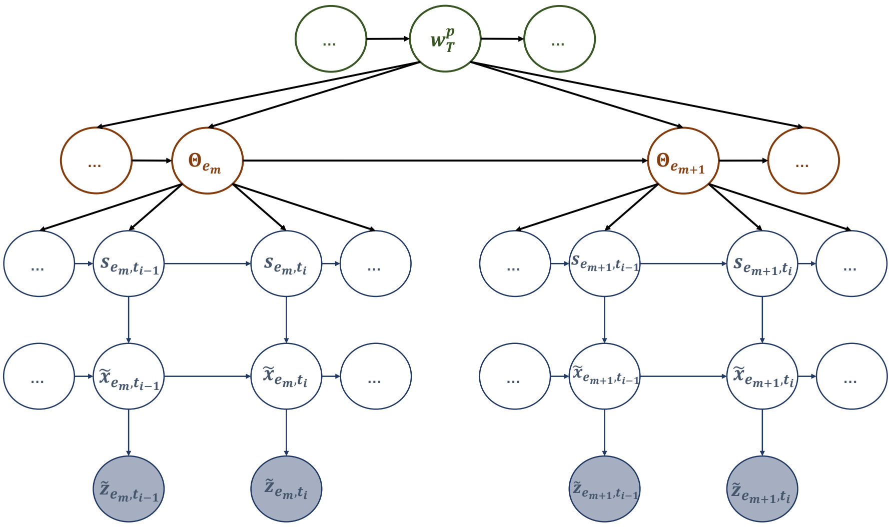


Figure 5. A multi-scale GDBN representing sub-dictionary 2 that encodes the dynamic rules generating the UAV's positions to travel among the hotspots in different events.

We coupled the two M-GDBN models mentioned earlier at the event scale as shown in Figure 6. This was done because multiple events make up a complete mission. We have yet to investigate coupling at the word scale. However, this coupling technique can be useful if the UAV is performing various missions. For instance, after serving active users in a specific cell, the UAV can return to its initial station for recharging before proceeding to another mission. In this way, by learning the dynamics of real-life scenarios, which include users' activities and the emergence of hotspots, the UAV can plan its actions at the word scale. For the rest of the paper, we will assume that the UAV is making plans at both the event and slot scales.

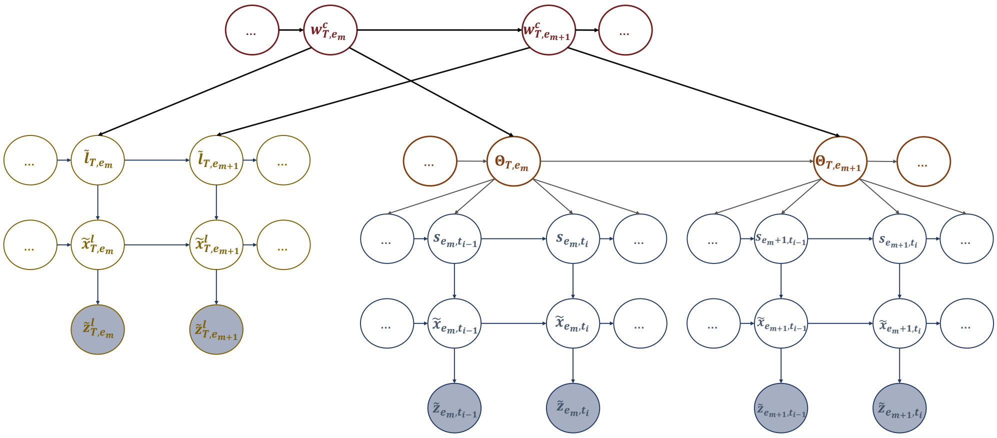


Figure 6. A coupled multi-scale GDBN (C-MGDBN) structures the acquired dictionaries by coupling the corresponding models at the event scale.

In the C-MGDBN depicted in Figure 6, the current discrete state is influenced by the state of its own chain and that of the neighboring chain from the previous event step. To avoid overwhelming complexity, we conducted a meta-clustering process by merging dependent nodes in the connected network into a single higher-dimensional node. In other words, $\Pr(\Theta_{T,e_{m+1}}|\Theta_{T,e_m},\tilde{l}_{T,e_m})$, and vice versa $\Pr(\tilde{l}_{T,e_{m+1}}|\tilde{l}_{T,e_m},\Theta_{T,e_m})$. To estimate these probabilities we need two transition matrices encoding the probabilistic relationships between words and tokens. Merging letters and tokens allows to simplify the case by coupling them into a higher node $w^c_{T,e_m} = [\tilde{l}_{T,e_m}, \Theta_{T,e_m}]$. The evolution of the coupling words w^c_{T,e_m} can be captured by the transition matrix defined as:

$$\mathbf{\Pi}_{w^c} = \begin{bmatrix} \Pr(w^c_1|w^c_1) & \Pr(w^c_1|w^c_2) & \dots & \Pr(w^c_1|w^c_C) \\ \Pr(w^c_2|w^c_1) & \Pr(w^c_2|w^c_2) & \dots & \Pr(w^c_2|w^c_C) \\ \vdots & \vdots & \vdots & \vdots \\ \Pr(w^c_C|w^c_1) & \Pr(w^c_C|w^c_2) & \dots & \Pr(w^c_C|w^c_C) \end{bmatrix}, \tag{19}$$

where $0 \leq \Pr(w^c_i|w^c_j) \leq 1$ and $\sum_{j=1}^{J} \Pr(w^l_i|w^c_j) = 1, \forall j$. $\mathbf{\Pi}_{w^c}$ can be considered as a combined transition matrix, formed by coupling (12) with (14).

4.3. Active Inference

During the active inference process, a UAV can learn, adapt, and perceive its body as a unit while interacting with the environment. The UAV's world model can be defined as a partially observable Markov decision process (POMDP). It involves a probability distribution $\Pr(\mathcal{Z}^l, \mathcal{Z}, \tilde{\mathcal{X}}^l, \tilde{\mathcal{X}}, \mathcal{S}, \tilde{\mathcal{L}}, \mathcal{A}^l, \mathcal{A}^p, \mathcal{W})$ that determines the joint probability of the UAV's observations, belief states, actions, and words (i.e., policies). In simpler terms, a word (or policy) refers to a set of actions. This concept is illustrated through events in Figure 7, and it can be expressed in the following format:

$$\begin{aligned} \Pr(\mathcal{Z}^l, \mathcal{Z}, \tilde{\mathcal{X}}^l, \tilde{\mathcal{X}}, \mathcal{S}, \tilde{\mathcal{L}}, \mathcal{A}^l, \mathcal{A}^p, \mathcal{W}) &= \Pr(\tilde{l}_0)\Pr(\tilde{x}^l_0)\Pr(w^c_0) \\ \prod_{e_m=1}^{E_m} \Pr(\tilde{z}^l_{e_m}|\tilde{x}_{e^l_m})\Pr(\tilde{x}^l_{e_m}|\tilde{l}_{e_m})\Pr(\tilde{l}_{e_m}|w^c_{e_m})&\Pr(w^c_{e_m}|a^l_{e_{m-1}})\Pr(a^l_{e_{m-1}}|w^c_{e_{m-1}})\times \\ \Pr(\tilde{s}_0)\Pr(\tilde{x}_0)\prod_{t_i=1}^{T_i} \Pr(\tilde{z}_{e_m,t_i}|\tilde{x}_{e_m,t_i})\Pr(\tilde{x}_{e_m,t_i}|\tilde{s}_{e_m,t_i})&\Pr(\tilde{s}_{e_m,t_i}|a^p_{e_m,t_{i-1}})\Pr(a^p_{e_m,t_{i-1}}|a^p_{e_m,t_{i-2}}, w^c_{e_{m-1}}). \end{aligned} \tag{20}$$

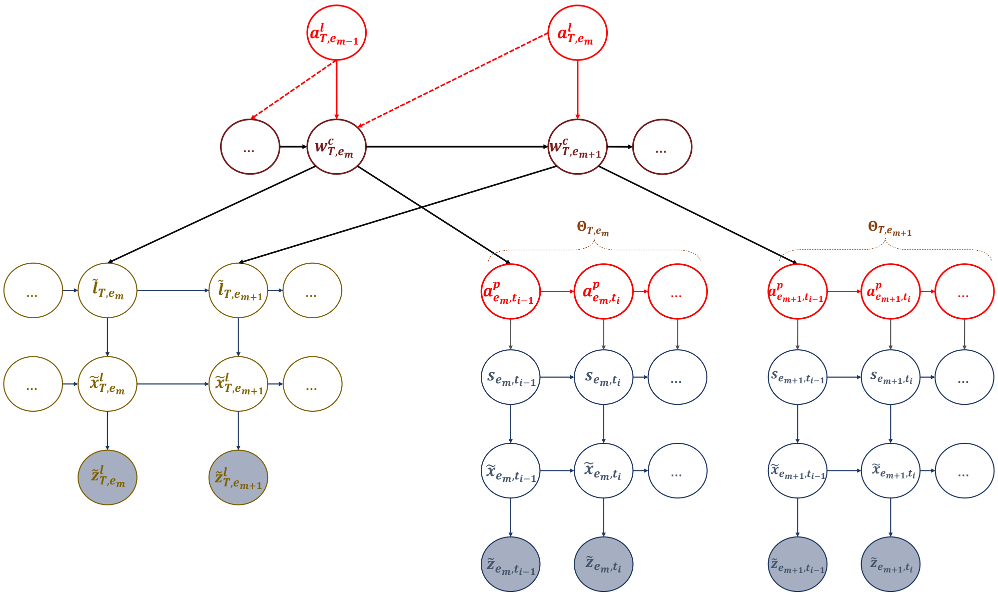


Figure 7. An active multi-scale GDBN involving the active states representing the actions that the UAV can perform and affect the dynamic rules generating UAV's positions to travel among the hotspots in different events.

4.3.1. Action Selection

The UAV performs two types of actions: one related to the targeted hotspot and the other pertaining to controlling its motion while moving towards it. To do this, the UAV relies on two AIn tables to select these actions. The first table encodes the relationship between the words and the discrete actions at the event scale defined as:

$$\mathbf{AIn_1} = \begin{bmatrix} \Pr(a_1^l|w_1^c) & \Pr(a_2^l|w_1^c) & \dots & \Pr(a_U^l|w_1^c) \\ \Pr(a_1^l|w_2^c) & \Pr(a_2^l|w_2^c) & \dots & \Pr(a_U^l|w_2^c) \\ \vdots & \vdots & \vdots & \vdots \\ \Pr(a_1^l|w_C^c) & \Pr(a_2^l|w_C^c) & \dots & \Pr(a_U^l|w_C^c) \end{bmatrix}, \tag{21}$$

where $0 \leq \Pr(a_i^l|w_j^c) \leq 1$ and $\sum_{j=1}^{J} \Pr(a_i^l|w_j^c) = 1, \forall j$. The other table encodes the relationship between the words and the continuous actions at the slot scale:

$$\mathbf{AIn_2} = \begin{bmatrix} \Pr(a_1^p|w_1^c) & \Pr(a_2^p|w_1^c) & \dots & \Pr(a_U^p|w_1^c) \\ \Pr(a_1^p|w_2^c) & \Pr(a_2^p|w_2^c) & \dots & \Pr(a_U^p|w_2^c) \\ \vdots & \vdots & \vdots & \vdots \\ \Pr(a_1^p|w_C^c) & \Pr(a_2^p|w_C^c) & \dots & \Pr(a_U^p|w_C^c) \end{bmatrix}, \tag{22}$$

where $0 \leq \Pr(a_i^p|w_j^c) \leq 1$ and $\sum_{j=1}^{J} \Pr(a_i^p|w_j^c) = 1, \forall j$.

