applied sciences

MDPI

Article

UAV Deployment Using Two Levels of Stigmergy for Unstructured Environments

Fidel Aznar [*,†], Maria del Mar Pujol López [†] and Ramón Rizo [†]

Department of Computer Science and Artificial Intelligence, University of Alicante, San Vicente del Raspeig, 03690 Alicante, Spain; mar.pujol@ua.es (M.d.M.P.L.); ramon.rizo@ua.es (R.R.)

* Correspondence: fidel@dccia.ua.es

† These authors contributed equally to this work.

Received: 29 September 2020; Accepted: 27 October 2020; Published: 30 October 2020

Abstract: This article will present two swarming behaviors for deployment in unstructured environments using unmanned aerial vehicles (UAVs). These behaviors will use stigmergy for communication. We found that there are currently few realistic deployment approaches that use stigmergy, due mainly to the difficulty of building transmitters and receivers for this type of communication. In this paper, we will provide the microscopic design of two behaviors with different technological and information requirements. We will compare them and also investigate how the number of agents influences the deployment. In this work, these behaviors will be exhaustively analyzed, taking into account different take-off time interval strategies, the number of collisions, and the time and energy required by the swarm. Numerous simulations will be conducted using unstructured maps generated at random, which will enable the establishment of the general functioning of the behaviors independently of the map used. Finally, we will show how both behaviors are capable of achieving the required deployment task in terms of covering time and energy consumed by the swarm. We will discuss how, depending on the type of map used, this task can be performed at a lower cost without using a more informed (but expensive) robotic swarm.

Keywords: swarm robotics; UAV deployment; stigmergy behavior

1. Introduction

Swarm robotics is an approach for the coordination of multirobot systems inspired by the emerging patterns observed in social insects. In this type of robotics, we work with large groups of robots so that miniaturization and cost reduction of the units are both crucial factors. In swarm robotics, individuals do not have to be cognitive, but cognition emerges from their interactions. There are many examples of this type of behavior in nature: bees use swarms to find food sources, ants collaboratively transport food, and fireflies use these strategies for synchronization.

In the field of robotics, multiple systems have been developed based on this approach, for example, systems focused on the formation of patterns (aggregation, self-organization, deployment, coverage, mapping), search (search for a goal, homing, foraging, etc.), and group behavior (cooperative transport, shepherding, flocking, etc.).

Among these behaviors, we are interested in the development of swarm behavior for deployment in unstructured areas. The coordination of robot teams for exploration is a complex problem, even more so when facing large areas, for example, in a devastated area after a disaster [1].

In these types of environments, unmanned aerial vehicles (UAVs) are a great advantage, given their ease of avoiding obstacles and their reduced sensory deprivation. However, these vehicles have problems with autonomy and reduced loads due to the small vehicles/weight [2].

We are currently experiencing a time of miniaturization and cost reduction for these devices, going from drones weighing 20 g (autonomies of 10 min, with costs close to 25$ per individual) to drones that are more autonomous, weighing less than 200 g (with greater load capacity and autonomies close to 15 min with a cost of approximately 100 $).

Miniaturization and cost reduction make it possible to use swarm techniques to model systems that contain a large number of individuals. This type of device satisfies the precepts of swarm robotics: simple individuals, with limited sensors and limited autonomy but with the possibility of having a large group to take advantage of their great mobility.

Therefore, it seems logical to propose the design of a behavior, oriented to UAVs, that enables the development of tasks in difficult, unstructured environments, with both signal availability and communication limitations. In addition, this behavior must take the limitations of the individual agents into consideration (e.g., the load capacity of micro-drones is on the order of 5 g, which means that the sensors that can be used in these devices are very limited).

In this article, two deployment behaviors for unstructured environments will be presented. These are distributed, decentralized, scalable behaviors that do not require global localization (and therefore are applicable to indoor environments). Both will be based on one of the most important indirect mechanisms of decentralized communication in swarm intelligence, stigmergy.

