Hindawi
International Journal of Aerospace Engineering
Volume 2020, Article ID 2027340, 20 pages
https://doi.org/10.1155/2020/2027340

Research Article

Self-Navigating UAVs for Supervising Moving Objects over Large-Scale Wireless Sensor Networks

Tien Pham Van,[1] **Nguyen Pham Van**,[1] **and Trung Ha Duyen**[2]

[1]*Department of Communications Engineering, Hanoi University of Science and Technology, Hanoi 844, Vietnam*
[2]*Department of Aerospace Electronics, Hanoi University of Science and Technology, Hanoi 844, Vietnam*

Correspondence should be addressed to Tien Pham Van; tien.phamvan1@hust.edu.vn

Received 13 December 2019; Revised 13 February 2020; Accepted 22 February 2020; Published 16 June 2020

Guest Editor: Zongyu Zuo

Increasingly inexpensive unmanned aerial vehicles (UAVs) are helpful for searching and tracking moving objects in ground events. Previous works either have assumed that data about the targets are sufficiently available, or they solely rely on on-board electronics (e.g., camera and radar) to chase them. In a searching mission, path planning is essentially preprogrammed before taking off. Meanwhile, a large-scale wireless sensor network (WSN) is a promising means for monitoring events continuously over immense areas. Due to disadvantageous networking conditions, it is nevertheless hard to maintain a centralized database with sufficient data to instantly estimate target positions. In this paper, we therefore propose an online self-navigation strategy for a UAV-WSN integrated system to supervise moving objects. A UAV on duty exploits data collected on the move from ground sensors together with its own sensing information. The UAV autonomously executes edge processing on the available data to find the best direction toward a target. The designed system eliminates the need of any centralized database (fed continuously by ground sensors) in making navigation decisions. We employ a local bivariate regression to formulate acquired sensor data, which lets the UAV optimally adjust its flying direction, synchronously to reported data and object motion. In addition, we also construct a comprehensive searching and tracking framework in which the UAV flexibly sets its operation mode. As a result, least communication and computation overhead is actually induced. Numerical results obtained from NS-3 and Matlab cosimulations have shown that the designed framework is clearly promising in terms of accuracy and overhead costs.

1. Introduction

Recently, combination of unmanned aerial vehicles (UAVs) and a wireless sensor network (WSN) has been attracting much attention from the research community. Increasingly inexpensive UAVs, thanks to their mobility and flexibility, may efficiently help maintain the network connectivity and relay data [1, 2]. While WSN is an appropriate solution to continuously gather data, its network nodes are not easily relocated and their wireless communication links are essentially unreliable [1–3]. This hinders capturing a full view over a large area. It challenges against instant operations in response to sporadic events as well. If these events are associated with moving objects, the problem is even more complicated [4, 5]. On the other hand, increasingly popular UAVs may effectively compensate the gap, thanks to its high dynamism and autonomy. If being navigated smartly, they may quickly approach event places, monitoring their evolution and even getting involved in handling operations [6–8]. However, over immense spaces such as fields, parks, forests, national borders, seas, and deserts, UAVs alone cannot search elaborately and extensively due to their limited flight time and distance. Their embedded sensing devices can merely "see" the ground underneath at a certain range. This desires data collection from ground sensors to enhance the searching capability of UAVs. As such, mutually compensated power of UAV and WSN means makes the combination suitable for many supervising applications, such as monitoring intruders, animals, fire, and vehicles.

Practically, connectivity in WSN is quite hard to be kept continuous, obstructing data concentration on gateways placed at long distances from sensors. This definitely hampers construction of an adequate measurement database for calculating the exact object location [9, 10]. However, known

searching and tracking strategies in literature heavily rely on such a centralized database. Locating these objects desires heavy data exchange and storage of history statistics [4, 5, 11]. Sensors are assumed to be densely populated and robust to provide sufficient statistics continuously, which is unrealistic in large areas. Additionally, it is costly in terms of energy and communication to collect data from distant sensors of many hops away [12, 13].

Technically, an event occurrence is expressed by dramatic changes of one or a few parameters, such as gas volume, sound wave, light, and radiation in the affected region [8, 11, 14, 15]. Unfortunately, data about the event may not be fully reported by sensors at once. Consequently, the system cannot immediately locate and track objects. Sometimes, necessary statistics can only be collected to accurately pinpoint a target object after the UAV flies through a relatively large suspected area [9–12]. This means that preprogramming the UAV trajectory before taking off cannot provide the optimal flight path.

By analyzing radio signals intensively, several works have introduced radio-based localization algorithms in literature [16–18]. Namely, radio signals are processed to predict the current target position. The authors of [16] introduce a trilateration algorithm in combination with the LMS (least mean square) method. The strategy proposes to insightfully learn the channel model to draw localization inference about the target position. Similarly, another RSSI- (received signal strength indicator-) based node localization scheme is proposed in [17]. The distributed weighted search localization algorithm (WSLA) was constructed, combining a node localization precision classification scheme and weight-based searches. The strategy brings up considerable improvement with respect to accuracy. Meanwhile, the authors of [18] constructed a RSSI-assisted localization scheme using machine learning and Kalman filter. They designed two learning algorithms: ridge regression (RR) and vector-output regularized least squares (vo-RLS), which effectively help reduce localization error. While the radio-analyzed approach is conceptually interesting and incurring little latency, its accuracy apparently depends on existence of obstacles. Furthermore, calibrating and training the model to map RSSI to the target position are potentially expensive. The method is therefore likely unfeasible when the mission takes place in a spacious area. In addition, random changes in the environment potentially affect the accuracy as well.

In a similar approach, the authors of [4, 19] introduced tracking methods based on capturing images of target objects and/or on recording their sound. Upon intensive image and acoustic data analysis, the system infers instant positions of a moving object that falls in the visible/audible range of sensors. Arranging directional imaging sensors in a node-dense grid, the authors of [4] designed a k-target tracking and m-connected WSN. The system works well on indoor places and on the road, successfully tracking multiple objects. The solution is however expensive and highly object-selective. On the other hand, both acoustic and image data were made use of in [19] to locate target objects. The authors employed the Gauss-Markov mobility model to predict the target trajectory, which helps restrict the suspected region to locate and track. The scheme is therefore highly energy-efficient. However, this approach is in general dependent on central gateway(s) to execute heavy computing tasks. Multimedia communication over resource-constrained WSN is apparently costly.

With respect to usage of on-board sensing electronics, in [20, 21], the authors proposed image-based navigation algorithms. This strategy obviously eases the work of users/-operators, given that UAVs provide visual data of targets. Advanced image processing techniques also enable quick object detection and bring up a high autonomy in surveying events. A template matching and morphological filtering combined algorithm is introduced in [20], which allows a UAV to track another cooperative vehicle within the distance of few meters. The tracking machine uses a monocular camera and exchanges data with the target to assure the mission success. Differently, multiple means (monocular camera, GPS receiver, and IMU sensor) were exploited in [21] for a UAV to supervise moving vehicles in multiple levels. The proposed scheme first visually detects a vehicle, then tracks its movement, and eventually processes data to navigate the flying UAV from the ground. However, in all of these strategies, on-board cameras can only scan a limited footprint on the ground. This approach is therefore suitable for tracking vehicles that are moving on the road only. It cannot deal with free movement of objects elsewhere. In this regard, our work opts for ground sensors as means of providing extra data. This allows the system to cover a larger UAV-traceable area. To extend the monitoring range, the authors of [15] let networked UAVs work in cooperation with WSNs to detect and monitor natural disaster events based on images and environment statistics (temperature, smoke, humidity, and light). A multilevel data analysis, advanced image processing in particular, and cloud- (ThingSpeak) assisted ground monitoring/maneuvering subsystem were designed to detect and locate such events as fire in the early stage. While the system is practical and applicable with respect to natural disasters, it is not flexible to tightly track particular moving objects. Moreover, UAVs do not seem to be highly autonomous.

From the data processing and representation perspective, ontology and domain knowledge modeling have been introduced to interpret event occurrences [22, 23]. The semantic presentation technology concentrates on intensively analyzing UAV-acquired data (videos, images) to visually understand the scene. By application of AI techniques for advanced image and video processing, it is able to detect, identify, and label objects of interest, which facilitates the context interpretation [22]. Ontology-driven representation using semantic statements such as those in the TrackPOI schema [23, 24] generates high-level descriptions of the scene, which supports UAVs and human operators to visually detect, differentiate and track moving objects. Note however that this approach does not target at searching any object or navigating UAVs in real time to acquire those data. The data processing operations can only be performed with the assumption that the flying UAVs have already visually found out objects and are now successfully tracking them. Furthermore, ontology-driven computation requires a high concentration of data collected and obviously induces long

delay due to heavy overhead generated by AI algorithms and semantic web. In the targeted context our study aims at, objects may move arbitrarily (unlike vehicles on an established road) at a high velocity, which demands instant response of inherently resource-limited UAVs to continuously keep track of. This consequently discourages the application of any heavy-overhead protocol. In reality, objects may require sensed data other than visual media, such as sound and radiation, to be detected and tracked. Thus, the ontology approach cannot be used as a key means to search moving objects and to self-navigate UAVs in tracking them in real-time. In principle, the technology may be employed in conjunction with our proposed self-navigation strategy to facilitate ground operations, but this is out of our study scope.

Remarkably, the authors of [8] introduce an interesting and promising framework in which UAVs are equipped with gas sensors to collect data. The study also devised a self-adaptive team of UAVs for monitoring gas emission events. However, in this approach, if the gas detection data collected by UAVs are not sufficient to fully cover the event area, the positioning and tracking mission likely fails. Preprogramming paths before departure, as mentioned earlier, potentially navigate the UAVs along nonoptimal trajectories. This means that the model works efficiently against only such events as gas emission, where no highly dynamic object is targeted. Relying solely on on-board sensors also excludes capability of monitoring ground objects that do not alter measurement results at the flying altitude. Similarly, the progressively spiral-out scheme [25] presents an exhaustive scanning method to discover a mobile target. A time-varying team of UAVs are coordinated to fly along spiral-out trajectories with the hope that the spiral-bounded area covers its location. Depending on how the estimated confidence area changes over time, some UAVs may leave the team or extra ones are invited to join in. While this scanning algorithm is truly aimed at searching mobile targets, its exhaustiveness costs more flying distance and UAVs than needed. Furthermore, as the UAVs fly along planned paths, regardless of how the target object moves, they do not continuously keep track of it.