The decisions made by the UAV to select actions that represent the targeted hotspot depend on the current word (i.e., the current location of the UAV), which is determined by the probability entries in (21). Thus, discrete actions are sampled from:

$$a_{e_m}^l \sim \Pr(.|w_{e_m}^c), \tag{23}$$

where $a_{e_m}^l$ is the selected discrete action at event e_m that impacts future environmental hidden states and observations at event e_{m+1}. This ensures that the decisions made by the UAV are targeted towards the desired hotspots. Once the targeted hotspot is chosen (i.e., $a_{e_m}^l$), the UAV will then select a second action ($a_{e_m}^p$) that dictates how it will reach the targeted hotspot. This action is determined by the UAV's starting hotspot and UAV's target (represented by word $w_{e_m}^c$) and involves a series of actions at a more detailed time scale (slot scale). At the beginning of event e_m, the UAV selects the initial continuous action at the initial time slot t_1 of that event according to:

$$a_{e_m,t_1}^p = \text{randint}(1, |\mathcal{A}^p|), \tag{24}$$

where $\mathcal{A}^p = \{North, South, East, West\}$, $|\mathcal{A}^p|$ is the total number of available predefined actions, and $\text{randint}(1, |\mathcal{A}^p|)$ is a function representing a uniform distribution that generates an integer uniformly between 1 and $|\mathcal{A}^p|$. During event e_m, the following continuous actions in the subsequent time slots t_i are chosen based on previous continuous actions and prediction errors. More details on this will be explained later.

4.3.2. Prediction and Perception

The UAV can anticipate the outcomes of joint actions at different time scales and levels of hierarchy. On a long-term scale, the UAV expects an increase in the number of served users after each event and every discrete action representing the targeted hotspots. This helps the UAV achieve its primary goal. On a smaller scale, while moving towards the targeted hotspot, the UAV anticipates reaching its second goal with each continuous action it takes during each time slot. Hence, the predictions are performed at two different temporal scales.

At the event scale, to predict the coupling word w_{T,e_m}^c, UAV employs a particle filter (PF) that propagates a set $\{w_{T,e_m}^{c(n)}, \omega_{T,e_m}^{l(n)}\}_{n=1}^{N}$ of equally weighted particles sampled from the matrix $\mathbf{\Pi}_{w^c}$ defined in (19). The UAV expresses its belief of how a specific word changes into another based on the performed action through a probabilistic form $\Pr(w_{T,e_m}^{c(n)}|w_{T,e_{m-1}}^{c(n)}, a_{T,e_{m-1}}^l)$. The predicted coupled word comprises the predicted generalized letter ($\tilde{l}^{(n)}_{T,e_m}$) and predicted token ($\Theta^{(n)}_{T,e_m}$) since the word is formed by coupling these two components. For each propagated particle, UAV employs a Kalman filter (KF) to predict the continuous state $\tilde{x}^{l(n)}_{T,e_m}$ explaining the dynamics of the data rate. The KF relies on the dynamic model defined in (15c), which can be represented by the probability distribution $\mathrm{Pr}(\tilde{x}^{l(n)}_{T,e_m}|\tilde{x}^{l(n)}_{T,e_{m-1}},\tilde{l}^{(n)}_{T,e_m})$. The posterior refers to the updated belief that forms after considering previous observations. It is connected to predictions and can be expressed as follows: $\pi(\tilde{x}^{l}_{T,e_m}) = \mathrm{Pr}(\tilde{x}^{l(n)}_{T,e_m},\tilde{l}^{(n)}_{T,e_m}|\tilde{z}^{l}_{T,e_{m-1}})$. As the UAV obtains new observations, diagnostic messages propagating in a bottom–up manner can be used to update the posterior according to:

$$\pi(\tilde{x}^{l}_{T,e_m}) = \pi(\tilde{x}^{l}_{T,e_m}) \times \lambda(\tilde{x}^{l}_{T,e_m}), \tag{25}$$

where $\lambda(\tilde{x}^{l}_{T,e_m}) = \mathrm{Pr}(z^{l}_{T,e_m}|\tilde{x}^{l}_{T,e_m})$. Likewise, particles weights are updated at the higher level following:

$$\omega^{l(n)}_{T,e_m} = \omega^{l(n)}_{T,e_m} \times \lambda(\tilde{l}_{T,e_m}), \tag{26}$$

where:

$$\lambda(\tilde{l}_{T,e_m}) = \lambda(\tilde{x}^{l}_{T,e_m})\mathrm{Pr}(\tilde{x}^{l}_{T,e_m}|\tilde{l}_{T,e_m}) = \mathrm{Pr}(z^{l}_{T,e_m}|\tilde{x}^{l}_{T,e_m})\mathrm{Pr}(\tilde{x}^{l}_{T,e_m}|\tilde{l}_{T,e_m}), \tag{27}$$

and $\mathrm{Pr}(\tilde{x}^{l}_{T,e_m}|\tilde{l}_{T,e_m}) \sim \mathcal{N}(\mu_{\tilde{l}_{T,e_m}},\sigma_{\tilde{l}_{T,e_m}})$.

On the other hand, at the slot scale, the UAV predicts the consequence of the continuous actions following the same approach explained earlier. By employing another PF, the UAV can predict the evolution of the discrete states s_{e_m,t_i} realizing the discrete zone of the UAV's trajectory forming a token Θ_{e_m}. The UAV believes that the discrete states evolve in accordance with $\mathrm{Pr}(s_{e_m,t_i}|s_{e_m,t_{i-1}},\Theta_{e_m},a^{p}_{e_m,t_{i-1}})$. The PF propagates a set of particles representing the predicted discrete states, $\{s^{(n)}_{e_m,t_i},\omega^{(n)}_{e_m,t_i}\}_{n=1}^{N}$, that are sampled using the transition matrix $\mathbf{\Pi}_{\mathbf{\Theta}}$ defined in (14). Consequently, a bank of KFs is employed to predict the continuous states representing the UAV's positions using the dynamic model defined in (15d), which can be expressed as $\mathrm{Pr}(\tilde{x}_{e_m,t_i}|\tilde{x}_{e_m,t_{i-1}},s^{(n)}_{e_m,t_i})$. The posterior associated with the predicted states is given by:

$$\pi(\tilde{x}_{e_m,t_i}) = \mathrm{Pr}(\tilde{x}^{(n)}_{e_m,t_i},s^{(n)}_{e_m,t_i}|\tilde{z}_{e_m,t_{i-1}}) = \int \mathrm{Pr}(\tilde{x}_{e_m,t_i}|\tilde{x}_{e_m,t_{i-1}},s^{(n)}_{e_m,t_i})\lambda(\tilde{x}^{(n)}_{e_m,t_{i-1}})d\tilde{x}_{e_m,t_{i-1}}, \tag{28}$$

where $\lambda(\tilde{x}^{(n)}_{e_m,t_{i-1}}) = \mathrm{Pr}(\tilde{z}_{e_m,t_{i-1}}|\tilde{x}_{e_m,t_{i-1}})$ is the diagnostic message propagated in a bottom–up manner after observing $\tilde{z}_{e_m,t_{i-1}}$ at time slot t_{i-1}. When a new observation is received, diagnostic messages can be utilized to update the UAV's belief in hidden states. The belief in continuous states can be corrected by updating the posterior using:

$$\pi(\tilde{x}_{e_m,t_i}) = \pi(\tilde{x}_{e_m,t_i}) \times \lambda(\tilde{x}^{(n)}_{e_m,t_i}). \tag{29}$$

Meanwhile, the belief in discrete states can be updated by adjusting the weights of the particles following:

$$\omega^{(n)}_{e_m,t_i} = \omega^{(n)}_{e_m,t_i} \times \lambda(\tilde{s}_{e_m,t_i}), \tag{30}$$

where $\lambda(\tilde{s}_{e_m,t_i}) = \lambda(\tilde{x}_{e_m,t_i})\mathrm{Pr}(\tilde{x}_{e_m,t_i}|s_{e_m,t_i})$.

4.3.3. Abnormality Measures and Action Update

At each level of the hierarchy, the messages that predict what should happen are compared to the sensory messages that report what is actually happening. This comparison results in several indicators of abnormalities and prediction errors. We can determine how well the current observations match the model's predictions by examining these indicators at each level. Additionally, we can use the prediction errors to figure out how to prevent these abnormalities from occurring in the future. The observations of the UAV are influenced by its actions. Thus, if an abnormality is detected, it means that the actions taken were incorrect. The UAV can use the prediction errors to make necessary corrections and prevent abnormalities in the future.

The UAV has the capability to evaluate ongoing actions by utilizing an abnormality indicator that calculates the difference between predicted states and observations. This is achieved through the calculation of the Bhattacharyya distance as follows:

$$\Upsilon_{\tilde{x}_{em,t_i}} = -ln\Big(\mathcal{BC}\big(\pi(\tilde{x}_{em,t_i}), \lambda(\tilde{x}^{(n)}_{em,t_i})\big)\Big) = -ln\int\sqrt{\pi(\tilde{x}_{em,t_i})\lambda(\tilde{x}^{(n)}_{em,t_i})}d\tilde{x}_{em,t_i}, \tag{31}$$

where $\mathcal{BC}$ is the Bhattacharyya coefficient. It is to note that during exploration, the UAV's expected states realize the target position, while during exploitation, the UAV's expected states are guided by the tokens.

The abnormality indicator defined in (31) is associated with prediction errors calculated as:

$$\mathcal{E}_{\tilde{x}_{em,t_i}} = [\tilde{x}_{em,t_i}, \dot{\mathcal{E}}_{\tilde{x}_{em,t_i}}] = [\tilde{x}_{em,t_i}, \mathrm{H}^{-1}\mathcal{E}_{\tilde{z}_{em,t_i}}], \tag{32}$$

where $\mathcal{E}_{\tilde{z}_{em,t_i}} \sim \mathcal{N}(\mu_{\mathcal{E}_{\tilde{z}_{em,t_i}}}, \Sigma_{\mathcal{E}_{\tilde{z}_{em,t_i}}})$ depicts the prediction errors computed in the observation space, which is characterized by the following statistical properties:

$$\tilde{\mu}_{\mathcal{E}_{\tilde{z}_{em,t_i}}} = \tilde{z}_{em,t_i} - \mathrm{H}\tilde{x}_{em,t_i}, \tag{33a}$$

$$\Sigma_{\mathcal{E}_{\tilde{z}_{em,t_i}}} = \mathrm{H}\Sigma_{\mathcal{E}_{\tilde{z}_{em,t_i}}}\mathrm{H}^{\intercal} + \mathrm{R}, \tag{33b}$$

where (33a) is the Kalman innovation and (33b) is the innovation covariance.

In case the UAV encounters abnormal situations, it can use prediction errors to rectify its previous actions through first-order Euler integration following:

$$a^{p}_{em,t_i} = a^{p}_{em,t_{i-1}} + \Delta_{t_i}\dot{\mu}_{\tilde{x}_{em,t_i}}, \tag{34}$$

where Δ_{t_i} is the step size.

On the other hand, the UAV can assess the discrete actions representing the targeted hotspots only after completing a full mission that includes a sequence of events. This is because the UAV needs to determine if the selected hotspots were efficiently reached in their designated order to achieve the intended goal of maximizing the sum rate. As previously stated, a series of actions (or generalized letters) form a word, and the UAV checks whether the resulting word fulfills the intended goal. Therefore, to evaluate the formed word, it is necessary to consider the cumulative abnormality indicator. This indicator adds up the abnormalities that measure the divergence between what was expected and what was observed at each event. The abnormality indicator itself is defined as:

$$\Upsilon_{\tilde{x}_{T,em}} = -ln\Big(\mathcal{BC}\big(\pi(\tilde{x}_{T,e_m}), \lambda(\tilde{z}_{T,e_m})\big)\Big) = -ln\int\sqrt{\pi(\tilde{x}_{T,e_m})\lambda(\tilde{z}_{T,e_m})}d\tilde{x}_{T,e_m}. \tag{35}$$

while the cumulative abnormality indicator is defined as follows:

$$\Upsilon_{\tilde{x}_T} = \sum_{e_m=1}^{E}\Upsilon_{\tilde{x}_{T,em}}, \tag{36}$$

where E is the total number of events that occur during the flight mission, which lasts for a duration of T.