In stigmergy, information is stored in the environment, which makes it possible to coordinate agents. This type of communication has several advantages that make it especially attractive for swarm systems: it is simple, scalable, and robust and can be integrated with the environment. However, stigmergy is also one of the most complex communication systems to be implemented in real robots. Taking this aspect into account, in this work, we will consider two different stigmergy approaches without losing sight of what characteristics would be required for final implementation.

The first behavior is oriented to swarms with individuals with very limited perception capacities who are not able to locate themselves locally. Therefore, this behavior will not require localization capabilities of any kind and will require the local communication of only minimal information. Our intention is that this behavior will be easy implementable with existing hardware. The second behavior will require local localization capability, providing a greater convergence but requiring a sensor, such as the one presented in [3] or more generally in [4], to support localization.

With this article, we wish to answer mainly the following questions: can we develop a swarm behavior that uses stigmergy without the need to design specific hardware? How does the information available to agents affect the development of the behavior? How does the number of parallel agents influence the task? Does the energy consumed by the swarm depend on the sensor/emitter of stigmergy?

The rest of this article will be structured as follows. On the one hand, the state-of-the-art will be presented, with an exhaustive analysis of the existing deployment strategies. Special emphasis will be placed on those strategies that consider the special characteristics of UAVs. Next, we will describe in greater detail the problem of deployment in unstructured environments to be modeled, together with the type of swarm and individuals to be used. The simulator used and the metrics that will be used later will also be detailed. Subsequently, the two microscopic swarm behaviors will be presented, along with a description of their functioning and the roles of the individuals that compose them. In the results section, a set of tests will be developed to verify the behaviors and to compare both strategies for the aggregation of individuals and their operation in different types of environments (simple, complex, and hard-to-access areas). Finally, the results will be discussed, and the article will be concluded by proposing potential future lines of research.

2. Related Work

Multirobot deployment has become a fundamental research topic in the field of multirobot systems. In this section, we will provide a brief review of the existing types of swarm deployment and coverage of terrestrial robots. Then, we will review some applications designed to be executed using aerial vehicles and comment on some examples of stigmergy communication in swarm systems.

Although many applications are related to multirobot deployment, many of them rely on a preestablished communication network or external localization service. This precludes application to emergency tasks, where we are not able to assume the existence of communication networks or GPS. Therefore, there are several studies where the external communication network is substituted by the deployment of a new network of sensor nodes or radio beacons. For example, in [5] uses a distributed algorithm to guide a group of robots through a network of sensors, in [6] defines an artificial pheromone map which consists of arbitrary graph of special devices called "place agents". However, this and other similar strategies, such as [7], define the prior deployment of the sensor network, which may be difficult or not practical in emergency situations.

Another alternative is to include the deployment process in the behavior. In this case, the behavior consists of two phases: beacon deployment and robot deployment. [8] deploys a network of static radio beacons to form a long-range communication network to aid robot exploration. However, in a general way, this strategy cannot be used with micro or small-sized UAVs (the most commonly used type of UAVs in indoor environments) because of their low load capacity and limited autonomy [9].

In [10,11], a different strategy is presented, where each individual integrates a beacon, creating their own communications network. However, the design of this behavior is intended for the physical characteristics of terrestrial robots s-bot, which makes it very difficult to generalize to other types of vehicles due to their different types of sensors and actuators.

In our case, we are interested in developing behavior that uses the advantages of indoor flying vehicles, rather than terrestrial ones, in emergency situations. For example, these vehicles can avoid many obstacles that limit ground vehicles and enable sensor data to be obtained from a vantage point [12–16].

This type of device requires specific considerations and algorithms to ensure navigability [4,17]. In outdoor UAVs, GPS, which can be adjusted by using onboard inertial sensors, is the most common localization system. For example, in [18], stigmergy is used in a behavior to search for a target. This behavior requires all the agents to use global location to be able to perceive and place virtual pheromones in a specific location.