In this work, we therefore construct an online (dynamic) self-navigating strategy where a UAV makes use of measurement data reported from ground sensors to navigate itself. Namely, the UAV is steered by referring to instant sensed data, rather than being path-planned before taking off. Its on-board camera(s) or other sensing means is also employed for confirming the presence of the target object and tracking its movement. The problems we attempt to address are (i) how the UAV should be self-navigated to find out and catch up with the moving object in different levels of data availability and (ii) how the available sensed data are processed to create the best navigation hints. Our technical contributions reported in this paper accordingly include the following:

(1) We build a system model in which UAVs cowork with sensors to locate and track events associated with moving objects. The UAVs collect data from preexisting ground sensors as they are flying. In particular, the UAVs may also proactively deploy sensors on the fly to supplement sensed data about the moving object if necessary. The UAVs are directed based on processing data acquired on the move and on captured images to detect the object

(2) We propose an online self-navigation strategy based on the above data sources. The dynamic steering scheme is carried out without referring to any centralized database. A local bivariate regression is constructed to formulate the scalar field from measurement data, which supports calculating/predicting the best flying direction regularly. The work presents an efficient transformation of the bivariate function. The local regression can accordingly be implemented by applying known algorithms for univariate polynomials

(3) Dealing with object motion, we introduce a "wait-in-front" tactic for a UAV to catch up with the moving object quicker. Namely, by observing its motion, the UAV heads toward the next predicted position of the object, instead of its current one. This means that the flying direction is deflected from the gradient vector by a calculated angle

(4) We present a maneuvering framework in which the UAV flexibly changes its path adjustment policy depending on gathered measurement data. Specifically, UAV steering may switch from gradient vector-driven to heading straightforward, as well as from searching to tracking or the other way around. The goal is to save communication load and edge processing cost aboard. Besides, the flying UAV autonomously restores searching operations when the object has moved away from its tracking range

The rest of this paper is organized as follows. Section 2 describes the system model, highlighting the integrated UAV-WSN structure and UAV self-navigation strategy. It also explains how an object influences the measured parameter, followed by a comprehensive protocol for searching and tracking missions. Section 3 details a local bivariate regression formulation for calculating the gradient vector at any point. Then, algorithms for dealing with moving objects are presented in Section 4. A detailed description of how UAVs are self-navigated in an online manner to catch up with them is also presented. Section 5 subsequently analyzes overhead of our self-navigation strategy in terms of complexity and communication load. To demonstrate the soundness of the proposed framework, we report NS-3 and Matlab-based cosimulation results in Section 6. Statistical performance of the self-navigation in terms of accuracy and costs is concretely discussed therein. Finally, Section 7 draws our conclusions.

2. System Model

Being composed of UAVs and wireless sensors, the proposed system is visually indicated in Figure 1. Measurement data are collected from ground sensors, both directly and via

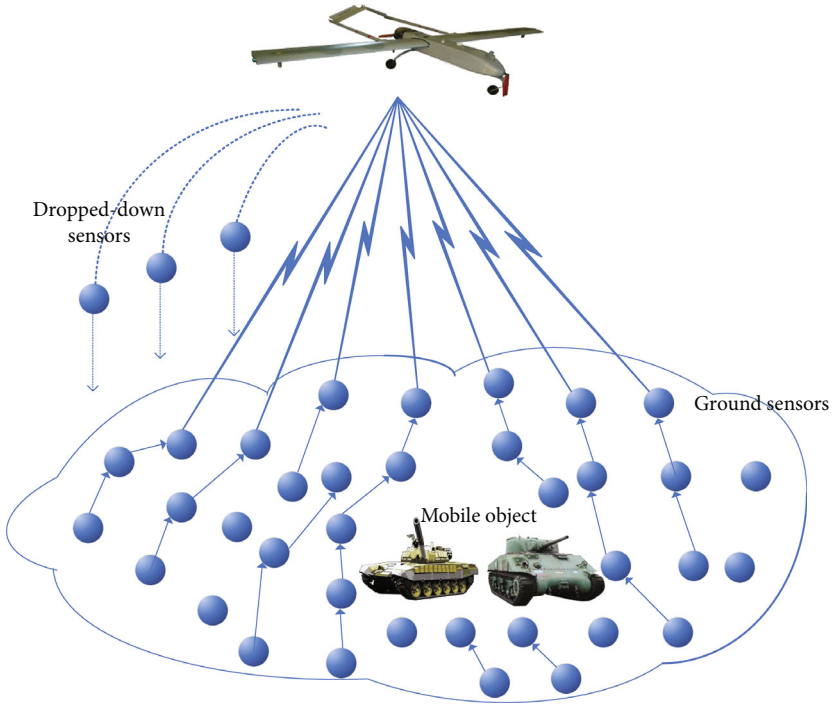


Figure 1: System layout—a UAV in searching and tracking an object over a large area where ground wireless sensors have been deployed. The UAV collects data from sensors underneath on the move and may drop down extra ones.

multiple hop paths. Occasionally, a UAV may drop down sensors where and when needed to get more data for locating an object. In addition, the following conditions hold:

(i) The UAV knows its current position and that of sensors from which it has just received measurement data

(ii) The terrain under surveillance is relatively flat, so that the problem of searching and tracking is presented in 2D space

(iii) The UAV at standby state is signaled to fly searching once an object presence is suspected but does not have sufficient data at once to know its exact location

Once a detectable object appears and moves around in the area covered by the WSN, we say that an event occurs. The presence of the object (e.g., vehicle, animal, and radiation source) will cause abnormal changes in some measurement data within its surrounding region. A UAV on its surveillance mission needs to sense these changes. Without referring to any centralized database of sufficient data, the UAV has to rely on data sent from ground sensors to be self-navigated. Due to imperfection of the ground sensor network, it should collect data on the move to gradually know better about the event. Understanding the rule of changes associated with the object appearance, the UAV may appropriately shape its trajectory in a dynamic manner.

2.1. Object and Event Description. Conceptually, an event is the circumstance in which one or a few remarkable objects appear in the area covered by the large-scale WSN. Its occurrence triggers the system to detect, locate, and track moving objects. The presence of a target object results in the fact that some parameter (either primitive or compound), whose value is position-dependent, excessively increases or decreases toward extreme values [8]. Let us denote the parameter, which will be measured by ground sensors, as S. It is assumed that the impact of the object presence on the parameter values at surrounding positions is transient; i.e., it quickly disappears when the object has moved away. In reality, such objects may be vehicles, machines, animals, intruders, radiation sources, etc. that emit light, radio, radiation in general, sound, or the like. Apparently, sensed data will reflect instant intensity of these signals. Without loss of generality, we assume that the parameter values get higher in the object-affected area [8, 11, 26]. Figure 2 shows a typical plot of S values over the object-affected ground space at some time instant, which forms a cone Ω. There exist three sections separated by two thresholds $S_{\text{thr}}^{\text{low}}$ and $S_{\text{thr}}^{\text{high}}$: definitely "no" ($S < S_{\text{thr}}^{\text{low}}$), likely ($S_{\text{thr}}^{\text{low}} \le S < S_{\text{thr}}^{\text{high}}$), definitely "yes" ($S \ge S_{\text{thr}}^{\text{high}}$), which literally indicate the possibility of object existence thereon.

The projection of the cone top onto the ground plane should be the center of the object if it is present. As soon as the searching UAV acquires sufficient sensor data, it gets to know the position. The UAV is then navigated directly toward the object. In reality, this good luck however does not happen right at the beginning of the search due to the facts that

(i) ground sensors do not always successfully transmit their data to a distant centralized point

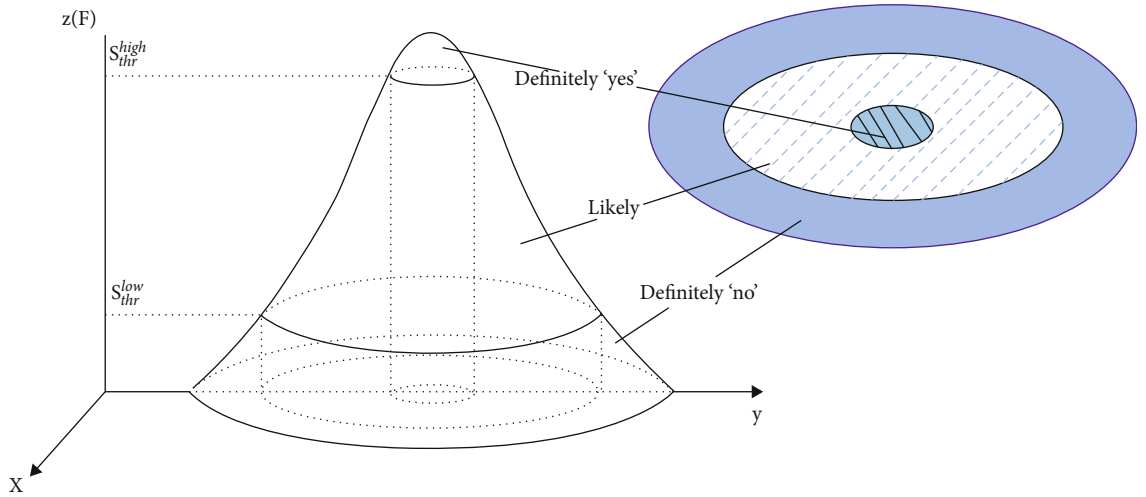


Figure 2: A typical plot representing values of the position-dependent parameter S in the vicinity of the target object—a "cone" shape reflects the impact of the object on the parameter that will be measured by ground sensors. The three sections (definitely "yes," likely, and definitely "yes") literally express how possibly the object is really located on.

(ii) the object-affected area is large, taking time for the UAV to capture data around cone peaks

(iii) occurrence of the event itself may, to some extent, adversely affect the sensor network connectivity, which hinders centralizing data

As a result, the UAV only obtains data from its nearby sensors, which likely excludes ones close to the object in early time of the search. A question raised here is, given that limited information, how the UAV should be directed to move as close as possible to the object. Its answer would be that the UAV just flies along the direction that observes the highest spike of measurement values.

If multiple events occur in parallel, separate objects should be observed. In such a case, it simply dispatches one UAV to search and track each. To identify and label each object, UAVs essentially rely on image-based recognition or on special on-board electronics that process, for example, ultrasound or radiation signals. Another alternative is to track the "cone" Ω associated with each object. In principle, separate objects should create different shapes of Ω plot. Furthermore, by tracking each moving object tightly (by continual data updates from ground sensors), the system may also predict its position better and hence alleviate the confusion probability.

2.2. Online Self-Navigation Strategy. While the UAV is flying, data reported from connected sensors let it model a scalar field Φ represented by parameter S. Formulation of acquired measurement data can be made at least for the space nearby: $\Phi = F(x, y)$, where (x, y) is the current UAV position projected (orthogonally) onto the ground (hereafter loosely called UAV position/location). The object place practically observes a sudden change of gradient [8, 26], or a climax of the measured parameter. Obviously, when the UAV always flies along the gradient vector, it definitely reaches the climax in the end.

Knowing the explicit expression of function F, the UAV can determine the gradient vector of scalar field Φ at its current position (x, y):

$$\nabla F = \frac{\partial F}{\partial x} \times \vec{i} + \frac{\partial F}{\partial y} \times \vec{j} = \left[\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}\right]^T. \quad (1)$$

Once the gradient vector has been computed, the UAV is dynamically self-navigated according to the velocity vector. It can be expressed as

$$\begin{cases} \overrightarrow{V_u}(x, y) = \kappa \times \nabla F, \\ |\phi| \le \phi^{\max}, \end{cases} \quad (2)$$

where κ is the proportional factor that depends on the instant ground speed of the UAV and ϕ and $\phi^{\max}$ are the turning angle and its maximally allowed values, respectively.