In case the UAV detects a high cumulative abnormality, this indicates that the entire mission was unsuccessful. In this case, the UAV must correct the action selection process by updating its strategy of forming the word. This can be done by updating the active inference table defined in (21) as follows:

$$\Pr(a^{l}_{e_m}|w^{c}_{e_m}) = \Pr(a^{l}_{e_m}|w^{c}_{e_m}) - \gamma, \tag{37}$$

where the gradient γ determines the amount by which the probability should be decreased.

Additionally, if the mission is successful with minimal abnormalities, the transition matrix specified in (12) will be modified as follows:

$$\Pr(\tilde{l}_i|\tilde{l}_j) = \Pr(\tilde{l}_i|\tilde{l}_j) + \bar{\gamma}, \tag{38}$$

where i and j are part of the successful word representing the sequence of hotspots visited by the UAU during its successful mission and $\bar{\gamma}$ is the gradient that determines the amount by which the probability should be increased.

5. Numerical Results and Discussion

In this section, we will thoroughly assess how well the proposed framework performs in designing a trajectory for the UAV that effectively allows it to attain the highest total sum-rate possible with the cell. In our simulations, we examined a situation where a single UAV is providing service to several users who are located in different hotspots across a square geographic area of 1000×1000 m^2. The main simulation parameters are listed in Table 2. It is assumed that the altitude of the UAV remains constant at $z_u = 100$ m [65]. Throughout the training process, we place a total of $N = 80$ hotspots in various random locations across the geographical area. The frequency of user presence and requests within each hotspot adheres to the Poisson distribution. We generated a training set $\mathcal{D}$ that consists of M examples corresponding to different realizations. Each realization (m) consists of seven hotspots picked randomly from the N total hotspots, and the users' requests in each hotspot were generated following Poisson distribution. The TSPWP method was used to solve the M examples in $\mathcal{D}$, generating M trajectories (TSPWP instances) and M sequences of the order in which the hotspots are visited, which were saved in $\mathcal{L}^+$ and $\mathcal{Q}^+$, respectively.

Table 2. Simulation Parameters.

Parameter	Value	Parameter	Value
P_u	1 W	α	2
B_{RB}	180 KHz	σ^2	-104 dBm
μ_{Los}	3	μ_{NLos}	23
N	80	M	1000

We evaluated the TSPWP performance by conducting a thorough analysis of completion time and cost with profit metrics for different numbers of hotspots to determine the optimal α and β values mentioned in (6a). In Figure 8, we see how the completion time of TSPWP was impacted by various α and β values, as well as changes in the number of hotspots. Meanwhile, Figure 9 displays the TSPWP performance in terms of cost with profit for different α and β settings while also altering the number of hotspots. It is evident from Figure 8 that the completion time increases as the number of hotspots increases, as having more hotspots makes the trajectory longer. It is worth noting that the cost with profit rose gradually as the number of hotspots increased, especially between five and twenty, as shown in Figure 9. However, after twenty hotspots, the cost with profit slightly rose due to the reduction of profit (i.e., the accumulated sum-rate) from the cost (i.e., the traveling distance between the hotspots). This effect became stable for higher hotspots and had a minimal impact on the overall cost with profit. By analyzing the data, we have found that the ideal α and β values for achieving both minimal completion time and maximum profit with cost are 0.9 and 0.1, respectively. Therefore, we will use these values when implementing TSPWP with 2-OPT.

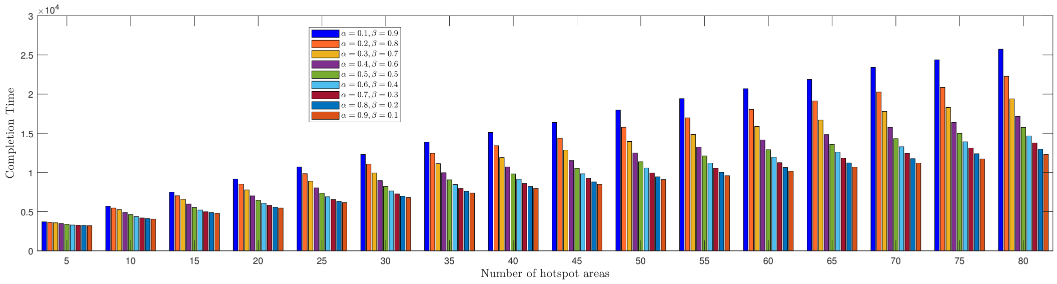


Figure 8. TSPWP's completion time performance for varying alpha and beta values, as well as changes in the number of hotspots.

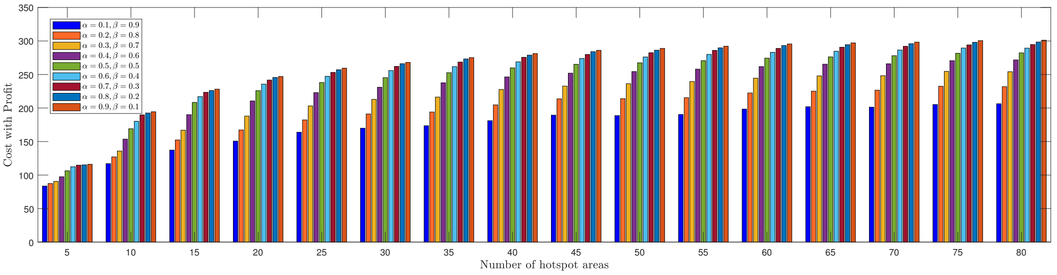
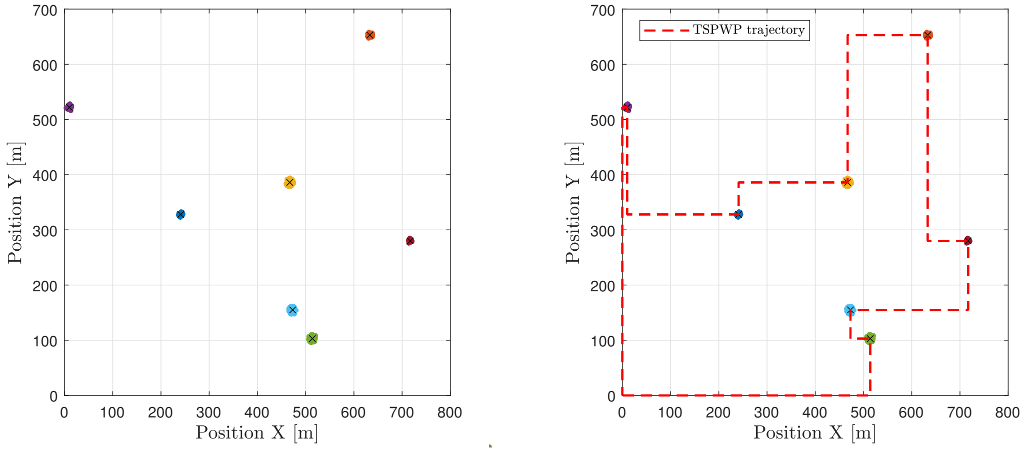


Figure 9. TSPWP's cost with profit performance for varying alpha and beta values, as well as changes in the number of hotspots.

To solve each realization m, we used the TSPWP with $\alpha = 0.9$ and $\beta = 0.1$, as previously mentioned. The TSPWP with 2-OPT gave us the solution (i.e., the TSPWP instance), which includes the trajectory and the order of the hotspots to visit. We then created two sub-dictionaries from the M TSPWP instances. The first sub-dictionary comprised all the words that made up the TSPWP trajectories, which use letters to represent the hotspots (explained in Section 4.2.1). The second sub-dictionary contained all the tokens that showed the path between two adjacent letters (hotspots), as described in Section 4.2.1.

In the example shown in Figure 10a, there is one realization with seven hotspots scattered randomly in the geographic area. Each hotspot has some active users who need resources. The goal is to start from the initial station at the origin, visit each hotspot only once, serve the users there, and then return to the origin within a specific time frame. The realization depicted in Figure 10a is used as input to the TSPWP with 2-OPT method. Latter will produce the TSPWP instance, which includes the trajectory and the order of visited hotspots, as demonstrated in Figure 10b. To create the global dictionary, TSPWP instances from M examples are utilized, which include sub-dictionary 1 and sub-dictionary 2. Sub-dictionary 1 records the events that take place during the flight mission, such as when the UAV reaches hotspot j after departing from hotspot i. The process of detecting different events and forming a word representing the sequence of hotspots served during a flight mission is illustrated in Figure 11a. In this process, hotspots are considered as letters, and the full trajectory represents a word. The first event occurs after reaching the letter "g" starting from "o". The second event occurs after reaching "f" from "g", and so on for the third and subsequent events. The final event occurs when the UAV returns to the initial location, represented by the letter "o", starting from "a". Therefore, the word describing the mission is defined as "w = o, g, f, e, d, c, b, a, o". By contrast, if we cluster the trajectory data (which include positions and velocities), we can see the resulting clusters in Figure 11b. Each event that was previously detected will be linked to the set of clusters that form the path from one letter to another, as illustrated in Figure 11b. A token is created for each event, and all the tokens are combined to form the resulting word, which represents the path followed during the mission. Throughout the training process, the same procedure is carried out for M examples in order to create the words that indicate the sequence of targeted hotspots and the words that describe the movement from one hotspot to the next.

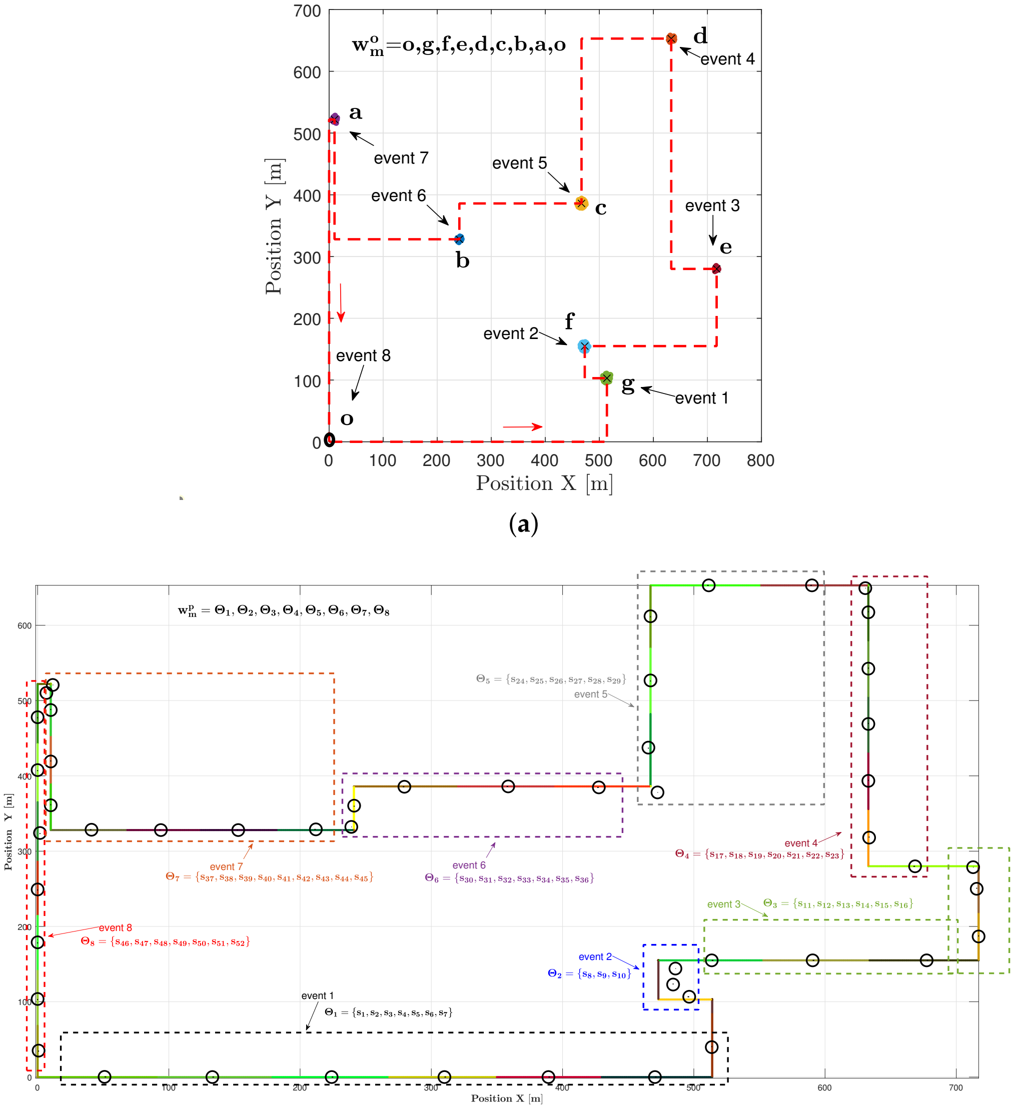
These two sets of words are coupled statistically to create a world model that the UAV will use during the active inference (testing) process to plan a suitable trajectory based on encountered situations (realizations).