However, such systems cannot be used indoors (signal reception occlusion) or in all outdoor locations and situations [19], so another approximation, such as local localization, must be considered.

Robot local localization commonly requires environment maps and odometry sensing [20]. Environment maps may be unavailable, and the online creation of maps requires powerful processing that may not be available on small flying robots [4,21]. Furthermore, such approaches do not scale appropriately with large swarms [22]. Moreover, there are simple relative localization applications that are suitable for small UAVs; for example, in [23] presents the relative positioning of sensors in reference to nearby static robots with a decentralized approach. Nowadays, one of the main technologies required to make the leap from flying a single UAV to flying a decentralized swarm is an accurate and reliable intra-swarm relative localization technology [4].

One of the first applications related to UAV deployment can be found in [24]. This application uses a predetermined communication network as the basis for guiding UAVs. Although this behavior does not require an external communications network, it does need a predeployment phase, as in the terrestrial application in [5], which is not applicable in real emergency situations. In a later work, in [25] reconfigures the network of sensors to maximize system performance, but the method requires a group of terrestrial robots to transport the sensor network, which is not practical for our needs.

More recently, ref. [23,26,27] present several behaviors where a fully distributed deployment is achieved while creating a communication network. These systems do not require existing communication networks or a global localization system. In addition, it is an effective strategy that is designed to save energy when using UAVs, which is very important for increasing swarm autonomy. in [23,27] assume that a robot can temporarily attach to the ceiling or land on the ground for efficient surveillance over extended periods of time. However, as the authors discuss, this behavior requires special environment features (straight walls, 90 degree angles, etc.) to be executed (it is not applicable for unstructured environments) and does not take into account uncertainty (for example, the errors in sensor readings and robot actuators), which is very important in UAVs systems.

Using the ideas presented in [23,26–29], a UAV swarm can establish their own communications network. This network can be used for communicating swarm tasks or even to help other robots or a human team with subsequent work [5,10,11,24,30,31].

In this paper, we will present a behavior inspired by the work of [28], in terms of establishing their own network of communications, and by [23,27], in terms of adapting the behavior to UAVs using a power saving state. However, in contrast to previous articles, our distributed behavior will be executable in any environment, without limitations on the specific features or size.

For this purpose, a virtual sematectonic stigmergy system will be used. The use of this type of communication is not novel in the field of swarm robotics; the literature contains several applications using stigmergy as a communication mechanism. In [32], swarm behavior that can develop collaborative tasks through sematectonic stigmergy is presented. ref. [33] discusses a theoretical approach to the use of stigmergy in UAVs using stigmergy potential fields. To implement this type of communication, it is assumed that we have a local positioning mechanism [4] and a efficient system of communication between individuals, for example, the use of mesh networks of XBEEs devices [34].

We want to highlight that the probabilistic microscopic model and the stigmergy system that will be presented here take into account the inherent uncertainty of real robotic systems, which is not considered in other approaches that assume ideal worlds, such as [23,27,29].

3. Methods

This section will initially detail the swarm on which the deployment task is planned. Details of the types of base vehicles and the swarm simulator developed for this work will also be presented. Additionally, the process of generating unstructured environments using a cellular automaton will be described briefly. The two stigmergy-based swarm behaviors for deployment tasks will be described in detail.

Stigmergy provides an indirect means of communication between individuals through the environment. There are several types of stigmergy depending on how the communication is achieved. If communication is established through the placement of markers in the environment, we call the approach marker-based stigmergy. If individuals base their actions on the state of the solution, we refer to the approach as sematectonic stigmergy. If we work with discrete signals, we refer to strategy as qualitative stigmergy. On the other hand, if the communicated signals are continuous, we refer to the approach as quantitative stigmergy.

In the first behavior, an approximation of quantitative sematectonic stigmergy will be presented. In this case, some individuals will be deposited in areas of the environment and used as the main elements in the communication of pheromones; that is, the individuals themselves are defined as active markers of the environment.