However, the UAV gets sensed information from ground sensors that are not too far way; the expression of F is only true at its surrounding space. When the object is still distant, the expression-valid domain does not cover the object position. This means that, with a limited data from ground sensors nearby, the UAV knows only the closer part of the cone while moving. Repetition of F formulation after each certain moving length is consequently required. Namely, it takes time for the UAV to locate the peak position (x_c, y_c) of the cone Ω. Upon getting to know the peak, the UAV is signaled *peak found* as seen in Figure 3. It then simply heads, as fast as possible, along the straight line thereto (state *straight line*). There are two possible cases when the UAV reaches the peak:

(i) The object is successfully *found* thereat. Confirmation of the object presence may be realized based on on-board sensing devices. For a simplified presentation, we hereafter assume that images captured by

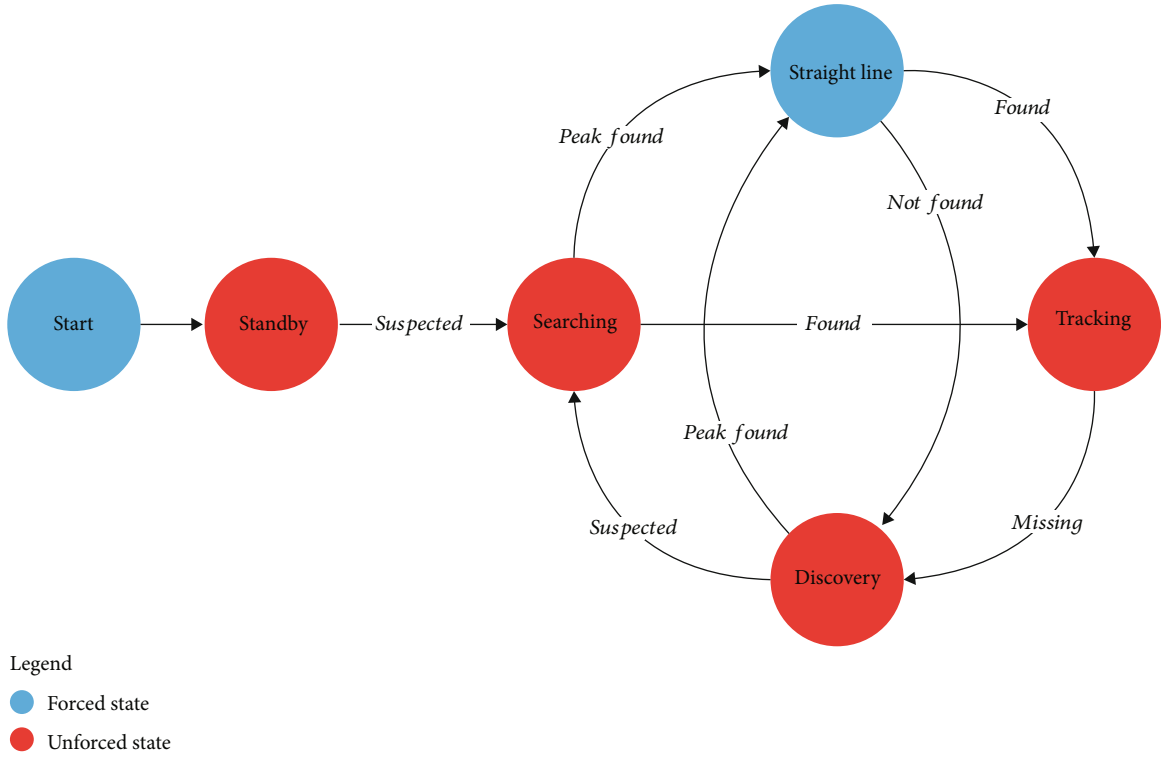


Figure 3: Finite state machine of the searching and tracking process. Calculation of gradient vector is needed only in early time at a *searching* state.

on-board cameras are processed for this purpose. Signal *found* may also be generated if some measured values exceeding $S_{\text{thr}}^{\text{high}}$ are received. The UAV now terminates its *searching* phase and switches to *tracking* state as seen in the figure. Its self-navigation behaviors in the tracking mode will be later described in Section 4. Briefly speaking, the process mainly relies on the on-board camera(s), instead of ground sensors

(ii) The UAV does not detect any object; it is just a local peak (x_c, y_c). Another possibility is that the object has already run too far away. In either situation, the UAV is signaled *not found* and moves to work in state *discovery*. As getting extra ground data, the UAV may be sometime informed of abnormally increasing values (greater than $S_{\text{thr}}^{\text{low}}$) at a place. It then returns to the searching state on signal *suspected*. On the other hand, if the UAV luckily finds out another peak thanks to further ground data, it simply reiterates another *straight line* journey

During the tracking stage, if the object moves out of the localizable range, signal missing is generated, bringing the UAV to the *discovery* state as well. As such, gradient vector is not estimated all the time of searching and tracking. Its calculation is necessary only when ground sensor data are insufficient. Even in the searching phase, the UAV will cease the periodical gradient update when a peak is located. These behaviors are later embodied in an algorithm devised in Section 4.2. It is also noticed that, the UAV normally stays at the *standby* state, being ready for taking off. When a value of $S > S_{\text{thr}}^{\text{low}}$ is received, it instantly departs for the source position. Besides, during its mission, the UAV should automatically land on if the tracking mission ends, or it is about to run out of energy or fuel.

To assure that the UAV is self-navigated heading to expected positions, even in case of wind and atmospheric disturbances, application of a supplementary adjustment model [27, 28] can be incorporated. With respect to positioning, if the UAV relies on satellite-based positioning systems (e.g., GPS), the impact of bad weather (e.g., fog, smog, and rain) on the accuracy is said to be insignificant [29]. Furthermore, it may also improve the accuracy by correlating its own data to the sensor-reported samples. In confirming the object appearance, severe reduction of visibility due to bad weather may degrade camera images. To overcome this challenge, the UAV may be equipped with supplementary on-board sensors that measure ultrasound, radiation, or the like, to detect the target object better.

3. Data Formulation and Processing

In the searching phase, the UAV is flying in the object-affected region where values of S are greater than $S_{\text{thr}}^{\text{low}}$. To update the gradient vector periodically, it keeps collecting data from ground sensors. Based on the data, function F is formulated, whose expression is valid only for surrounding positions. This section presents an appropriate

Table 1: Definition of key symbols.

Symbol	Meaning/explanation
Φ	The scalar field of parameter S whose value changes upon object arrival
Ω	The object-affected region, observing abnormal growth in values of parameter S
$S_{\text{thr}}^{\text{high}}, S_{\text{thr}}^{\text{low}}$	The low and high levels of parameter S that define probabilistic sections of object presence
F_n	The n-degree bivariate polynomial to be regressed for estimating the gradient vector
N	The number of terms in F_n, being equal to $(n+1)(n+2)/2$
m	The number of measurement samples available right before regression of F_n
$\overrightarrow{g_i}$	Gradient vector at the current position of the field
$V_{\text{u}}, V_{\text{obj}}$	Ground velocity of the UAV and the object, respectively
$\mathbf{a}$	The vector whose coefficients a_k (calculated by local regression) define F_n
P	The scalar whose elements p_k are sorted in increasing order of monomials $m_k = a_k p_k = a_{i,j} x^{i-j} y^j$. Lemma 1 later shows the relation between k and i, j
$\mathbf{P}$	The $m \times N$ matrix formed by arranging scalar P into its rows
$\mathbf{W}$	The $m \times m$ square matrix whose coefficients are all equal to $w(x, y)$. This factor weighs the influence of sensor measurements on $\mathbf{a}$
$\mathbf{F}$	The vector whose coordinates are m samples of F_n reported by ground sensors (used for regression)
U_i, P_i	The orthogonal projection position of UAV onto the ground, object position at the time interval i
$T_{\text{up}}, T_{\text{cap}}$	The time intervals, respectively, for updating the gradient vector and for capturing images to confirm the object presence
α_i	The deviation angle to steer the UAV against the gradient vector at the regression/update interval i
$R_{\text{scan}}, R_{\text{sens}}$	The radius of visible range by on-board camera and object localizable range by ground sensor data, respectively. Lemma 2 later mentions sufficient conditions on them to keep track of the mobile object
$L_{\text{msg}}, L_{\text{reg}}, R_{\text{data}}$	The length of sensor messages, data load per regression, and average uplink throughput from ground sensors to the UAV, respectively
$D_{\text{max}}, D_{\text{avg}}$	The tracking deviation (being, respectively, max and average)—distance between the projection of the UAV onto the ground and the object center
$\mathscr{P}, \mathscr{P}_0$	The energy consumption power values on moving and on hovering, respectively

nonparametric regression scheme for the purpose. For the ease of reading, Table 1 explains the meaning of all important symbols.

3.1. Formulation Expression of Ground Data. As stated in the previous section, F is a bivariate function of coordinates x and y on the 2D ground plane. There are theoretically two ways to formulate F at a certain point: interpolation and regression. We choose local regression for the following reasons:

(i) The number of measurement samples may be large, making polynomial interpolation intractable. New measurement samples may come in at a high rate, and their values are time-varying

(ii) In reality, the impact of an event on the measured parameter is maximum at its center and gradually decreases along the distance therefrom. This implicates a necessity of weighing measurement values, taking into account their source position.

Given that at present time t, parameter S, whose value is a function of coordinates x and y, is expanded as a n-degree polynomial bivariate function $F_n(x, y)$. It can be expressed as

$$\begin{aligned} F_n(x,y) &= \varepsilon + a_{0,0} + a_{1,0}x + a_{1,1}y + a_{2,0}x^2 + a_{2,1}xy + a_{2,2}y^2 \\ &\quad + a_{3,0}x^3 + a_{3,1}x^2y + a_{3,2}xy^2 + a_{3,3}y^3 + \cdots + a_{n,0}x^n \\ &\quad + a_{n,1}x^{n-1}y + a_{n,2}x^{n-2}y^2 + \cdots + a_{n,n}y^n \\ &= \varepsilon + \sum_{i=0}^{n} \sum_{j=0}^{i} a_{i,j} x^{i-j} y^j, \end{aligned} \tag{3}$$

where ε represents the noise influence on measurements. In words, $F_n(x, y)$ is the sum of monomials arranged in an increasing total order of the two variables. For the ease of local regression at edge processing, we propose to transform F_n in Equation (3) into 1D indexing for a.

Lemma 1. *There exists a bijective map between two-dimension array $a_{i,j}$ and one-dimension one a_k, whose index also locates corresponding terms of F_n.*

Proof. Let us define scalar $P = [1, x, y, x^2, xy, y^2, \cdots, y^n]$, and vector $\mathbf{a} = [a_{0,0}, a_{1,0}, a_{1,1}, a_{2,0}, a_{2,1}, a_{2,2}, \cdots, a_{n,n}]^T$. Coefficients $a_{i,j} \mid i = \overline{0, n}, j = \overline{0, i}$ characterize polynomial $F_n(x, y)$, being calculated by local regression. Obviously, the size of P and $\mathbf{a}$

is $N = (n+1)(n+2)/2$. We now number elements of them by a single index $k = 0, ((n+1)\bar{(}n+2)/21)$. Looking at expression of F_n in Equation (3), one can realize that

$$k = \frac{i(i+1)}{2} + j. \tag{4}$$

Then, P and $\mathbf{a}$ are rewritten as $[p_0, p_1, p_2, \cdots, p_k \cdots, p_{N-1}]$ and $[a_0, a_1, a_2, \cdots, a_k \cdots, a_{N-1}]^T$, respectively. Note that $j \le i$ and that i, j, and k must be nonnegative integers. Given a value of k, the corresponding value of i is the maximum integer that satisfies Equation (4). We can accordingly locate the kth term of $F_n(x, y)$ (not counting the noise)—monomial $m_k = a_k \times p_k = a_{i,j} x^{i-j} y^j$—where i and j indexes are calculated as

$$\begin{aligned} &(4) \Rightarrow i^2 + 3i - 2k \ge 0, \\ &\begin{cases} i = \left\lceil \dfrac{\sqrt{9+8k} - 3}{2} \right\rceil, \\ j = k - \dfrac{1}{2} \left\lceil \dfrac{\sqrt{9+8k} - 3}{2} \right\rceil \left\lceil \dfrac{\sqrt{9+8k} - 1}{2} \right\rceil. \end{cases} \end{aligned} \tag{5}$$

Once values of i and j are found out, the kth term of P, i.e., $x^{i-j} y^j$, is determined. This completes the proof.