(**a**) (**b**)

Figure 10. An example of one realization: (**a**) Seven hotspots scattered randomly across the geographical area labeled with different letters, and each has a varying number of active users requesting service. (**b**) The trajectory provided by the TSPWP.

Let us take a look at how a UAV, using active inference, completes a mission. For instance, suppose there are 11 hotspots in a given testing scenario as shown in Figure 12. The UAV will rely on the world model, made up of two sub-dictionaries, that it learned during training to successfully navigate the testing scenario. First, the UAV examines the current letters and matches them against the words listed in sub-dictionary 1. This process helps to establish how closely they resemble each other in the current testing scenario. After that, the UAV chooses the closest word from the dictionary and uses it as a starting point to create the initial graph. The goal is to expand the graph by adding new letters to form a word that enables an efficient trajectory to reach all hotspots (letters) and serve their users as quickly as possible. To achieve this, one letter is added during each iteration, with the number of iterations depending on the size of the reference graph and the number of new letters required to include all available letters in the current configuration. To update the graph and make it directed, one link must be removed from the reference graph, and two links must be added to the newly added letter or node at every iteration. The transition matrix, which encodes the probabilistic relationships among the letters, is crucial at each step and can be found in Figure 13. This matrix determines whether it is possible to transition from a letter already present in the reference graph to the newly added letter. The transition matrix is learned after solving M examples during training and allows for the generation of words based on probability entries.



(**a**)

(**b**)

Figure 11. The process of forming the dictionary: (**a**) The events that have been occurred during the flight and the generated word consisting of the letters visited by the UAV. Event 1 occurs after reaching letter g starting from letter o. Event 2 occurs after reaching letter f from g. Event 3 occurs after reaching letter e from f. Event 4 occurs after reaching letter d from e. Event 5 occurs after reaching letter c from d. Event 6 occurs after reaching letter b from c. Event 7 occurs after reaching letter a from b. Event 8 occurs after returning to the origin from a. (**b**) The clusters obtained after clustering the trajectory. Clusters are labeled as letters. The generated tokens each consist of several letters corresponding to a specific event and thus explaining the path to follow between two adjacent letters.

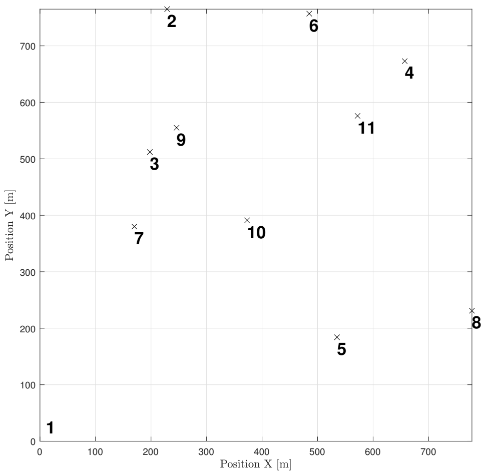


Figure 12. An example of a testing realization including 11 hotspots.

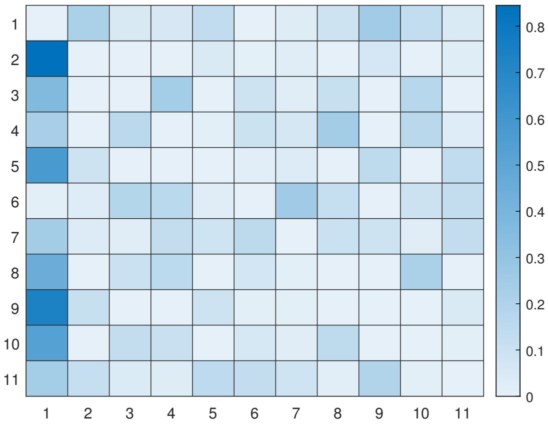
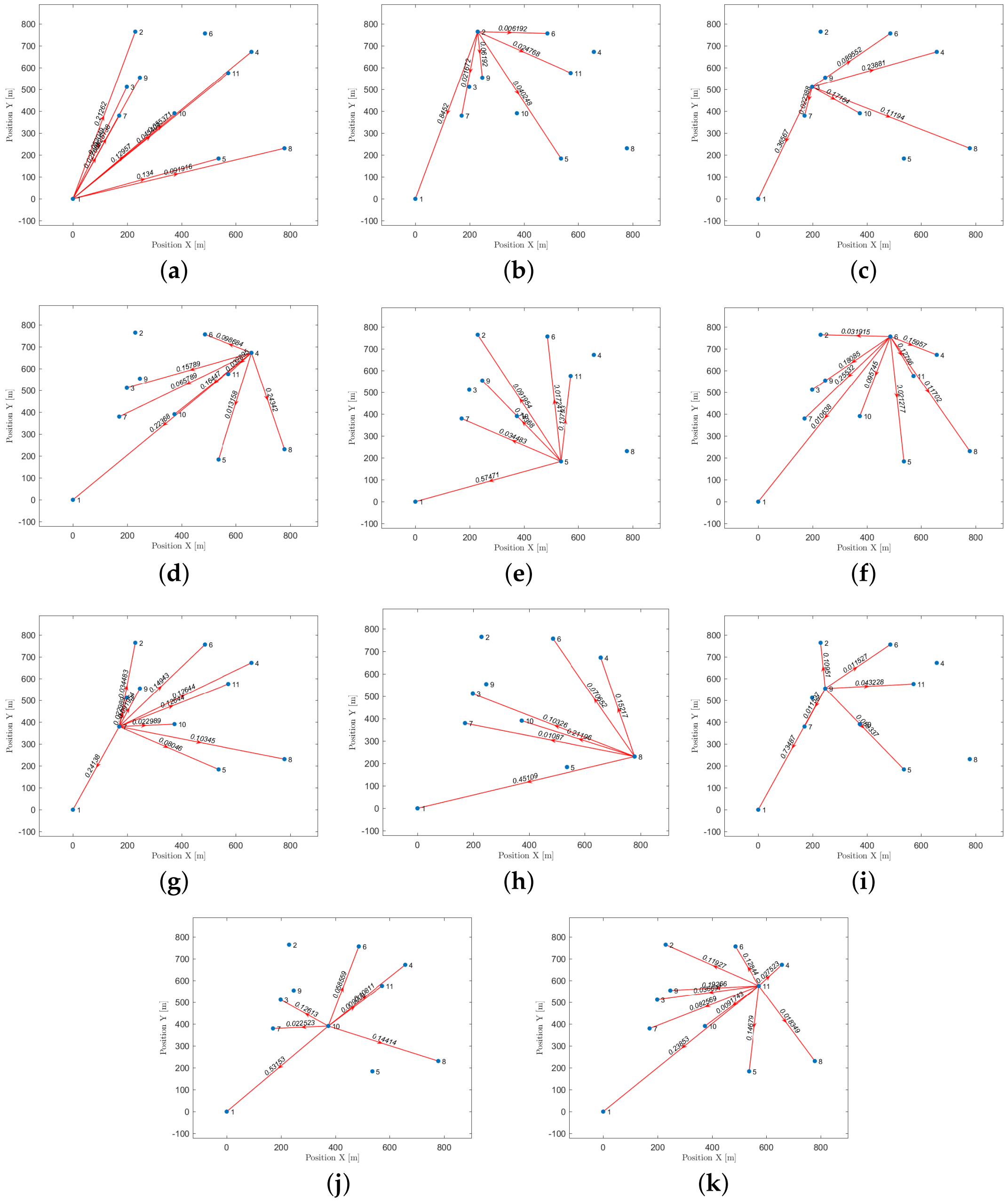


Figure 13. The transition matrix encoding the probabilities of passing from one letter to another based on the examples solved during training.

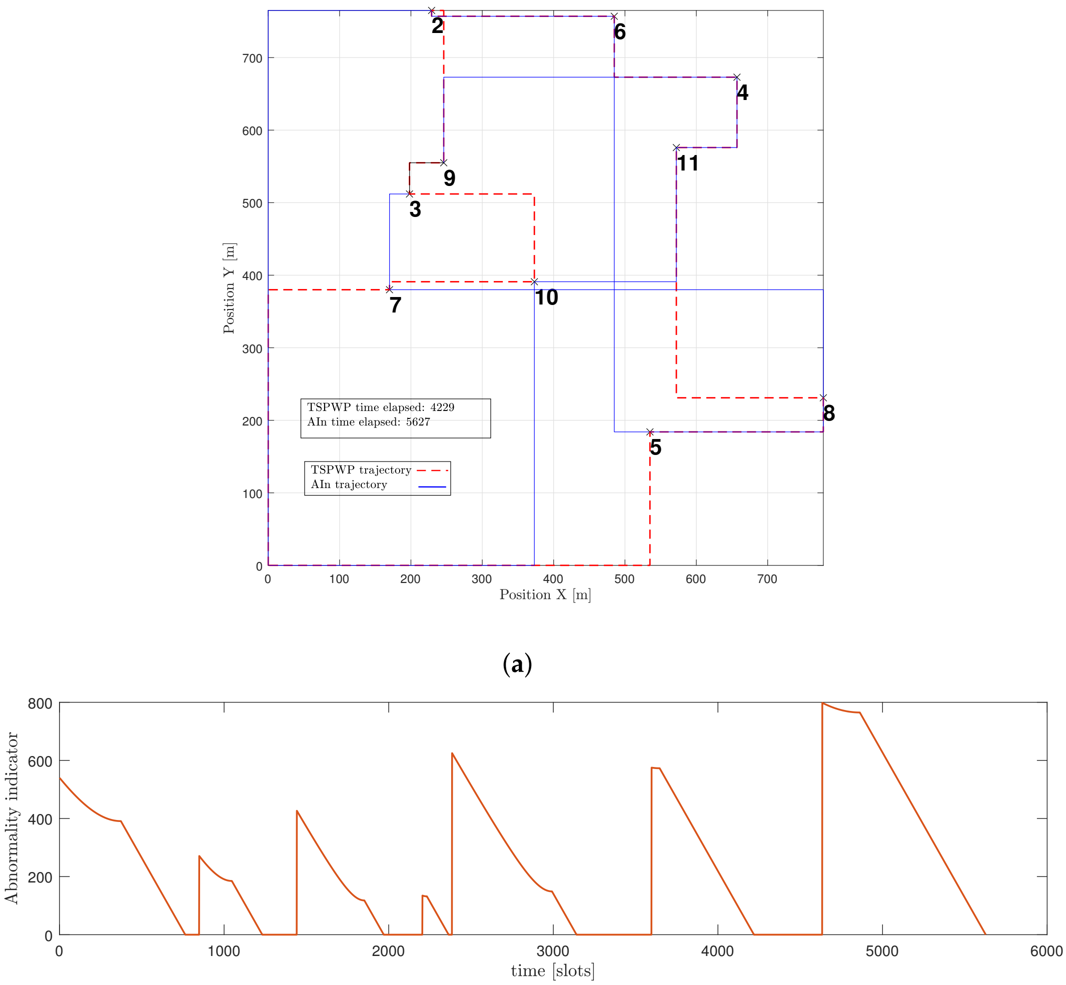
Figure 14 displays all the available pathways from the 11 hotspots to other letters. Depending on the current letter, one can determine which letters are reachable. For instance, if one starts at letter 1 (the initial location), one cannot transition to letter 6, but one can transition to the other 9 letters with varying probabilities. Similarly, if one reaches letter 2, one cannot go towards letters 3, 4, 8, and 10, and so on. It is worth noting that the probability values provided by the world model prevent unnecessary transitions that will not help the UAV reach its desired goal.