The second behavior will use a hybrid of sematectonic, marker-based, and quantitative stigmergy. In this case, a group of individuals will be deposited as markers, but these markers will form a local gradient that will indicate the direction of navigation. The gradient of a pheromone map is managed by this same group of individuals.

3.1. Description of the Swarm

3.1.1. The Individuals

As discussed above, our work will focus on the use of pheromones to design various swarm behaviors capable of performing a deployment task in a structured environment. We will use a swarm robotics simulator, which we will detail later, to assess and validate these behaviors. We design a plausible physical swarm close to our physical robots. Then, we will describe the two types of individuals that will be modeled in our simulator.

We will focus on two types of UAVs, which we will call micro informed (micro) and mini informed (mini), depending on the information and, therefore, the sensorization available to them. We consider vehicles that do not exceed 200 g and that range in size from 6 cm to a maximum of 25 cm. We will assume that the linear speeds they can achieve for safe indoor navigation do not exceed 1 m/s. Both types of drones have a flight control board that allows simple movement commands to be sent to the drone for later execution.

The input commands may be high level (rotate 90^{o}, advance 1 m), similar to those used by the current consumer drones (https://www.parrot.com/, https://www.dji.com/). To simplify the interface, we will assume that we can send a displacement vector to the drones with the information of the movement to be performed at a given moment and that the control board will adjust the state of the drone according to this information.

Both types of drones use the same set of basic sensors. They will have IMU inertial unit data and a digital compass to obtain their orientation relative to north. These sensors are widely used and integrated by default in most control boards and commercial systems. In addition, the microdrones will have a barometric sensor (which makes it easy to stabilize the flight height), and the *mini* drones will have an ultrasound system in addition to the barometric system, for low-altitude flight. The *mini* drones also have a GPS tracking system, although it will not be used in this work since we intend to verify the behaviors presented here in indoor environments or where the GPS signal is not reliable.

We also require the drones to be able to avoid indoor obstacles. We will assume that they have one of the many currently available avoidance systems, although due to the size of the devices the most appropriate system is visual navigation. We do not require high accuracy since we need only a repulsion vector to the area in the visual field of the drone that contains more obstacles.

The drones can perform detection among themselves by using BLE beacons and know if they are near, far or out of range (for example, using the protocol established by Apple ibeacons (https://developer.apple.com/ibeacon/)). If they are close and in the field of vision, the obstacle avoidance system will return a repulsion vector toward the position of the detected drone. The microdrones will not have a relative location system, but the minidrones will. We will assume that they use a system similar to [3] that allows them to be located in the near space. The Bluetooth technology used for the location will also be used for short-range communications among drones. The autonomy of the drones will be fixed at 15 min of continuous flight. However, as will be discussed later, autonomy increases considerably if other tasks that save the energy consumed by the engines are performed.

As will be seen later, the proposed pheromone system can be implemented directly with the drones described above without any additional sensors.

3.1.2. The Swarm

The proposed drones are characterized by their simplicity and low cost, which limits the type of sensors to be used but makes it possible to use large swarms. With a cost of less than $20 per individual in the case of micro drones, we can raise swarms of hundreds of agents. The behaviors that follow will have this philosophy: with a system of many drones of very low cost, a collision not only does not hinder the achievement of the desired behavior, but can actually help, as we will see later.

Given the simplicity of the individuals, we will not have sensors for the concentration of individuals, such as those proposed in [27]. However, we assume that we can perform multipoint communications, such as can be achieved with the devices presented in [34]. These communications could be used indirectly to determine the concentration or, as in the case of this article, to articulate a virtual evaporation system, as required by the pheromone system presented below.

The behaviors that we will see below do not allow redeployment since, given the characteristics of the swarm enumerated previously, it does not contribute substantially to the development. However, it could be added in the future without much difficulty (if the autonomy of the drones and a limited number of them require such an addition). Therefore, the swarm coverage metric will be determined as the area currently seen by the drones, given their position. We will assume that they have a sensor that allows them to see a concentric area centered on their position.