It is then inferred from Lemma 1 that $F_n(x, y) = P\,\mathbf{a} + \varepsilon$. Sensor data may practically incur noise. We do not intensively address the issue in this work, but assume that the noise is position-independent. At the same time, Kalman filter also helps alleviate the white-noise impact on measurement data [18]. Furthermore, the information accuracy, in case of bad weather (e.g., fog, smog, and rain), can also be improved by raising the transmission redundancy. Dropping down extra sensors from the UAV and increasing the frequency of data reporting, among other alternatives, may be reinforced for this purpose. The noise accordingly does not influence the calculation of gradient, for which ε can computationally be considered part of $a_{0,0}$ in subsequent formulation steps. At each point $\mathrm{P}(x, y)$, coordinates of the gradient vector are consequently set as

$$\begin{cases} \dfrac{\partial F_n}{\partial x} = \dfrac{\partial P}{\partial x}\,\mathbf{a}, \\ \dfrac{\partial F_n}{\partial y} = \dfrac{\partial P}{\partial y}\,\mathbf{a}. \end{cases} \tag{6}$$

Finally, the gradient vector can be calculated out when coefficients a_k are determined. Theoretically, the vector may directly be estimated as presented in [30]. However, this method provides only the average slope value for a long range, rather than partial derivatives at a discrete point. Furthermore, the method apparently induces heavy computation load. The next section details a local regression strategy to calculate coefficients of $\mathbf{a}$.

3.2. Measurement Data Regression. Calculation of partial derivatives of function F_n at point $\mathrm{P}(x, y)$ desires presence of sensed and/or predicted values of S at positions nearby. How much such a measured sample at point $\mathrm{P}_i = (x_i, y_i)$ influences the calculation depends on their Euclidean distance ($\|P_i\text{-}P\|$). The closer to each other the two points are, the heavier the impact can be. This is quantified by the weighing factor of w. Lemma 1 helps translate the bivariate to univariate-wise representation. It is expressed as

$$F_n(x, y) = \sum_{k=0}^{N-1} p_k(x, y)\, a_k, \tag{7}$$

which eases the application of local univariate regression to formulate field Φ. Given a value of k, the corresponding monomial of $m_k = a_{i,j}\, x^{i-j}\, y^j$ is determined according to Equation (5). Coefficients a_k can be estimated based on a weighted least square method. At first, loss function $\mathrm{RSS}(\mathbf{a})$ is given by

$$\begin{aligned} \mathrm{RSS}(\mathbf{a}) = \sum_{i=1}^{m} &\left[F_n(x_i, y_i) - \sum_{k=0}^{N-1} p_k(x_i, y_i) a_k \right]^T \times w(x_i, y_i) \\ &\times \left[F_n(x_i, y_i) - \sum_{k=0}^{N-1} p_k(x_i, y_i) a_k \right], \end{aligned} \tag{8}$$

where $w(x, y)$ is the weighing function associated with the UAV's current position. It is supposed that ground sensors and dropped-down ones have just reported values of parameter S for at least m positions. Let us denote a vector $\mathbf{F} = [F_n(x_1, y_1), F_n(x_2, y_2), \cdots, F_n(x_m, y_m)]^T$ and a $m \times m$ diagonal matrix $\mathbf{W}$ whose coefficients $w_{i,j}$ are given by

$$w_{i,j} = \begin{cases} w(x_i, y_i) & \text{if } i = j, \\ 0 & \text{if } i \ne j. \end{cases} \tag{9}$$

We next extend scalar P to a $m \times N$ matrix $\mathbf{P}$ as follows:

$$\begin{aligned} \mathbf{P} &= [P(x_1, y_1), P(x_2, y_2), \cdots, P(x_m, y_m)]^T \\ &= \begin{bmatrix} p_0(x_1, y_1) & p_1(x_1, y_1) & \cdots & p_{N-1}(x_1, y_1) \\ p_0(x_2, y_2) & p_1(x_2, y_2) & \cdots & p_{N-1}(x_1, y_1) \\ \cdot & \cdot & \cdots & \\ \cdot & \cdot & \cdots & \\ \cdot & \cdot & \cdots & \\ p_0(x_m, y_m) & p_1(x_m, y_m) & \cdots & p_{N-1}(x_m, y_m) \end{bmatrix}. \end{aligned} \tag{10}$$

Subsequently, RSS($\mathbf{a}$) in Equation (8) is rewritten as

$$\mathrm{RSS}(\mathbf{a}) = Tr\left(\mathbf{W}(\mathbf{F} - \mathbf{P}\,\mathbf{a})^T(\mathbf{F} - \mathbf{P}\,\mathbf{a})\right). \tag{11}$$

Minimizing RSS($\mathbf{a}$) expressed in Equation (11) by letting its partial derivative with respect to $\mathbf{a}$ be 0 as in [31, 32], the solution is determined as

$$\mathbf{a} = \left(\mathbf{P}^T\mathbf{W}\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{W}\,\mathbf{F}. \tag{12}$$

As the number of coefficients in $\mathbf{a}$ equals N, for the solution $\mathbf{a}$ in Equation (12) to be deterministic, the UAV must acquire at least m measured samples in its vicinity as

$$m_{\sum} \geq m = \lambda\, m_{\sum} \geq \frac{(n+1)(n+2)}{2}, \tag{13}$$

where $m_{\sum}$ and λ are, respectively, the total measurement samples available and whose fraction was used as input for the regression.

Regarding the weighing matrix $\mathbf{W}$, values of $w(x, y)$ in the vicinity of the current UAV position (x_u, y_u) are set to comply with the Gaussian distribution as

$$w(x, y) = \exp\left(-\frac{(x - x_u)^2}{2\sigma_x^2} - \frac{(y - y_u)^2}{2\sigma_y^2}\right), \tag{14}$$

where σ_x^2 and σ_y^2 are, respectively, variances of x and y.

4. Dealing with Object Motion

If the target object does not change its location, the UAV is directed following the gradient vector until it finds out a peak (x_c, y_c) as described in Section 2.2. While searching and tracking a moving object, the UAV may nevertheless observe movement of its measurement cone. In this case, it needs to adjust its flying direction adaptively to the movement. This section presents how the adjustment should be made, followed by a description of tracking behaviors.

4.1. Updating Motion Information of Objects. Cone Ω of a static object is essentially immobile. Updating its information from ground sensors underneath is just a matter of enhancing the known part of the cone gradually. A searching UAV ceases its periodical update when the known part covers the position (x_c, y_c). In reality, the target event may nonetheless be associated with a mobile object. Recall that the object is detected by parameter S that is quickly restored to normal values once it moves away (e.g., light, radiation, and sound). The cone shape in such a case remains intact, despite its movement on the horizontal Oxy plane. Assume that the movement in the ith interval is characterized by a vector $\overrightarrow{d_i} = \overrightarrow{P_iP_{i+1}}$ as illustrated in Figure 4. Here, P_i is the position of the UAV at the beginning time t_i of the interval, and P_{i+1} is its expected image at time $t_i + T_{\mathrm{up}}$, where T_{up} is the period length of each update interval.

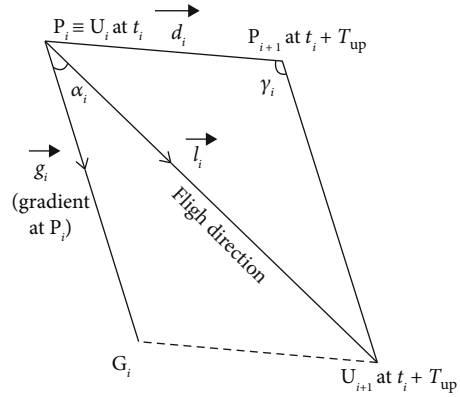


Figure 4: Self-navigation in early time of searching an object on the move—the UAV adjusts its flying direction on "wait-in-front" tactic, heading toward U_{i+1} instead of G_i.

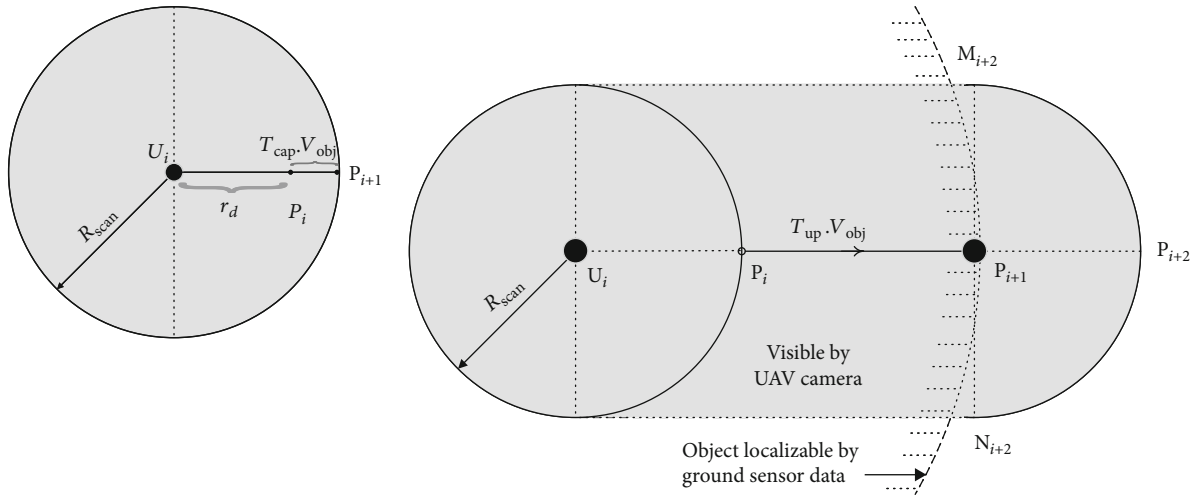
Coefficients a_k are time-varying and need updating regularly while the object is moving. Each update is performed right before the navigation decision for the interval is made. Because the cone shape remains unchanged, the following equation is always true at any time instant t_i:

$$\begin{aligned}\sum_{k=0}^{N-1} p_k\left(x + d_x^{(i-1)}, y + d_y^{(i-1)}\right) a_k(t_i) &\equiv \sum_{k=0}^{N-1} p_k(x, y)\, a_k\left(t_i + T_{\mathrm{up}}\right) \\ &\equiv \sum_{k=0}^{N-1} p_k\left(x - d_x^{(i)}, y - d_y^{(i)}\right) a_k(t_i),\end{aligned} \tag{15}$$

where $d_x^{(i)}$ and $d_y^{(i)}$ are, respectively, projections of vector $\overrightarrow{d_i}$ onto axis x and axis y. Equation (15) also helps the UAV detect the cone movement in the previous interval $(i-1)$, represented by vector $\overrightarrow{d_{i-1}}$. At the beginning time t_i of the i th update interval, the UAV is at U_i, whose gradient vector is $\overrightarrow{g_i}$. If the object stood fixed throughout the period, the UAV would move along $\overrightarrow{g_i}$. However, because the object is predicted to move according to vector $\overrightarrow{d_i}$, the UAV should move along the diagonal of the parallelogram, reaching U_{i+1} at the end of the period. At that time, the system also checks the value of S at position P_{i+1} against Equation (15). Calculation of deviation angle α_i between gradient vector $\overrightarrow{g_i}$ and motion vector $\overrightarrow{l_i}$ is needed to direct the UAV thereto. At first, recalling Equation (2), distance $\mathrm{P}_i\mathrm{G}_i = b_i$ is estimated as if the object did not move:

$$b_i = \int_{t_i}^{t_i + T_{up}} \kappa(t)\, |\overrightarrow{g_i}|\, dt. \tag{16}$$

The actual flying distance $\mathrm{U}_i\mathrm{U}_{i+1} = |\overrightarrow{l_i}|$ within the interval T_{up} is subsequently computed as



(a) Camera-visible tracking

(b) Ground sensor-assisted localization

Figure 5: Camera-visible range (a) and ground sensor-localizable range (b)—they influence assurance of tracking success.

$$\left|\overrightarrow{l_i}\right| = \sqrt{\left|\overrightarrow{d_i}\right|^2 + b_i^2 - 2\left|\overrightarrow{d_i}\right| b_i \cos\gamma_i} = \sqrt{\left|\overrightarrow{d_i}\right|^2 + b_i^2 + 2\left|\overrightarrow{d_i}\right| b_i \frac{\overrightarrow{g_i}\cdot\overrightarrow{d_i}}{\left|\overrightarrow{g_i}\right|\left|\overrightarrow{d_i}\right|}}, \quad (17)$$

where $\gamma_i = \pi - \arccos\left(\overrightarrow{g_i}\cdot\overrightarrow{d_i}/(|\overrightarrow{g_i}|\ |\overrightarrow{d_i}|)\right)$.