The example shown in Figure 15a expresses a word generated by the UAV through the proposed method but before it fully converged. The generated word is not optimal as it contains hotspots in the wrong order, which causes the mission to take longer and increases the time needed to return to the initial location. Furthermore, Figure 15b shows that the UAV detected abnormalities during most of the operation events. When the UAV detects abnormalities in its position, it is usually because it is not close enough to its goal. The UAV aims for a specific letter that represents its target. It is drawn towards that goal and then assesses its distance from the goal after each continuous action that represents its velocity. If there are any abnormalities, the UAV can use prediction errors to correct its actions and adjust its path to reach the targeted letter. For instance, during event 1, the UAV perceived high abnormalities and prediction errors while it was still far from the intended letter, with the starting letter being 1 and the targeted being 10. However, utilizing the prediction error, the UAV was able to adjust its actions and reach the destination faster. This resulted in the abnormality signals gradually decreasing until they reached zero, indicating that the UAV had indeed arrived at the targeted destination.



(**a**) (**b**) (**c**) (**d**) (**e**) (**f**) (**g**) (**h**) (**i**) (**j**) (**k**)

Figure 14. The transition probabilities suggested by the world model to generate a word that might solve the current realization: (**a**) Possible letters to target starting from letter 1. (**b**) Possible letters to target starting from letter 2. (**c**) Possible letters to target starting from letter 3. (**d**) Possible letters to target starting from letter 4. (**e**) Possible letters to target starting from letter 5. (**f**) Possible letters to target starting from letter 6. (**g**) Possible letters to target starting from letter 7. (**h**) Possible letters to target starting from letter 8. (**i**) Possible letters to target starting from letter 9. (**j**) Possible letters to target starting from letter 10. (**k**) Possible letters to target starting from letter 11.

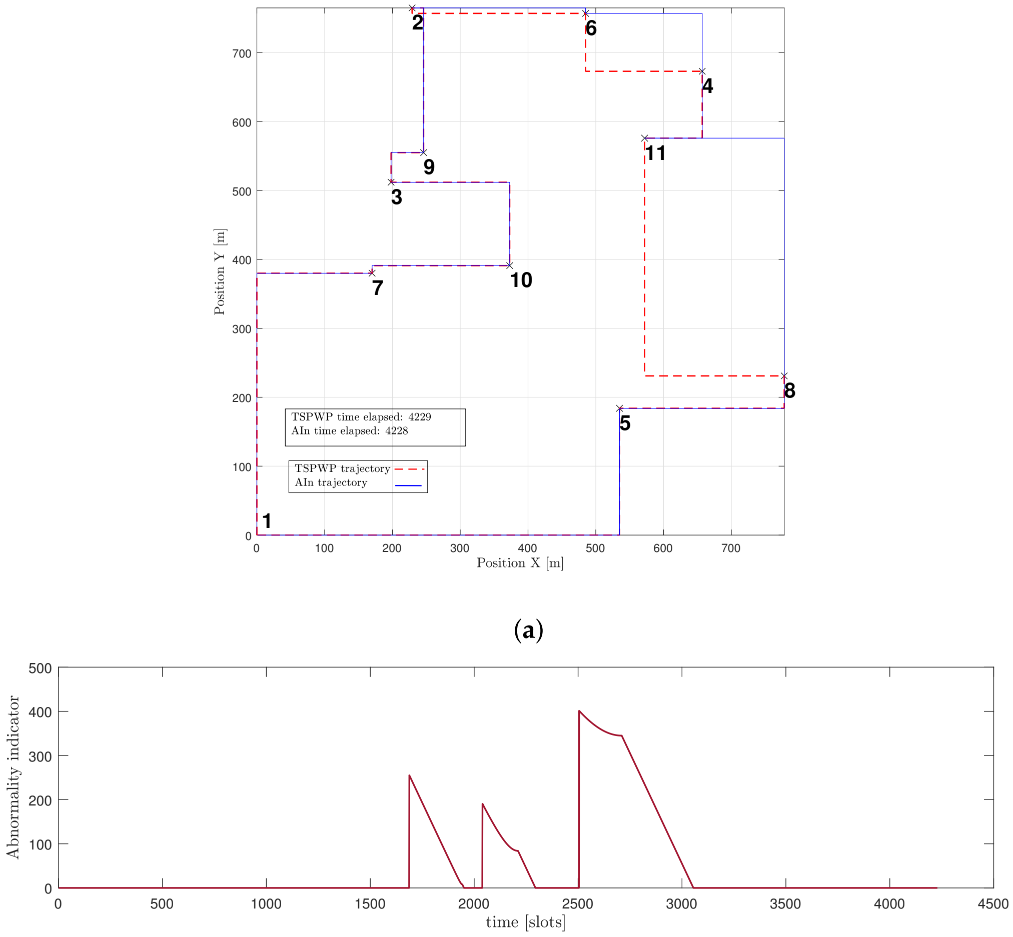


(**a**)

(**b**)

Figure 15. A word generated using active inference before convergence: (**a**) The trajectory followed by the UAV based on active inference before the convergence. (**b**) The abnormalities that occurred during the flight mission.

Figure 16a presents another example of a word created by the UAV after convergence. The proposed approach enabled the UAV to design a trajectory that is comparable to the one generated by the TSPWP with 2-OPT, with a similar completion time. It is noticeable that the UAV was successful in reducing high abnormalities in various events, as depicted in Figure 16b, compared to the example shown before convergence. This reduction is due to the UAV's ability to differentiate between similar events encountered before and deduce the optimal path immediately.



(**a**)

(**b**)

Figure 16. A word generated using active inference after convergence: (**a**) The trajectory followed by the UAV based on active inference before the convergence. (**b**) The abnormalities that occurred during the flight mission.

Figure 17 displays the updated transition matrix for 11 letters, which includes corrected probability entries detailing the possible transitions between the available letters. This updated transition matrix was rectified using the one exhibited in Figure 13.

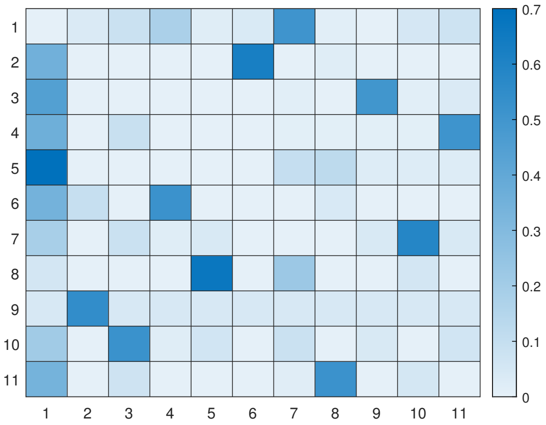
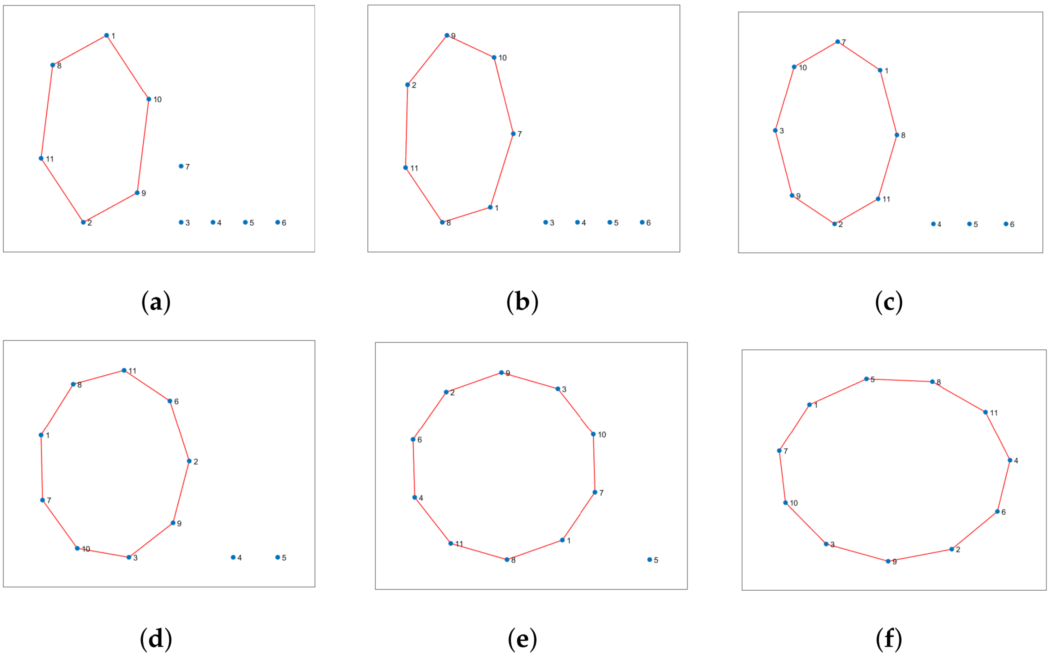


Figure 17. The updated transition matrix encoding the probabilities of passing from one letter to another after convergence to solve the example shown in Figure 12.

The process of creating new words is shown in Figure 18. The first step is to select a reference word from the dictionary by comparing the available letters in the current realization with the encoded words in the dictionary. The UAV selects the word with the highest probability of being a match based on the similarity of its letters to the available ones. The matching letters from the most similar word are then used as a reference for creating new words. This reference word is represented graphically as a closed loop, as demonstrated in Figure 18a. The initial graph is expanded by adding one letter at a time, as illustrated in the figure. This insertion approach dramatically reduces the likelihood of the UAV needing to determine the optimal visiting order. For instance, if there are 11 nodes to visit, and each node must be visited only once, there are approximately 11! (∼39 million) possible word combinations for which to find the correct order, which is a time-consuming and challenging task, particularly when using a trial-and-error method. However, the proposed word formation mechanism decreases the number of possible combinations from 11! to just 40. In Figure 18a, there are six potential ways to create a new word by adding the first letter to the reference graph. Figure 18b has seven possible words, while the other graphs feature eight, nine, and ten options. The total number of combinations is 40, which is calculated by adding the number of edges in each graph.