Our intention is to be able to use this behavior to inspect unstructured indoor areas that are difficult to access, such as buildings after a disaster and caves. We will therefore assume that all drones will be launched from the same position: the main (possibly only) entrance.

To verify the operation of the swarm depending on the developed behavior, we will use additional metrics. We will analyze the total flight time of the swarm and the time required to perform deployment, and we will verify the average number of collisions (both between drones and between drones and the environment). Finally, as mentioned in [27,29], in this type of vehicle, it is essential to consider the energy required by the swarm. We will therefore analyze the average energy consumed by the swarm to complete deployment in an environment.

3.1.3. Simulator and Environments

In swarm robotics, it is essential to develop an exhaustive analysis of the simulated behavior before real implementation of the system. This article focuses on this analysis, although we have attempted to not leave out how this physical concretion would work.

Hundreds of simulations have been developed to test behaviors, so we require a fast simulator to be able to test hundreds of agents in a multitude of environments. Therefore, we use our simulator, called Multi Agent Swarm simulator in SCAla (MASCA) for the development of the tests. This simulator is briefly described below.

MASCA is based on the MASON platform [35], which is a multiagent simulation platform that can simulate any type of process. MASON provides facilities for simulations in text mode, 2D and 3D. It has a multitude of simulation-oriented data structures that are optimally developed in terms of access times to allow ultra-fast simulations. We have developed MASCA on this platform. Our aim was to provide a system capable of rapidly simulating swarms with thousands of individuals. We therefore want to focus on simulating the emergency process closest to swarm intelligence, where interaction and simulation with large swarms is fundamental. This is not often the goal of simulators, where there are marked limitations on the number of agents and the size of environments that can be simulated [35]. The next step (when we know that the proposed global behavior works as expected), which is beyond the scope of this article, focuses on the control, performing a more realistic physical simulation (dynamics of fluids, trajectories and 3D collisions), with a much more limited number of individuals. This simulation is performed prior to the implementation of behaviors and adjustment in physical robots.

MASCA enables the modeling of a range sensors, cameras, pheromone sensors, compasses, local and global location sensors, and several types of actuators. In our concrete case, we have worked with holonomic actuators for multirotor type vehicles. This simulator also makes it possible to use both local and global communication.

Using SCALA as a base language, we provide a functional interface to the system that enables the parallelization of a multitude of tasks in a secure manner. Due to speed issues, we use 2D physics for our simulations and assume that robots are able to maintain themselves constantly in a plane. This ability is not a problem for current micro or minidrones since barometric and ultrasonic sensors are available to measure their height relative to the ground, and the flight controller itself is responsible for stabilizing the height. Although the base physics is 2D, the simulator supports a 3D environment for flight and is able to develop routes at different heights without interfering with the collision calculations.

One of the advantages of our MASCA simulator is the ability to use any image as the basis for a navigation map, which allows us to easily design hundreds of maps for different test environments. In our specific case, we are interested in working with unstructured environments, so the first step is to generate these environments. We have developed a system based on cellular automatons that generate environments with these characteristics and that allows us to conduct our simulations in a great diversity of environments.

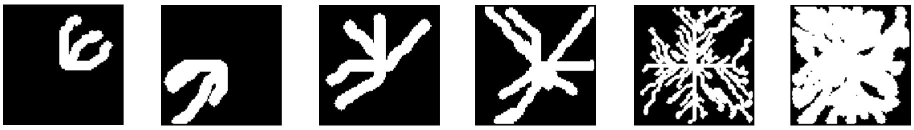
The operation is simple and is based on the idea, often used in video games, of how to generate “coherent” caves in a random way. The idea is to start from an initial random map (where each cell can be occupied or free) and iterate from the initial map via some simple rules, for example, a tile becomes a wall if it was empty and if more than ca_b of its immediate neighbors were walls, or becames empty if it was a wall and less than ca_a of its immediate neighbors were walls. Each iteration makes each tile more uniform, and the entropy of the environment is gradually reduced. The fundamental advantages of this type of generation are that practically all the levels generated are different. This is a very simple strategy to implement, and the results, as will be seen later, are realistic.