Finally, deviation angle α_i can be found at the UAV as

$$\alpha_i = \arccos \frac{b_i + |\overrightarrow{d_i}|\left(\overrightarrow{g_i}\cdot\overrightarrow{d_i}/|\overrightarrow{g_i}|\ |\overrightarrow{d_i}|\right)}{\sqrt{b_i^2 + \left|\overrightarrow{d_i}\right|^2 + 2\,b_i\,|\overrightarrow{d_i}|\left(\overrightarrow{g_i}\cdot\overrightarrow{d_i}/|\overrightarrow{g_i}|\ |\overrightarrow{d_i}|\right)}}. \quad (18)$$

The UAV is then self-navigated according to this deviation angle, i.e., along vector $\overrightarrow{l_i}$, instead of gradient $\overrightarrow{g_i}$.

4.2. Tracking Motion Objects. Once finding out the moving object, the UAV switches to tracking mode as depicted in Figure 3. As stated earlier in Section 2, on-board sensing devices, embedded cameras in particular, continuously help the UAV chase the object. To let the object be within its visible range, the tracking UAV must

(1) move to keep the orthogonal projection of the UAV onto the ground close to the object

(2) capture and process images frequently enough, so that

$$r_d + V_{obj}\,T_{cap} < R_{scan}, \quad (19)$$

where T_{cap} denotes the aforementioned capturing and processing interval, r_d is the distance between the UAV projection onto the ground and the object center, and R_{scan} is the scanning radius of the on-board sensing devices. Figure 5(a) illustrates this rule.

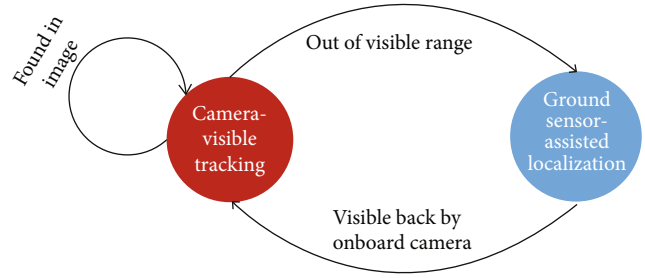


Figure 6: Sub-FSM of state *tracking* in Figure 3—the object is continuously tracked by on-board cameras or ground sensor-assisted localization.

On the other hand, if the object moves out of visible range, violating the above rule, the system falls in one of the following cases:

(1) The object moves absolutely far away, triggering signal *missing*. This brings the system to state *discovery* as indicated in Figure 3. The UAV then waits for further data to move back to the *searching* state, or more luckily, directly to any peak found

(2) The system can still position the object center thanks to sufficient data provided by ground sensors. This means that the object is still marked *found*

In the second case, if the UAV moves in the right direction in the next update interval and the scanning radius is large enough, it quickly "sees" the object again by the camera. Figure 6 illustrates these behaviors, further detailing state *tracking*.

```
/* initialization */;
peak_found = false;
object_found = false;
i =0;
/* searching */;
while (!peak_found) do
    U_i = current_position = P_i;
    execute local regression to determine a_k(t);
    determine gradient g_i at U_i according to Equation (6);
    if (a peak(x_c, y_c) is located) then
        peak_found = true;
        break;
    end
    calculate b_i according to Equation (16);
    locate P_{i+1} from Equation |d_i| = ||P_{i+1} − P_i||;
    estimate the deviated angle α_i according to Equation (18);
    calculate |l_i| according to Equation (17);
    navigate the UAV according to angle α;
    i++.
end
calculate coordinates of peak (x_c, y_c);
navigate the UAV in a straight line to (x_c, y_c);
capture ground image;
if object_found in image then
    object_found = true;
    switch to tracking mode;
else
    switch to discovery mode;
end
```

Algorithm 1: Monitoring a moving object: from searching to tracking.

Lemma 2. *As soon as a tracked object gets out of the visible range, the sufficient condition for the UAV to get it visible back is that*

$$R_{sens} > 2R_{scan} > 2V_{obj}T_{up}, \tag{20}$$

where R_{sens} and R_{scan}, respectively, denote the radius of object localizable and visible ranges.

Proof. Let us assume that at time instant t_i, the orthogonal projection point of the UAV onto the ground and the object center are located at U_i and P_i, respectively, as illustrated in Figure 5(b). After an update interval T_{up}, the object moves to the next positions P_{i+1} at time $t_i + T_{up}$.

In the worst case, the object center is located right at the border of circle (U_i, R_{scan}), and it quickly moves away from the visible range along the radius at velocity V_{obj}. This implicates a transient invisible time. Now that the on-board camera cannot see the object anymore, image capturing and processing do not help. The system must therefore rely on ground sensor data to assure its object localization. If the data are still available in the range

$$R_{sens} > R_{scan} + V_{obj}T_{up}, \tag{21}$$

then the object center is still localizable. In addition, as soon as the next update comes, the UAV gets to know position P_{i+1}, which is at distance of $V_{obj}T_{up}$ away from the previous location. The UAV accordingly directs itself to P_{i+1}, getting the camera-scanned region filled in gray as seen in Figure 5(b). Assuming that within a short update interval, the object does not significantly changes its velocity, the sufficient condition for the object to be seen by the camera is that

$$\frac{\|M_{i+2} - N_{i+2}\|}{2} > \|P_{i+2} - P_{i+1}\| \Leftrightarrow R_{scan} > V_{obj}T_{up}. \tag{22}$$

Inequalities (21) and (22) are obviously true if constraint (20) holds. This completes the proof.

Note that constraint (20) in Lemma 2 is not a necessary condition; i.e., the UAV may still fully keep track of the object even if it is violated. For an example, in case $R_{scan} \ll V_{obj}T_{up}$, the UAV can still recover the camera-visible state right in the next update interval, if the object moves slowly or within a limited turning angle.

Eventually, Algorithm 1 summarizes the process of chasing the moving object. It details the FSM previously presented in Section 2.2. [Re]calculation of flying direction is executed based on both gradient and deviation angle. The

algorithm also indicates how the UAV changes its steering tactic in different operation modes. Apparently, escaping from gradient-based driving does weaken the need of data acquisition and processing.

4.3. On-the-Fly Deployment of Sensors. While searching and tracking the moving object, the UAV expects to have sufficient collected data from ground sensors as soon as possible, so that the very object position is pinpointed accurately. Note however that sensors may be sparsely distributed or partially corrupted due to the event occurrence, raising the need to deploy on-the-fly ones by the UAV itself. The following positions should be considered dropping sensors down nearby:

(i) Predicted position of P_{i+1} when the UAV is at U_i

(ii) Expected arrival point U_{i+1}

(iii) Predicted object center position

(iv) Positions in the vicinity of the UAV to meet inequality (13)

(v) Surrounding positions to augment satisfaction of the conditions claimed in Lemma 2

At the beginning of each update period, if the system finds out the above points within the drop-down radius of the UAV, its carry-on sensors will get ready. Should more UAVs join in the mission, they should also be dispatched to cooperatively drop down. A dropping schedule may nonetheless be discarded if existing sensors nearby already transmit sufficient measurement samples.

5. Overhead Analysis

Now that the searching and tracking strategies have clearly been clarified, let us evaluate their complexity and data communication load. It can be inferred from the analytic model and algorithms presented in Sections 3 and 4 that the overhead is influenced by the UAV steering policy, operation mode, object mobility, data updating frequency, and regression parameters.

5.1. Complexity. The heaviest load pertains to matrix chain multiplication in Equation (12) to find out coefficients of vector **a**. This is in principle executed in each update interval as stated in Section 3.2. Recall however that it is carried out only in early searching stage. As soon as the UAV successfully locates the object center, the UAV is directed to the location without any extra calculation. In state *tracking* of FSM depicted in Figure 3, the UAV purely processes discrete ground images.

As indicated in Table 2, the complexity is $\mathbf{O}(2N^3 + N^2 m + 2Nm^2)$. Given that $N = ((n+1)(n+2))/2$ as explained in Section 3.2, the coefficients can be estimated in polynomial time of the degree of F_n and the number of sensor measurement samples (i.e., m) used for the regression. Our simulations shows that the object center is found within just a few ten update intervals, triggering signal *peak found* as indicated in Figure 3. The UAV subsequently stops regression computation. Note also that the cumulative calculation load depends on the frequency of data update as well.

Table 2: Computation load of steps in calculating coefficients of vector **a** according to Equation (12).

st.	Operation	Out size	Multiplications	Notes
1	$\mathbf{P^T W}$	$N \times m$	Nmm	
2	$\mathbf{P^T WP}$	$N \times N$	NmN	
3	$\left(\mathbf{P^T WP}\right)^{-1}$	$N \times N$	NNN	Inversion
4	$\mathbf{P^T W}$	$N \times m$	Nmm	Same as step 1
5	$\mathbf{P^T WF}$	$N \times 1$	Nmm	
6	$\left(\mathbf{P^T WP}\right)^{-1}\mathbf{PWF}$	$N \times 1$	NNN	
Total (step 4 omitted): $2N^3 + N^2 m + 2Nm^2$				

If an on-board camera is employed to confirm the existence of the object, training tasks must be executed followed by periodical object detection. UAVs nowadays are computationally powerful enough to get the jobs done [33]. They can anyway offload heavy training tasks to a cloudlet if wishing to reduce the edge processing load [34]. Upon acquiring the training results, they may easily and quickly carry out detection jobs.

5.2. Data Communication Load. Periodical local regression to update the gradient vector requires availability of measurement data from ground sensors. As mentioned earlier in Section 3, at least m sensor samples are needed for each local regression instance of $F_n(x, y)$. Data volume L_{reg} to be collected from ground sensors per regression is thus estimated as

$$L_{\mathrm{reg}} = m\, L_{\mathrm{msg}}, \tag{23}$$

where L_{msg} denotes the length of message carrying a measurement sample. Referring back to Equation (13), we may now calculate the total data throughput flowing from ground sensors to the UAV:

$$R_{\mathrm{data}} = \frac{(n+1)(n+2)}{2\, T_{\mathrm{up}}}\, L_{\mathrm{msg}}. \tag{24}$$

Note that the above data amount is desired only before the object position is located. As soon as signal *found* in the FSM depicted in Figure 3 is generated, the UAV, without regression update, flies straightforward to the detected center position.

6. Performance Evaluation Results

We verified the proposed strategy by constructing NS-3 and Matlab co-simulations. Data communication and formulation were realized in the well-known network simulation tool NS-3. Meanwhile, Simulink UAV Library for Robotics System Toolbox™ was employed to adjust and validate flying

Table 3: System parameter setup for simulation scenarios.