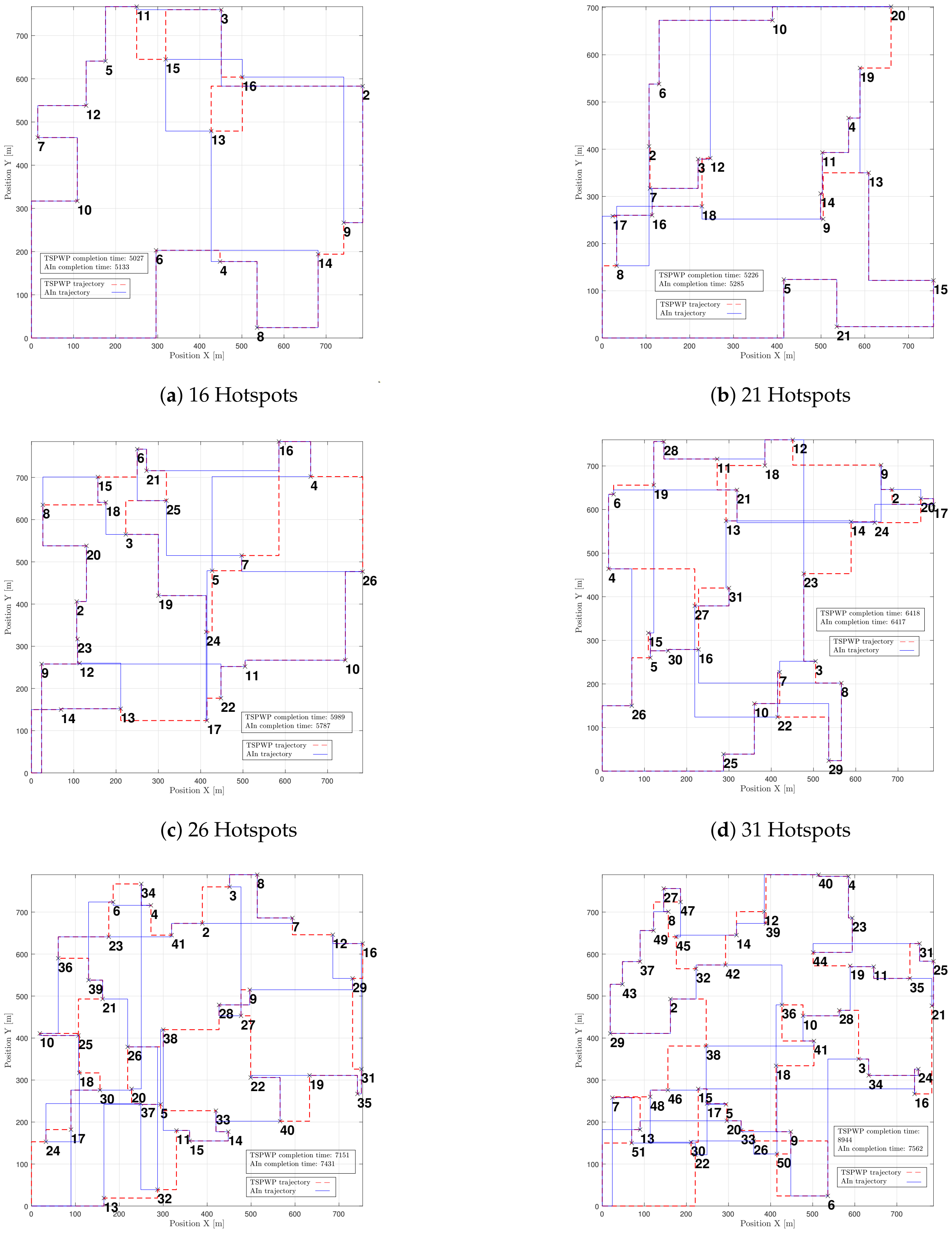


(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 18. This is a graphic explanation of the process for creating new words from a base word found in the dictionary: (**a**) The reference word is represented graphically, and the new letters encountered in the new situation should be added to the reference graph. (**b**) The updated graph (word) after adding letter 7. (**c**) The updated graph (word) after adding letter 3. (**d**) The updated graph (word) after adding letter 6. (**e**) The updated graph (word) after adding letter 4. (**f**) The updated graph (word) after adding letter 5.

In Figure 19, one can see different examples with different numbers of hotspot areas. The trajectories generated by the proposed method (AIn) and the TSPWP using 2-OPT are also shown, along with their respective completion times. It is evident that the proposed approach produces alternative solutions when compared to the TSPWP with 2-OPT. In some cases, it also results in a quicker completion time as shown in Figure 19c,d,f. This highlights the adaptability of the proposed method in deriving reasonable solutions that surpass those of the TSPWP.



(**a**) 16 Hotspots

(**b**) 21 Hotspots

(**c**) 26 Hotspots

(**d**) 31 Hotspots

(**e**) 41 Hotspots

(**f**) 51 Hotspots

Figure 19. The figure displays various examples with varying numbers of hotspot areas, along with the solutions produced by the proposed method (AIn) and the TSPWP utilizing 2-OPT.

As shown in Figure 20, we tested the scalability of the proposed method (AIn) and compared the cumulative sum-rate convergence for various hotspots. We observed that as the number of hotspots increased, the cumulative sum-rate also increased. However, it took longer to find the best solution and reach convergence with more hotspots. This is because there were more possible generated words to test, which takes longer. By contrast, Figure 21 shows the cumulative abnormality for various numbers of hotspots. The trend of the cumulative abnormality is contrary to the cumulative sum-rate. It begins with high values and gradually decreases until reaching quasi-zero at convergence. As the number of hotspots increases, the time taken to reach quasi-zero abnormality also increases.

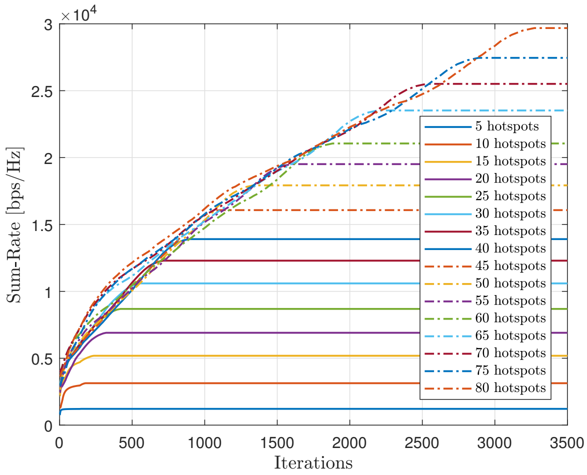


Figure 20. Convergence of the proposed approach (AIn) in terms of sum-rate for different numbers of hotspots.

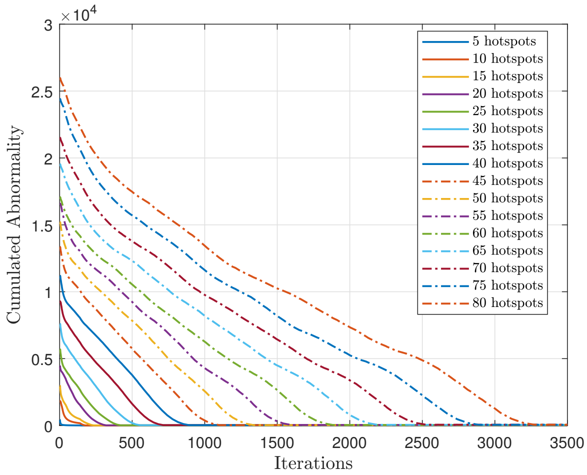


Figure 21. Cumulative abnormality convergence of the proposed approach (AIn) for different numbers of hotspots.

In Figure 22, we can see the average sum-rate of the proposed method at convergence for various numbers of hotspots, compared to the analytical sum-rate. It is clear that the proposed approach achieves the expected analytical sum-rate after convergence, regardless of the number of hotspots.

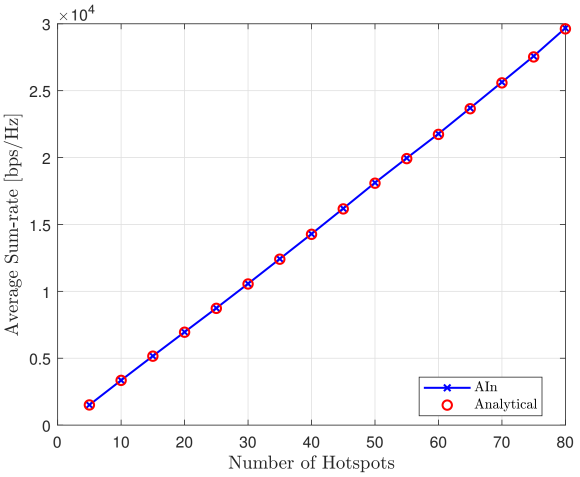


Figure 22. The average sum-rate of the proposed approach (AIn) compared to the analytical value for various number of hotspots.

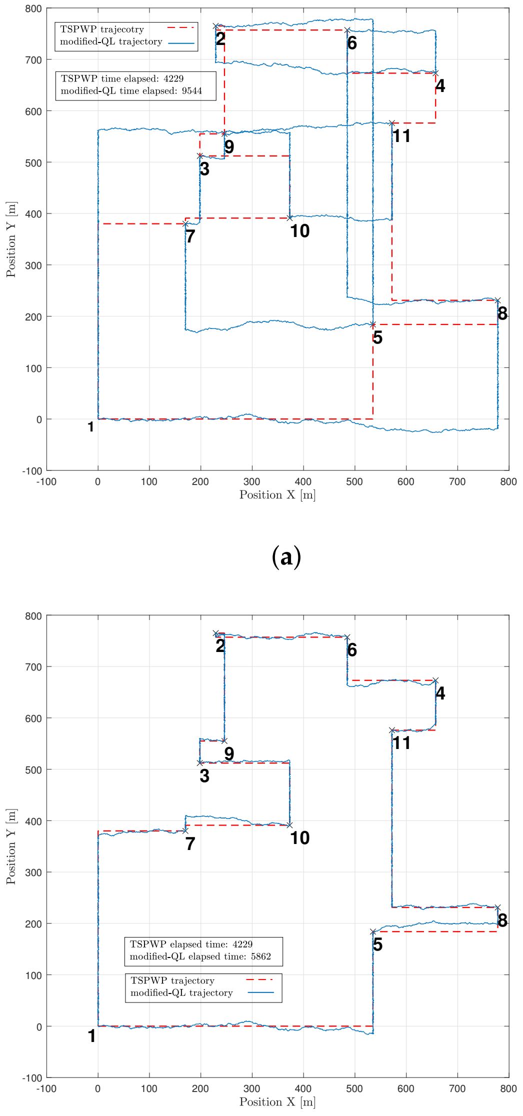
Comparison with Modified Q-Learning

In this section, we compare the performance of the proposed approach (AIn) with a modified version of the conventional Q-learning (QL) [66]. To ensure a fair comparison, the modified-QL follows the same logic as the proposed approach. Thus, the modified version uses two probabilistic q-tables—one for mapping discrete states (hotspots) to discrete actions (targeted letters) and another for mapping discrete environmental regions to continuous actions (velocity). Unlike traditional QL, the q-values in these tables are represented as probability entries that range between 0 and 1.

As in the proposed method, we can see that the discrete states stand for the letters, and the discrete environmental regions stand for the clusters. In addition, the available letters during a specific realization make up the discrete action space, while four continuous actions representing different directions (*Up*, *Down*, *Left*, *Right*) make up the continuous action space. The reward function in modified-QL was designed using the TSPWP instances. If the modified-QL behaves similarly to the TSPWP, it will receive a positive reward (+1). Otherwise, the reward is zero.

In Figure 23, an example similar to the one in Figure 10a is shown to illustrate how the modified-QL algorithm solved the mission both before and after convergence. Prior to convergence (Figure 23a), the modified-QL selected the wrong order of letters to visit, leading to a longer completion time. However, after convergence (Figure 23b), the algorithm discovered the correct order of letters, resulting in a reduced completion time, although it still fell short of the completion time achieved by the TSPWP with 2-OPT due to a slight deviation from the correct path. It is important to note that the agent's movement was limited to traveling between two boundaries to simplify the process, which reduced the environmental states it could discover. Consequently, the modified-QL agent's movements were guided by the TSPWP through positive and zero rewards.

Figure 24 displays the gathered sum-rate in relation to the number of iterations, providing insight into the modified-QL's overall performance and scalability with varying numbers of hotspots. It is clear that as the number of hotspots increases, both the collected sum-rate and the time to converge will also increase with the modified-QL. Despite requiring more iterations, the modified-QL achieved the same sum-rate at convergence as the proposed method.