More specifically, the function used to generate all these environments is presented in Algorithm 1. Initially, function RandomMapWithEmptyCells generates a random map where *Pe* of the total cells will be empty. Once the random map has been generated, we apply the rule of the cellular automaton through the procedure ApplyCellularAutomataRule for each map cell. Furthermore, we must adapt the map with JoinCaves so that the different generated areas are connected to each other and a dilatation process with a Disk of size *D* to produce more uniform environments. Figure 1 presents a group of maps generated with this algorithm.

Algorithm 1 Unestructured Environment Generator

```
function GenerateEnvironment($size, Pe, D$)
    $M \leftarrow$ RandomMapWithEmptyCells ($size$, $Pe$)
    SetBorders ($M$)
    for each $cell$ in $M$ do
        ApplyCellularAutomataRule ($M, cell, 4, 5$)
    end for
    JoinCaves ($M$)
    $M \leftarrow$ Dilation ($M$, Disk($D$))
    SetBorders ($M$)
    return M
end function
procedure ApplyCellularAutomataRule($M, cell, a, b$)
    $S \leftarrow$ GetNeighborhoodSquare ($cell$)
    if AdjacentWall($S$)$< a$ then
        $M[cell] \leftarrow \emptyset$
    end if
    if AdjacentWall($S$)$> b$ then
        $M[cell] \leftarrow$ Wall
    end if
end procedure
```

Finally, once a map has been generated, it is used with our physical simulator to develop the simulation tasks required to thoroughly test our proposed behaviors. In Figure 2, an example of a map and its 3D visualization, together with the first steps of a deployment behavior, are presented.



(0.11,3) (0.11,4) (0.11,4) (0.15,3) (0.25,1) (0.25,3)

Figure 1. Example of map generation using Algorithm 1. The first parameter is the randomness index required by the automaton *Pe* (the higher the index is, the greater the generation of roads). The second parameter indicates the intensity of a morphological expansion applied to the map for smoothing *D*. The map size is 15 m × 15 m.

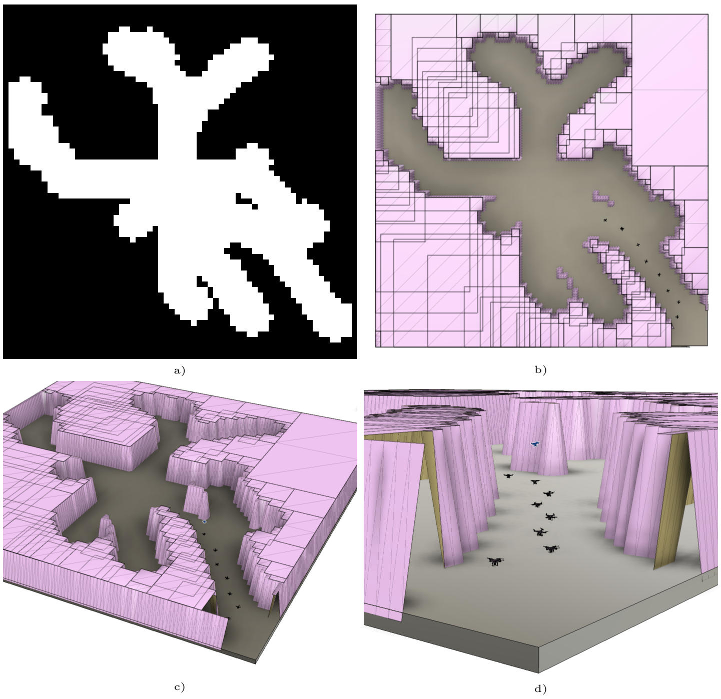


Figure 2. (**a**) Example of environment obtained by Algorithm 1, representing an area of 15 m × 15 m. (**b**) 3D visualization of this environment. Several bottom-right walls have been removed to highlight the access point to the environment. (**c**,**d**) Example of drone entry process and first steps of deployment behavior.