No.	Category	Parameter	Value	Notes
1	Network	Number of sensors	2500	
2	Network	Ground area	6000 m × 6000 m	
3	Network	Routing	AODV	
4	Communication link	Wireless standard	IEEE802.11	Protocol suite
5	Communication link	Tx power	16 dBm	Max value
6	Communication link	Rx sensitivity	-96 dBm	
7	Data	Message size	256 bytes	
8	Data	Tx period	2-10 s	Also update period
9	UAV	Max ground speed	15 m/s	
10	UAV	Max acceleration	2 m/s^2	
11	UAV	Scanning radius	30 m	By on-board camera
12	UAV	Max altitude	150 m	

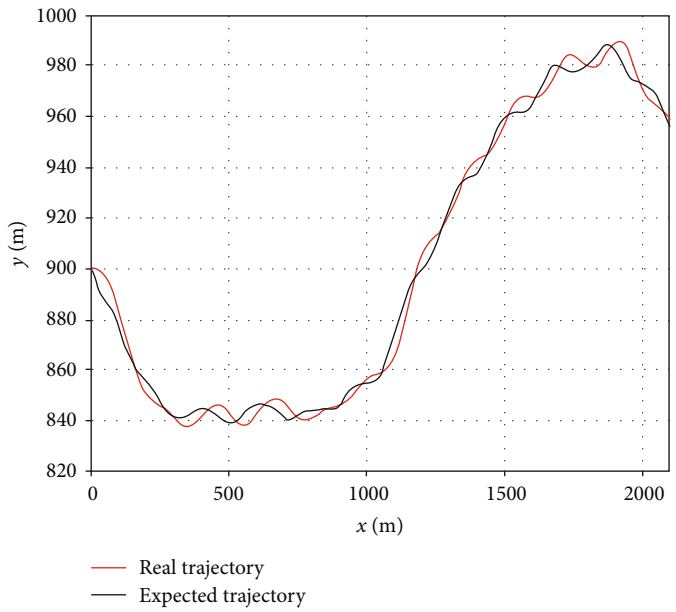
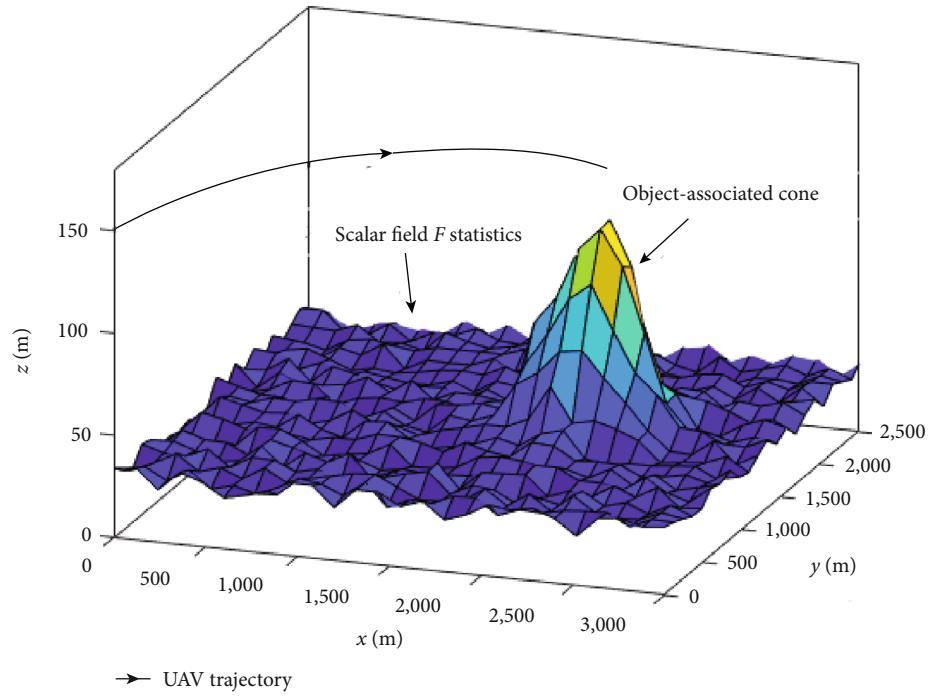


Figure 7: Expected trajectory (gradient-driven) and real trajectory (adjusted by Simulink)—only minor discrepancy is observed, even at the highest object velocity of 10 m/s.

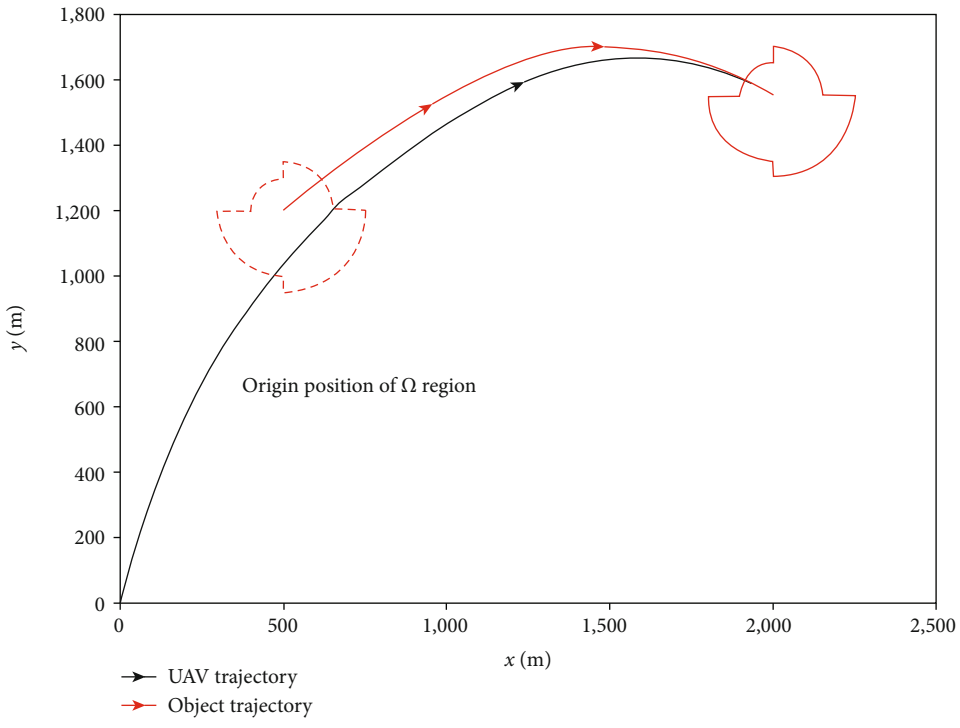
trajectories upon each self-navigation operation. Specifically, how the exact heading trajectory looks like in each update interval is shaped by the Simulink tool. With the application of the module, the navigating calculation has already taken into account external factors such as wind and atmospheric disturbances [27, 28]. Networking configurations and system setup are concretely described in Table 3. In all the simulation scenarios, sensor-measured data also underwent a Kalman filter-based preprocessing stage [18] to be cleaned.

Simulation results show that expected trajectories basically agree with those adjusted by Simulink, as an example seen in Figure 7. The UAV reached its expected arrival position in all update intervals.

While our proposed framework (hearafter called *online self-nav*) is novelly different, we attempted to simulate the best matching schemes we know, for comparative study. At first, the most promising state-of-the-art approach we know, *self-adaptive team* [8], was shaped to fit the context. As mentioned in Section 1, this scheme solely relies on on-board sensors to detect a gas emission event. For fair comparisons, the team of five UAVs thereon were, however, assumed to also communicate with ground sensors while they were flying. In addition, we also resimulated another relevant approach in organizing a dynamic team of UAVs for searching and tracking the event, to which further comparison is made. Namely, the PSO- (progressively spiral-out optimization-) based algorithm devised in [25] was numerically evaluated in the same system context. In this searching method, a time-varying team of UAVs were path-planned according to predefined rules for exhaustive scanning. They basically flied along spiral-out trajectories, whose bounded area is expected to cover the target moving region.



(a) UAV flies toward the object center position

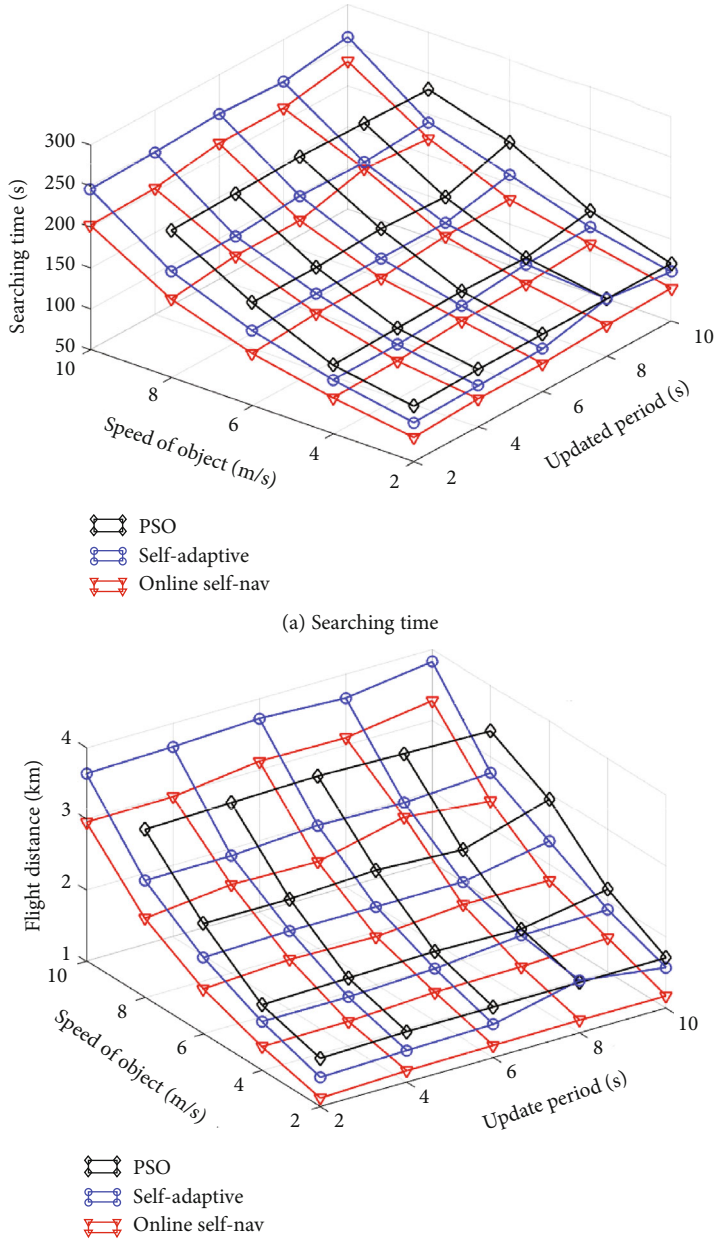


(b) Converged trajectories of the UAV and object

Figure 8: Movement of the UAV toward the moving target object. Gradient calculation in conjunction with "wait-in-front" self-navigation strategy helped the UAV catch the target after a relatively short time of searching.

6.1. Success of Online Self-Navigation. We let the UAV depart at the origin (0 m, 0 m), whereas the object initially arose at position (500 m, 1200 m). The object subsequently moved at variable speed within 10 m/s.