(**a**)

(**b**)

Figure 23. An example of the realization shown in Figure 12: (**a**) The trajectory followed by the UAV using the modified-QL before convergence. (**b**) The trajectory followed by the UAV using the modified-QL after convergence.

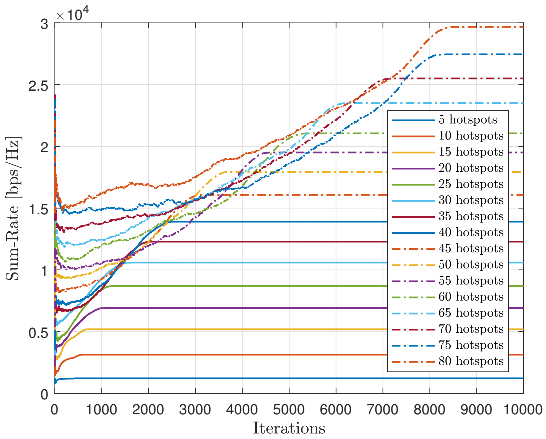


Figure 24. Convergence of the modified-QL in terms of sum-rate for different numbers of hotspots.

In Figure 25, we compared the convergence time of the proposed method (AIn) to that of the modified-QL, as we varied the number of hotspots. The results show that the proposed method requires less time to converge than the modified-QL. This difference was more noticeable as we increased the number of hotspots, with the gap between the two trends increasing. The modified-QL took longer to converge as we increased the number of hotspots, and it did so at a faster rate than AIn due to its random nature, which led to a higher number of possible words to try compared to AIn.

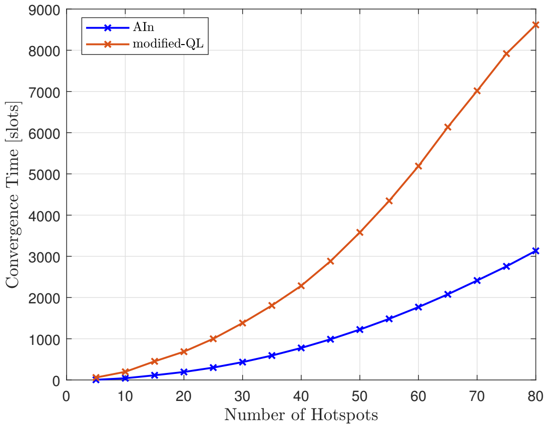


Figure 25. The convergence time of the proposed approach (AIn) compared to the convergence time of the modified-QL for different numbers of hotspots.

Figure 26 compares the completion time of our proposed method, AIn, to that of modified-QL and TSPWP with 2-OPT as the number of hotspots varies. The results show that modified-QL took longer to complete the missions due to slight deviations from the reference trajectories designed by TSPWP. These deviations were caused by the random actions performed before the convergence. On the other hand, AIn is able to complete missions faster than modified-QL thanks to its ability to deduce certain paths based on the world model and calculate prediction errors to correct continuous actions. This allows AIn to reach the target destination more quickly.

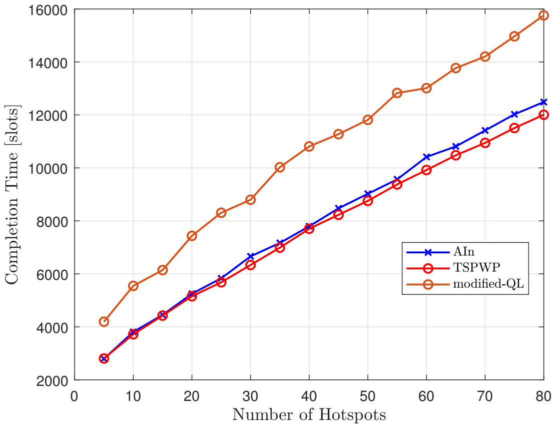


Figure 26. The performance of the proposed approach (AIn) in terms of completion time after convergence compared with TSPWP for different numbers of hotspots.

6. Conclusions and Future Directions

This paper studied the trajectory design problem in UAV-assisted wireless networks. In the considered system, a single UAV provides on-demand uplink communication service to ground users by flying around the environment. To solve this problem, we have proposed a goal-directed method based on active inference, consisting of two computation units. The first unit builds a world model to understand the surrounding environment, while the second unit makes decisions to minimize a cost function and achieve preferred outcomes. The world model represents a global dictionary that has been learned from instances generated by the TSPWP using a 2-OPT algorithm to solve various offline examples. The dictionary includes letters for hotspots, tokens for local paths, and words for complete trajectories and order of hotspots. By analyzing the dictionary, we can understand the decision maker's grammar, specifically the TSPWP strategy, and how it utilizes the available letters to form tokens and words. To accurately represent the properties of TSPWP graphs at different levels of abstraction and time scales, we developed a novel hierarchical representation called the coupled multi-scale generalized dynamic Bayesian network (C-MGDBN) that structures the gathered knowledge (i.e., the global dictionary).

Simulation results indicate that the proposed method performs better than the traditional Q-learning algorithm. It provides quick, stable, and alternative solutions with good generalization capabilities. Additionally, the results demonstrate that our approach can be scaled up to larger instances, despite being trained on smaller ones, proving its effectiveness in generalization. Furthermore, we have proven that our method can solve a complex problem (known as NP-hard) by significantly reducing the number of actions the UAV needs to take to solve a specific example.

In future work, we plan to tackle the challenge of determining the optimal solution when there are more hotspot areas but a fixed flight duration. We will also address the challenge of new hotspots appearing and old ones disappearing while the UAV is completing its current mission. Lastly, we will investigate coupling at the word scale in future studies.

Author Contributions: Conceptualization, A.K. and C.R.; methodology, A.K.; software, A.K.; validation, A.K., K.K. and C.R.; formal analysis, A.K., K.K., C.R.; investigation, A.K. and K.K.; resources, A.K. and K.K.; data curation, A.K.; writing—original draft preparation, A.K. and K.K.; writing—review and editing, A.K., K.K., L.M., M.M. and C.R.; visualization, A.K. and K.K.; supervision, M.M. and C.R.; project administration, L.M.; funding acquisition, L.M.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned aerial vehicle
LoS	Line of sight
NOMA	Non-orthogonal multiple access
GPS	Global positioning system
IoT	Internet of things
AI	Artificial intelligence
ML	Machine learning
RL	Reinforcement learning
TSPWP	Travelling salesman problem with profits
GDBN	Generalized dynamic Bayesian network
C-MGDBN	Coupled multi-scale generalized dynamic Bayesian network
DP	Dynamic programming
WSN	Wireless sensor node
MILP	Mixed integer linear programming
TSP	Travelling salesman problem

GA	Genetic algorithm
PSO	Particle swarm optimization
ACO	Ant colony optimization
QoE	Quality of experience
QL	Q-learning
DQL	Deep Q-learning
FBS	Flying base station
GU	Ground users
RB	Resource block
OFDMA	Orthogonal frequency division multiple access
NLoS	Non-line-of-sight
AWGN	Additive White Gaussian Noise
C-GDBN	Coupled Generalized dynamic Bayesian network
M-GDBN	Multi-scale generalized dynamic Bayesian network
GNG	Growing neural gas
POMDP	Partially observable Markov decision process
KF	Kalman filter
PF	Particle filter

References

1. Li, B.; Fei, Z.; Zhang, Y. UAV Communications for 5G and Beyond: Recent Advances and Future Trends. *IEEE Internet Things J.* **2019**, *6*, 2241–2263. [CrossRef]
2. Krayani, A.; Baydoun, M.; Marcenaro, L.; Gao, Y.; Regazzoni, C.S. Smart Jammer Detection for Self-Aware Cognitive UAV Radios. In Proceedings of the 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications, London, UK, 31 August–1 September 2020; pp. 1–7. [CrossRef]
3. Zhou, Y.; Yeoh, P.L.; Chen, H.; Li, Y.; Schober, R.; Zhuo, L.; Vucetic, B. Improving Physical Layer Security via a UAV Friendly Jammer for Unknown Eavesdropper Location. *IEEE Trans. Veh. Technol.* **2018**, *67*, 11280–11284. [CrossRef]
4. Khawaja, W.; Ozdemir, O.; Guvenc, I. UAV Air-to-Ground Channel Characterization for mmWave Systems. In Proceedings of the 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, ON, Canada, 24–27 September 2017; pp. 1–5. [CrossRef]
5. Cheng, F.; Zhang, S.; Li, Z.; Chen, Y.; Zhao, N.; Yu, F.R.; Leung, V.C.M. UAV Trajectory Optimization for Data Offloading at the Edge of Multiple Cells. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6732–6736. [CrossRef]
6. Osseiran, A.; Boccardi, F.; Braun, V.; Kusume, K.; Marsch, P.; Maternia, M.; Queseth, O.; Schellmann, M.; Schotten, H.; Taoka, H.; et al. Scenarios for 5G mobile and wireless communications: The vision of the METIS project. *IEEE Commun. Mag.* **2014**, *52*, 26–35. [CrossRef]
7. Zeng, Y.; Zhang, R.; Lim, T.J. Wireless communications with unmanned aerial vehicles: Opportunities and challenges. *IEEE Commun. Mag.* **2016**, *54*, 36–42. [CrossRef]
8. Yang, D.; Wu, Q.; Zeng, Y.; Zhang, R. Energy Tradeoff in Ground-to-UAV Communication via Trajectory Design. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6721–6726. [CrossRef]
9. Wang, Q.; Chen, Z.; Li, H.; Li, S. Joint Power and Trajectory Design for Physical-Layer Secrecy in the UAV-Aided Mobile Relaying System. *IEEE Access* **2018**, *6*, 62849–62855. [CrossRef]
10. Yi, W.; Liu, Y.; Bodanese, E.; Nallanathan, A.; Karagiannidis, G.K. A Unified Spatial Framework for UAV-Aided MmWave Networks. *IEEE Trans. Commun.* **2019**, *67*, 8801–8817. [CrossRef]
11. Kandeepan, S.; Gomez, K.; Reynaud, L.; Rasheed, T. Aerial-terrestrial communications: Terrestrial cooperation and energy-efficient transmissions to aerial base stations. *IEEE Trans. Aerosp. Electron. Syst.* **2014**, *50*, 2715–2735. [CrossRef]
12. Zhang, S.; Zeng, Y.; Zhang, R. Cellular-Enabled UAV Communication: A Connectivity-Constrained Trajectory Optimization Perspective. *IEEE Trans. Commun.* **2019**, *67*, 2580–2604. [CrossRef]
13. Hammami, M.; Chaieb, C.; Ajib, W.; Elbiaze, H.; Glitho, R. UAV-Assisted Wireless Networks for Stringent Applications: Resource Allocation and Positioning. In Proceedings of the 2023 IEEE Wireless Communications and Networking Conference (WCNC), Scotland, UK, 26–29 March 2023; pp. 1–6. [CrossRef]
14. Zhang, X.; Hu, Y.; Chang, Z.; Min, G. AoI-Minimal Power and Trajectory Optimization for UAV-Assisted Wireless Networks. In Proceedings of the 2023 IEEE Wireless Communications and Networking Conference (WCNC), Scotland, UK, 26–29 March 2023; pp. 1–6. [CrossRef]
15. Wang, Y.; Fu, S.; Yao, C.; Zhang, H.; Yu, F.R. Caching Placement Optimization in UAV-assisted Cellular Networks: A Deep Reinforcement Learning based Framework. *IEEE Wirel. Commun. Lett.* **2023** . [CrossRef]
16. Wang, Y.; Yan, M.; Feng, G.; Qin, S.; Wei, F. Autonomous On-Demand Deployment for UAV Assisted Wireless Networks. *IEEE Trans. Wirel. Commun.* **2023** . [CrossRef]