A typical simulation instance of searching is depicted in Figure 8. It shows how the UAV successfully approached the object. Afterwards, it switched to the tracking state (as previously mentioned in Section 2.2). Remarkably, the distance between the projection of the UAV onto the ground and the object position got gradually shorter during the search. At the update period of 8 s and object velocity of 8 m/s, the UAV eventually reached the target after 25 update



(a) Searching time

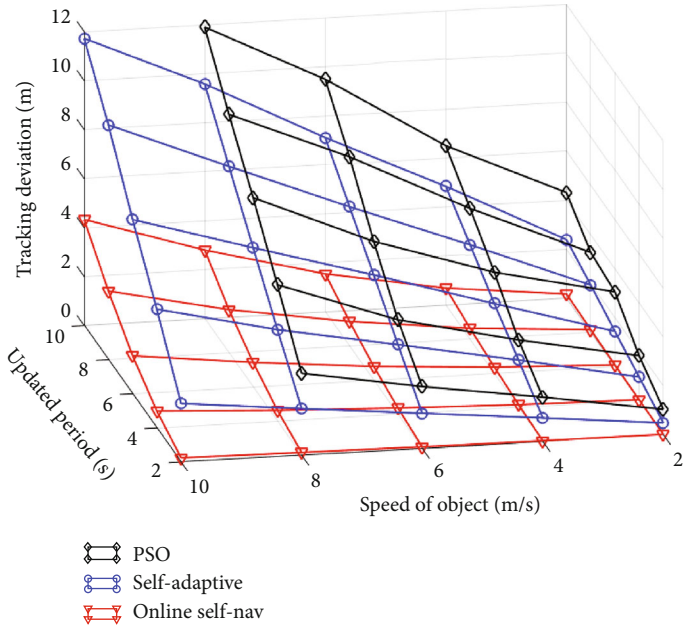
(b) Flying distance

Figure 9: Time and flying distance in the online self-navigation, self-adaptive, and PSO strategies before the UAV reaches the object and switches to *tracking* mode. Our proposed framework is annotated as "Online self-nav." The PSO team did not find out the object at a speed of 10 m/s.

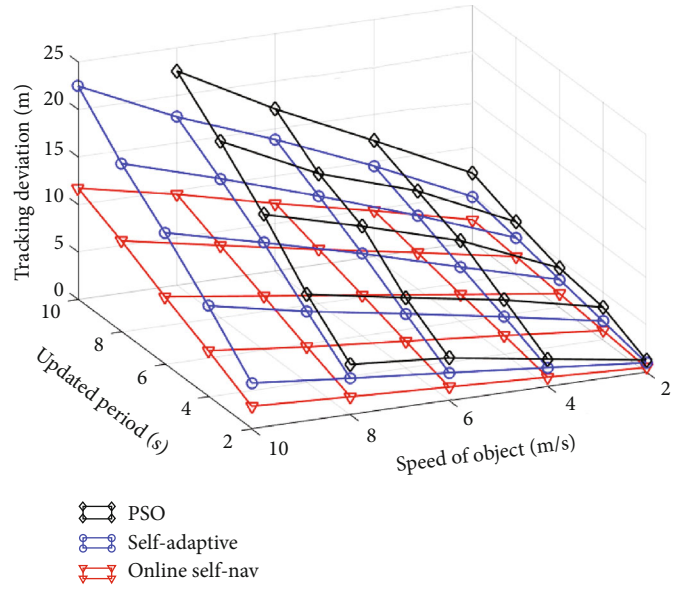
periods (around 200 s). The success consistently occurred in all the simulation instances we ever ran.

6.2. Searching Efficiency. Searching time and flying distance are key parameters to evaluate how well the UAV finds out the object. To make a fair comparison between our *online self-navigation* strategy and the *self-adaptive team* of five UAVs, we let our UAV be stationed at position (0 m, 900 m). Before taking off, the five UAVs of the team were, respectively, arranged at (0 m, 300 m); (0 m, 600 m); (0 m, 900 m); (0 m, 1200 m); and (0 m, 1500 m). The object initially arose at position (900 m, 900 m). Meanwhile, in the case of the aforementioned PSO scanning approach [25] (hereafter referred to as "PSO" for short), the team was initially formed by four UAVs. At the beginning of the search, they were located evenly at circle, centered at position (500 m, 900 m), with a radius of 20 m. During the search, the team called on another UAV to keep the search exhaustive. All the UAVs were also equipped with 30 m scanning cameras.

It can be inferred from Figure 9 that it is basically more costly, in terms of time and of flying distance, to find out the object when the gradient update interval (or *update period*) gets higher. This does implicate a trade-off between computation/communication and flying costs. However,



(a) Average tracking deviation



(b) Maximum tracking deviation

Figure 10: Tracking deviation error—the distance between the orthogonal projection of the UAV onto the ground and the object center during the tracking phase. The PSO team did not find out the object at a speed of 10 m/s.

the impact of the interval on the time and distance is not so significant as that of object mobility.

The figure shows that both temporal and spatial lengths are extended exponentially as the object velocity gets higher. Throughout all 25 scenarios formed by a combination of 5 update period (2 s, 4 s, 6 s, 8 s, and 10s) and object velocity (2 m/s, 4 m/s, 6 m/s, 8 m/s, and 10 m/s) values, we observe a consistent outperformance of our online self-navigation strategy compared to the other. As depicted in Figure 9, it took about 230 s (23 update periods) to reach the target object in the worst case (update period of 10s and object speed of 10 m/s). On the other hand, the self-adaptive team spent about 260 s on locating the object, which is about 13% longer. In this case, the distance difference is 0.54 km (3.82 km versus 3.28 km), meaning 16% longer. Note however that the cumulative flying distance of all the UAV team is up to 19.12 km, gravely longer than 3.28 km in ours. At the same period and speed, PSO observed a total fly distance of 4.41 km per UAV, but the team eventually failed to find out the object. The reason is that its velocity exceeded the limit set by the spiral-out radius and UAV mobility [25]. If the velocity was lowered to 8 m/s, it caught the object after about 230 s, which is 60s (or 35%) longer than ours. In this scenario, the average per-UAV flying distances of the online-self-nav,

self-adaptive, and PSO strategies are, respectively, 2.38 km, 2.78 km, and 3.36 km.

6.3. Tracking Accuracy. To evaluate how well the UAV keeps track of the moving object, we regularly collected their location data in the whole tracking process. Let us define *tracking deviation* at a time instant as the distance between the projection of the UAV onto the ground and the object center. Quantitatively, the average (D_{avg}) and maximum ($D_{\max}$) deviations are estimated as follows:

$$\begin{cases} D_{\text{avg}} = \sqrt{\dfrac{1}{N_s}\displaystyle\sum_{i=1}^{N_s}\left(x_u^{(i)} - x_c^{(i)}\right)^2 + \left(y_u^{(i)} - y_c^{(i)}\right)^2}, \\ D_{\max} = \displaystyle\max_{i=1,N_s}\left\{\sqrt{\left(x_u^{(i)} - x_c^{(i)}\right)^2 + \left(y_u^{(i)} - y_c^{(i)}\right)^2}\right\}, \end{cases} \tag{25}$$

where $(x_u^{(i)}, y_u^{(i)})$ and $(x_c^{(i)}, y_c^{(i)})$, respectively, represent locations of the UAV and object at update interval i and N_s is the number of positions used to calculate the tracking deviation.

As indicated in Figure 10, the deviation values essentially depend on object mobility more significantly than they do on update interval. In general, it is more challenging for the UAVs to track as the speed and the interval get greater. Specifically, in our proposed self-navigation strategy, the maximum and average deviations are only 0.01 m at an interval of 2 s and velocity of 2 m/s. They reach 4.91 m and 4.33 m at 10 s and 10 m/s, respectively. The deviation levels are anyway well within the scanning area of the on-board camera, whose radius is 30 m as indicated in Table 3.

The outperformance of our online navigation scheme, as seen in the chart, is clearly conclusive, with respect to both deviations. All the simulation instances observed better tracking accuracy in 25 pairs of update period and object velocity values, compared to the self-adaptive and PSO strategies. The maximum difference between the two pertaining to the pair of 10 s and 10 m/s is 17.55 m and 7.39 m on for $D_{\max}$ and D_{avg}, respectively. In the least challenging case (the pair of 2 s and 2 m/s), the self-adaptive team strategy incurs two deviations of 0.98 m and 0.48 m, which is far greater than ours—only 0.01 m as stated above. Meanwhile, the PSO team, respectively, suffers the deviations of 1.03 m and 1.26 m, which are gravely higher.

Overall, the online self-navigation framework brings up clear reduction with respect to searching time, flying distance, and tracking error, as listed in Table 4. As shown in the table, the worst performer was the PSO team.

6.4. Data Exchange Volume. Recall that data exchange between ground sensors and the UAV mainly occurs in the searching phase, when the object center position has not been located. As stated in Section 5.2, during the phase, the exchange volume depends on update interval T_{up} and the number of measurement samples transmitted. Basically, the amount should monotonically decrease as the update interval is lengthened. With respect to object mobility, the load also gets heavier as its velocity increases. This can be explained that more ground sensors gets involved in providing measurement data to cover a larger motion area.

Table 4: Average reduction of searching time, flying distance, and tracking deviation observed in our proposed WSN-assisted navigation, compared to self-adaptive and PSO strategies.

Strategy	Searching time (s)	Flying distance (km)	Tracking deviation
PSO team	194.32	2.82	5.06
Self-adaptive team	161.92	2.37	4.28
Self-nav.	136.64	1.93	0.84
Reduction from PSO	29.68%	31.56%	83.40%
Reduction from self-adapt.	15.61%	18.57%	80.37%

Figure 11 truly reflects this argument. With a message length of 256 bytes as indicated in Table 3, the data amount reaches 9.17 MB, 5.99 MB, 4.42 MB, 3.54 MB, and 2.97 MB, corresponding to object velocity of 10 m/s, 8 m/s, 6 m/s, 4 m/s, and 2 m/s, all at a minimum update interval of 2 s. When the update takes place less frequently, the amount is clearly reduced. The reduction intensity nonetheless goes down against the increasing order of update period. Note that the plotted statistics also cover the tracking phase, during which data exchange occurs sporadically (as explained in Sections 2.2 and 5.2).

6.5. Energy Consumption. In reality, a UAV spends its energy on flying motors, regression computation, and data communication with ground sensors. The first part accounts for major consumption and depends on motion parameters and flying time. The consumed power $\mathscr{P}$ is roughly approximated below:

$$\mathscr{P} = \mathscr{P}_0 + \beta V_u, \tag{26}$$

where $\mathscr{P}_0$ is the consumed power when the UAV hovers and β is a proportional factor. In this study, we do not deeply analyze consumption on taking off and landing, since the operations are assumed to take place distantly from the searching and tracking region. In the simulations, we set $\beta = 10$ and $\mathscr{P}_0 = 150$ W. Figure 12 plots consumption statistics for gradient update periods of 2 s, 4 s, 6 s, 8 s, and 10 s. Different velocities of the object were also checked to make the results more comprehensive.

The chart shows that the total energy consumed gets basically higher as the object moves faster. The gradient update interval nevertheless does not adhere to the same rule. This may be attributed to random vibration of motion trajectories. In contrast, it can be noticed that the consumption amount is drastically boosted against the object mobility. In all the simulations, the values range from 24.13 KJ at an update period of 2 s and object velocity of 2 m/s to 68.45 KJ at 10 s and 10 m/s, respectively.

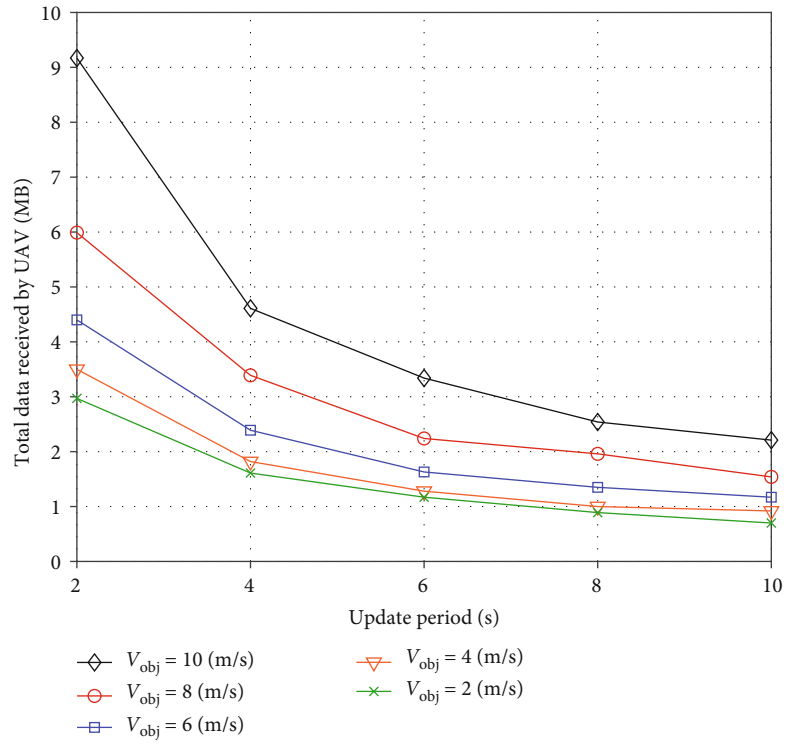


Figure 11: Data load exchanged between UAV and sensors. The volume strongly depends on velocity of the target object, especially when data update interval is short.

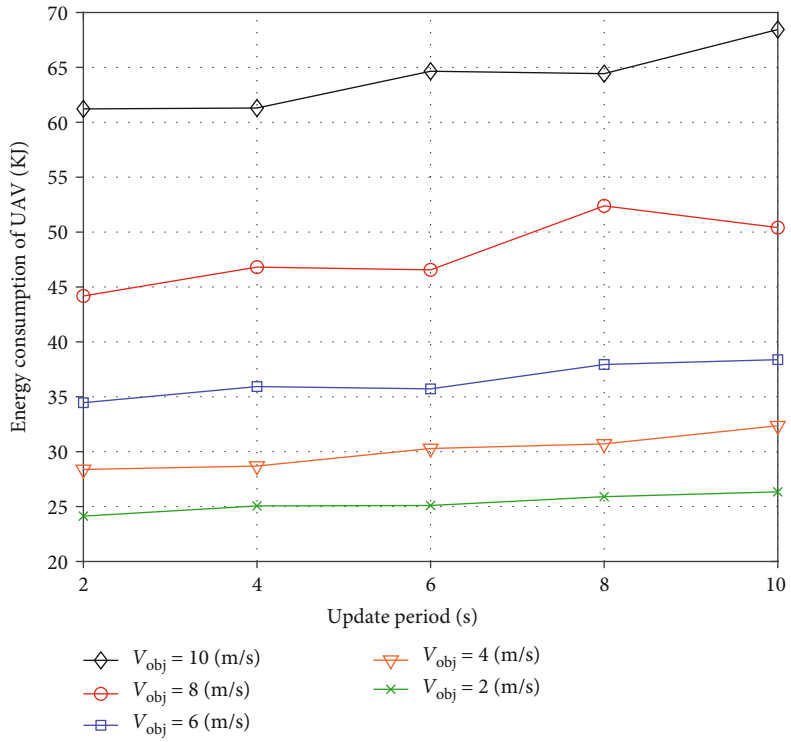


Figure 12: Energy consumption on searching—object movement again has more apparent impact than update period does.

7. Conclusions

We have presented an online self-navigation framework in which UAVs regularly update measurement data on the move. Periodical formulation of acquired data brings up helpful hints to steer UAVs toward moving objects. The local regression-based formulation enables calculating the gradient vector at any instant position. The vector is a good reference for steering the vehicles, especially in early searching time, when available information about targets is limited. Once ground data are sufficient, the formulation also helps instantly locate the exact position of objects. Flexibly switching between searching, tracking, and discovering states, the comprehensive maneuvering strategy considerably reduces both computation and communication loads.

Numerical results of NS-3 and Matlab cosimulations consistently agree with the theoretical expectation. Our proposed framework clearly shows its outperformance in searching and tracking. Compared to the best state-of-the-art method we know, reduction of over 15%, 18%, and 80% is, respectively, observed with respect to searching time, flying distance, and tracking deviation. Simulation statistics also indicate how the supervising performance depends on object mobility and frequency of data update.

Data Availability

Simulation experiments were carried out using NS-3 and Matlab. All data, except those of sensor node configurations and motion trajectories, used to support the findings of this study are included within the article. The latter are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgments

Our research was partially sponsored by the Ministry of Science and Technology of Vietnam under research project "Research and development of Internet of Things platform (IoT), application to management of high technology, industrial zones," mission code KC.01.17/16-20.

References

[1] H. J. Na and S. J. Yoo, "PSO-based dynamic UAV positioning algorithm for sensing information acquisition in wireless sensor networks," *IEEE Access*, vol. 7, pp. 77499–77513, 2019.

[2] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "A framework for using unmanned aerial vehicles for data collection in linear wireless sensor networks," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1-2, pp. 437–453, 2014.

[3] A. A. Awan, M. A. Khan, A. N. Malik et al., "Quality of service-based node relocation technique for mobile sensor networks," *Hindawi Wireless Communications and Mobile Computing*, vol. 2019, article 5043187, 13 pages, 2019.

[4] A. Tripathi, H. P. Gupta, T. Dutta, D. Kumar, S. Jit, and K. K. Shukla, "A target tracking system using directional nodes in wireless sensor networks," *IEEE Systems Journal*, vol. 13, no. 2, pp. 1618–1627, 2019.

[5] M. Vazquez-Olguin, Y. S. Shmaliy, and O. Ibarra-Manzano, "Object tracking over distributed WSNs with consensus on estimates and missing data," *IEEE Access*, vol. 7, pp. 39448–39458, 2019.

[6] W. Meng, Z. He, R. Su, P. K. Yadav, and R. Teo, "Decentralized multi-UAV flight autonomy for moving convoys search and track," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1480–1487, 2017.

[7] R. R. Pitre, X. R. Li, and R. Delbalzo, "UAV route planning for joint search and track missions—an information-value approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2551–2565, 2012.

[8] H. Yuan, C. Xiao, W. Zhan et al., "Target detection, positioning and tracking using new UAV gas sensor systems: simulation and analysis," *Springer Journal of Intelligent & Robotic Systems*, vol. 94, no. 3-4, pp. 871–882, 2019.

[9] S. Mini, S. K. Udgata, and S. L. Sabat, "*M*-Connected Coverage Problem in Wireless Sensor Networks," *ISRN Sensor Networks*, vol. 2012, Article ID 858021, 9 pages, 2012.

[10] Z. Kang, H. Zeng, H. Hu, Q. Xiong, and G. Xu, "Multi-objective optimized connectivity restoring of disjoint segments using mobile data collectors in wireless sensor network," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, 2017.

[11] A. Ez-Zaidi and S. Rakrak, "A comparative study of target tracking approaches in wireless sensor networks," *Hindawi Journal of Sensors*, vol. 2016, article 3270659, 11 pages, 2016.

[12] K. L. Ang, J. K. P. Seng, and A. M. Zungeru, "Optimizing energy consumption for big data collection in large-scale wireless sensor networks with mobile collectors," *IEEE Systems Journal*, vol. 12, no. 1, pp. 616–626, 2018.

[13] J. Zhang, P. Hu, F. Xie, J. Long, and A. He, "An energy efficient and reliable in-network data aggregation scheme for WSN," *IEEE Access*, vol. 6, pp. 71857–71870, 2018.

[14] O. Demigha, W. Hidouci, and T. Ahmed, "On energy efficiency in collaborative target tracking in wireless sensor network: a review," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1210–1222, 2013.

[15] A. Sharma, P. K. Singh, A. Sharma, and R. Kumar, "An efficient architecture for the accurate detection and monitoring of an event through the sky," *Elsevier Computer Communications*, vol. 148, pp. 115–128, 2019.

[16] J. Du, J. Diouris, and Y. Wang, "A RSSI-based parameter tracking strategy for constrained position localization," *EURASIP Journal on Advances in Signal Processing*, vol. 2017, no. 1, 2017.

[17] Y. Yao, Q. Han, X. Xu, and N. Jiang, "A RSSI-based distributed weighted search localization algorithm for WSNs," *International Journal of Distributed Sensor Networks*, vol. 11, no. 4, Article ID 293403, 2015.

[18] S. Mahfouz, F. Mourad-Chehade, P. Honeine, and J. Farah, "Target tracking using machine learning and Kalman filter in wireless sensor networks," *IEEE Sensors Journal*, vol. 14, no. 10, pp. 3715–3725, 2014.

[19] S. Xiao, W. Li, H. Jiang, Z. Xu, and Z. Hu, "Trajectroy prediction for target tracking using acoustic and image hybrid wireless multimedia sensors networks," *Springer Multimedia Tools and Applications*, vol. 77, no. 10, pp. 12003–12022, 2018.

[20] R. Opromolla, G. Fasano, and D. Accardo, "A vision-based approach to UAV detection and tracking in cooperative applications," *Sensors*, vol. 18, no. 10, article 3391, 2018.

[21] X. Zhao, F. Pu, Z. Wang, H. Chen, and Z. Xu, "Detection, tracking, and geolocation of moving vehicle from UAV using monocular camera," *IEEE Access*, vol. 7, pp. 101160–101170, 2019.

[22] D. Cavaliere, V. Loia, A. Saggese, S. Senatore, and M. Vento, "A human-like description of scene events for a proper UAV-based video content analysis," *Knowledge-Based Systems*, vol. 178, no. 2, pp. 69–74, 2019.

[23] D. Cavaliere, V. Loia, A. Saggese, and S. Senatore, "Semantically enhanced UAVs to increase the aerial scene understanding," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 3, pp. 555–567, 2019.

[24] D. Cavaliere, V. Loia, and S. Senatore, "Towards an ontology design pattern for UAV video content analysis," *IEEE Access*, vol. 7, pp. 105342–105353, 2019.

[25] D. Brown and L. Sun, "Dynamic exhaustive mobile target search using unmanned aerial vehicles," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 6, pp. 3413–3423, 2019.

[26] S. Lian, Y. He, and J. Zhao, "CCD: locating event in wireless sensor network without locations," in *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, Valencia, Spain, 2011.

[27] R. W. Beard and T. W. McLain, "Small unmanned aircraft theory and practice," Princeton University Press, NJ, 2012, Chapter 4.

[28] MathWorks Robotics Team, *Robotics System Toolbox UAV Library*, Mathworks, 2019, https://www.mathworks.com/matlabcentral/fileexchange/68788-robotics-system-toolbox-uav-library.

[29] A. Ariffin, N. Aziz, and K. Othman, "Implementation of GPS for location tracking," in *2011 IEEE Control and System Graduate Research Colloquium*, Shah Alam, Malaysia, 2011.

[30] Q. Li, X. Lu, and A. Ullah, "Multivariate local polynomial regression for estimating average derivatives," *Journal of Nonparametric Statistics*, vol. 15, no. 4-5, pp. 607–624, 2003.

[31] J. Fan, I. Gijbels, T. Hu, and L. Huang, "A study of variable bandwidth selection for local polynomial regression," *Statistica Sinica*, vol. 6, no. 1, pp. 113–127, 1996.

[32] J. Taylor, *Strategies for mean and modal multivariate local regression, [Ph.D. thesis]*, Durham University, 2012.

[33] W. Chen, Z. Baojun, T. Linbo, and Z. Boya, "Small vehicles detection based on UAV," *The Journal of Engineering*, vol. 2019, no. 21, pp. 7894–7897, 2019.

[34] M. F. Pinto, A. L. M. Marcato, A. G. Melo, L. M. Honório, and C. Urdiales, "A framework for analyzing fog-cloud computing cooperation applied to information processing of UAVs," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 7497924, 14 pages, 2019.