17. Krayani, A.; Alam, A.S.; Marcenaro, L.; Nallanathan, A.; Regazzoni, C. A Novel Resource Allocation for Anti-Jamming in Cognitive-UAVs: An Active Inference Approach. *IEEE Commun. Lett.* **2022**, *26*, 2272–2276. [CrossRef]
18. Yuan, X.; Yang, T.; Hu, Y.; Xu, J.; Schmeink, A. Trajectory Design for UAV-Enabled Multiuser Wireless Power Transfer With Nonlinear Energy Harvesting. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 1105–1121. [CrossRef]
19. Li, L.; Li, W.; Wang, J.; Chen, X.; Peng, Q.; Huang, W. UAV Trajectory Optimization for Spectrum Cartography: A PPO Approach. *IEEE Commun. Lett.* **2023**, *27*, 1575–1579. [CrossRef]
20. Wang, J.; Wang, X.; Liu, X.; Cheng, C.T.; Xiao, F.; Liang, D. Trajectory Planning of UAV-enabled Data Uploading for Large-scale Dynamic Networks: A Trend Prediction Based Learning Approach. *IEEE Trans. Veh. Technol.* **2023**, *72*, 8272–8277. [CrossRef]
21. Yin, D.; Yang, X.; Yu, H.; Chen, S.; Wang, C. An Air-to-Ground Relay Communication Planning Method for UAVs Swarm Applications. *IEEE Trans. Intell. Veh.* **2023**, *8*, 2983–2997. [CrossRef]
22. Chen, G.; Zhai, X.B.; Li, C. Joint Optimization of Trajectory and User Association via Reinforcement Learning for UAV-Aided Data Collection in Wireless Networks. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 3128–3143. [CrossRef]
23. Zhang, Z.; Xu, C.; Li, Z.; Zhao, X.; Wu, R. Deep Reinforcement Learning for Aerial Data Collection in Hybrid-Powered NOMA-IoT Networks. *IEEE Internet Things J.* **2023**, *10*, 1761–1774. [CrossRef]
24. Zhu, B.; Bedeer, E.; Nguyen, H.H.; Barton, R.; Gao, Z. UAV Trajectory Planning for AoI-Minimal Data Collection in UAV-Aided IoT Networks by Transformer. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 1343–1358. [CrossRef]
25. Afifi, G.; Gadallah, Y. Cellular Network-Supported Machine Learning Techniques for Autonomous UAV Trajectory Planning. *IEEE Access* **2022**, *10*, 131996–132011. [CrossRef]
26. Hu, S.; Yuan, X.; Ni, W.; Wang, X. Trajectory Planning of Cellular-Connected UAV for Communication-Assisted Radar Sensing. *IEEE Trans. Commun.* **2022**, *70*, 6385–6396. [CrossRef]
27. Qin, Y.; Zhang, Z.; Li, X.; Huangfu, W.; Zhang, H. Deep Reinforcement Learning Based Resource Allocation and Trajectory Planning in Integrated Sensing and Communications UAV Network. *IEEE Trans. Wirel. Commun.* **2023** . [CrossRef]
28. Krayani, A.; Alam, A.S.; Marcenaro, L.; Nallanathan, A.; Regazzoni, C. An Emergent Self-Awareness Module for Physical Layer Security in Cognitive UAV Radios. *IEEE Trans. Cogn. Commun. Netw.* **2022**, *8*, 888–906. [CrossRef]
29. Krayani, A.; William, N.J.; Alam, A.S.; Marcenaro, L.; Qin, Z.; Nallanathan, A.; Regazzoni, C. Generalized Filtering with Transport Planning for Joint Modulation Conversion and Classification in AI-enabled Radios. In Proceedings of the ICC 2022 - IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 3759–3765. [CrossRef]
30. Li, X.; Wang, Q.; Liu, J.; Zhang, W. Trajectory Design and Generalization for UAV Enabled Networks:A Deep Reinforcement Learning Approach. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC), Seoul, Republic of Korea, 25–28 May 2020; pp. 1–6. [CrossRef]
31. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Yu, P.S. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Trans. Neural Networks Learn. Syst.* **2022**, *33*, 494–514. [CrossRef]
32. Griffiths, T.L.; Chater, N.; Kemp, C.; Perfors, A.; Tenenbaum, J.B. Probabilistic models of cognition: Exploring representations and inductive biases. *Trends Cogn. Sci.* **2010**, *14*, 357–364. [CrossRef]
33. Parr, T.; Friston, K.J. Uncertainty, epistemics and active inference. *J. R. Soc. Interface* **2017**, *14*, 20170376. [CrossRef]
34. Friston, K.; FitzGerald, T.; Rigoli, F.; Schwartenbeck, P.; Pezzulo, G. Active Inference: A Process Theory. *Neural Comput.* **2017**, *29*, 1–49. [CrossRef]
35. Friston, K. Active inference and free energy. *Behav. Brain Sci.* **2013**, *36*, 212–213. [CrossRef]
36. Parr, T.; Friston, K.; Pezzulo, G. Generative models for sequential dynamics in active inference. *Cogn. Neurodynamics* **2023**, 1–14. [CrossRef]
37. Friston, K.J.; Parr, T.; de Vries, B. The graphical brain: Belief propagation and active inference. *Netw. Neurosci.* **2017**, *1*, 381–414. [CrossRef]
38. Matsuo, Y.; LeCun, Y.; Sahani, M.; Precup, D.; Silver, D.; Sugiyama, M.; Uchibe, E.; Morimoto, J. Deep learning, reinforcement learning, and world models. *Neural Networks* **2022**, *152*, 267–275. [CrossRef]
39. Feillet, D.; Dejax, P.; Gendreau, M. Traveling Salesman Problems with Profits. *Transp. Sci.* **2005**, *39*, 188–205. [CrossRef]
40. Krayani, A.; Alam, A.S.; Calipari, M.; Marcenaro, L.; Nallanathan, A.; Regazzoni, C. Automatic Modulation Classification in Cognitive-IoT Radios using Generalized Dynamic Bayesian Networks. In Proceedings of the 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, 14 June–31 July 2021; pp. 235–240. [CrossRef]
41. Krayani, A.; Baydoun, M.; Marcenaro, L.; Alam, A.S.; Regazzoni, C. Self-Learning Bayesian Generative Models for Jammer Detection in Cognitive-UAV-Radios. In Proceedings of the GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020; pp. 1–7. [CrossRef]
42. Baydoun, M.; Campo, D.; Sanguineti, V.; Marcenaro, L.; Cavallaro, A.; Regazzoni, C. Learning Switching Models for Abnormality Detection for Autonomous Driving. In Proceedings of the 2018 21st International Conference on Information Fusion (FUSION), Cambridge, UK, 10–13 July 2018; pp. 2606–2613. [CrossRef]
43. Tran, D.H.; Vu, T.X.; Chatzinotas, S.; ShahbazPanahi, S.; Ottersten, B. Coarse Trajectory Design for Energy Minimization in UAV-Enabled. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9483–9496. [CrossRef]
44. Zixuan, Z.; Qinhao, W.; Bo, Z.; Xiaodong, Y.; Yuhua, T. UAV flight strategy algorithm based on dynamic programming. *J. Syst. Eng. Electron.* **2018**, *29*, 1293–1299. [CrossRef]

45. De Waen, J.; Dinh, H.T.; Cruz Torres, M.H.; Holvoet, T. Scalable multirotor UAV trajectory planning using mixed integer linear programming. In Proceedings of the 2017 European Conference on Mobile Robots (ECMR), Paris, France, 6–8 September 2017; pp. 1–6. [CrossRef]
46. Dhulkefl, E.; Durdu, A.; Terzioğlu, H. Dijkstra algorithm using UAV path planning. *Konya Mühendislik Bilim. Derg.* **2020**, *8*, 92–105. [CrossRef]
47. Karur, K.; Sharma, N.; Dharmatti, C.; Siegel, J.E. A survey of path planning algorithms for mobile robots. *Vehicles* **2021**, *3*, 448–468. [CrossRef]
48. Ibrahim, N.S.A.; Saparudin, F.A. Review on path planning algorithm for unmanned aerial vehicles. *Indones. J. Electr. Eng. Comput. Sci.* **2021**, *24* . [CrossRef]
49. Xie, J.; Garcia Carrillo, L.R.; Jin, L. Path Planning for UAV to Cover Multiple Separated Convex Polygonal Regions. *IEEE Access* **2020**, *8*, 51770–51785. [CrossRef]
50. Johnson, D.S.; McGeoch, L.A. The traveling salesman problem: A case study. In *Local Search in Combinatorial Optimization*; Aarts, E., Lenstra, J.K., Eds.; Princeton University Press: Princeton, NJ, USA, 2003; pp. 215–310. [CrossRef]
51. Chen, J.; Ye, F.; Li, Y. Travelling salesman problem for UAV path planning with two parallel optimization algorithms. In Proceedings of the 2017 Progress in Electromagnetics Research Symposium-Fall (PIERS-FALL), Singapore, 19–22 November 2017; pp. 832–837. [CrossRef]
52. Pehlivanoglu, Y.V.; Pehlivanoglu, P. An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems. *Appl. Soft Comput.* **2021**, *112*, 107796. [CrossRef]
53. Phung, M.D.; Ha, Q.P. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization. *Appl. Soft Comput.* **2021**, *107*, 107376. [CrossRef]
54. Yue, L.; Chen, H. Unmanned vehicle path planning using a novel ant colony algorithm. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 136. [CrossRef]
55. Bayerlein, H.; De Kerret, P.; Gesbert, D. Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning. In Proceedings of the 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Kalamata, Greece, 25–28 June 2018; pp. 1–5. [CrossRef]
56. Colonnese, S.; Cuomo, F.; Pagliari, G.; Chiaraviglio, L. Q-SQUARE: A Q-learning approach to provide a QoE aware UAV flight path in cellular networks. *Ad Hoc Networks* **2019**, *91*, 101872. [CrossRef]
57. Abeywickrama, H.V.; He, Y.; Dutkiewicz, E.; Jayawickrama, B.A.; Mueck, M. A Reinforcement Learning Approach for Fair User Coverage Using UAV Mounted Base Stations Under Energy Constraints. *IEEE Open J. Veh. Technol.* **2020**, *1*, 67–81. [CrossRef]
58. Zhang, Q.; Saad, W.; Bennis, M.; Lu, X.; Debbah, M.; Zuo, W. Predictive Deployment of UAV Base Stations in Wireless Networks: Machine Learning Meets Contract Theory. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 637–652. [CrossRef]
59. Liu, X.; Liu, Y.; Chen, Y.; Hanzo, L. Trajectory Design and Power Control for Multi-UAV Assisted Wireless Networks: A Machine Learning Approach. *IEEE Trans. Veh. Technol.* **2019**, *68*, 7957–7969. [CrossRef]
60. Hu, Y.; Chen, M.; Saad, W.; Poor, H.V.; Cui, S. Meta-Reinforcement Learning for Trajectory Design in Wireless UAV Networks. In Proceedings of the GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020; pp. 1–6. [CrossRef]
61. Yin, S.; Zhao, S.; Zhao, Y.; Yu, F.R. Intelligent Trajectory Design in UAV-Aided Communications With Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2019**, *68*, 8227–8231. [CrossRef]
62. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Wireless Communication Using Unmanned Aerial Vehicles (UAVs): Optimal Transport Theory for Hover Time Optimization. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 8052–8066. [CrossRef]
63. Applegate, D.L.; Bixby, R.E.; Chvatál, V.; Cook, W.J. *The Traveling Salesman Problem: A Computational Study*; Princeton University Press: Princeton, NJ, USA, 2006.
64. Krayani, A.; Alam, A.S.; Marcenaro, L.; Nallanathan, A.; Regazzoni, C. Automatic Jamming Signal Classification in Cognitive UAV Radios. *IEEE Trans. Veh. Technol.* **2022**, *71*, 12972–12988. [CrossRef]
65. Zeng, Y.; Zhang, R. Energy-Efficient UAV Communication with Trajectory Optimization. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 3747–3760. [CrossRef]
66. Watkins, C.; Dayan, P. Technical Note: Q-Learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.