3.2. Slant Beacon Deployment (SLABE)

3.2.1. Behavior Description

Quantitative marker-based stigmergy is the most common choice in designing swarm intelligence systems. Its simplicity, scalability, robustness, and integration with the environment make it an ideal mechanism for communication in this type of system. However, the main difficulty is the lack of functional systems that allow it to be implemented in real environments.

Sematectonic stigmergy has the advantage in this sense because it does not require any external elements to be returned for communication: the same individuals may be deposited in the environment to communicate. In nature, we can see several examples, such as ant cemetery clustering and wasp nest construction.

Slant beacon deployment (SLABE) behavior aims to cover an unstructured and a priori complex environment. It assumes the existence of a sufficiently large set of simple individuals that can cover the environment without redeployment. As in [27], we propose the existence of two distinct states of energy consumption. In UAVs, the highest consumption is established in flight, so this energy consumption must be correctly modeled to increase the autonomy of these aircraft, which is a priori very limited. As mentioned above, an initial launch area from which all drones will take off and will be defined. The first drone will be placed in the immediate neighborhood of the launch area and will go into the *beacon* state (also called *repeater*). Then, a new drone will take off and travel, simply avoiding obstacles, through the interior of the area to be explored. As soon as it detects that it is near the limit of the communication safety zone, it will stop the flight and return to the *beacon* status. This process will be repeated until the environment is fully covered.

The role of *beacon* (*repeater*) drones is to recommend "the best address" they have observed so far to all active drones that pass by. To do this, they will ask the drones to pass through their immediate vicinity in the direction they were heading just before leaving their area of influence. This direction will be absolute and obtained from the compass of the vehicles. The drones that do not pass through other vehicle again will be assumed to have managed to reach remote areas in the direction they took as they passed through this particular *beacon*. However, the directions recommended by the drones that repeatedly pass near the *beacon* should not be considered with the same importance since they do not lead to the vehicle moving away from the drone's zone of influence. There will also be a mechanism for local communication between the *beacons* to prioritize promising paths, but this will be described later.

The active drones will navigate while considering several factors, including the obstacles in the environment and the proximity to other active vehicles with which they can coexist at any given moment.They will also take into account the recommendations made by *beacon* drones.

This behavior does not require local localization, which makes it especially suitable for implementation with microdrones. In addition, the stigmergy used does not require any sensors that are not currently available: we simply need to determine whether a vehicle is in a certain range of action. For example, the iBeacons protocol provides the necessary technical requirements to implement this behavior. This makes it possible to detect the location of two devices in a discreet way (far, near, immediate) with sufficient accuracy, in addition to obtaining the RSSI signal strength.

3.2.2. Deployment Strategies

The previous behavior has not detailed how the robots will be introduced into the environment. We know that the deployment process will be conducted at a single point from which the vehicles will be introduced progressively. However, there are several criteria for deciding when to introduce a new individual into the system.

A detailed analysis of various deployment methods is presented in [27]. In this case, three main types of strategies are analyzed. Initially, the robots can be introduced one by one in a fixed interval of time, which is known as linear-temporal incremental deployment (LTID). This strategy does not consider whether there are drones in the active state, so we may find that time is wasted in the unexplored state. The single incremental deployment (SID) strategy verifies that no robot is active in the system and then adds a new robot. However, in this case, we find that only one robot is active in the system, which can penalize the deployment process in large environments. The latest strategy presented in this paper, adaptive group size (AGS), adds new robots to the environment on the basis of the density of robots in the environment. To implement this strategy, a sensor is required to estimate the density.

In our work, we will use two types of strategy presented in Figure 3. On the one hand, we will test the behaviors with LTID deployment. This type of deployment has the advantage of reducing the probability of collision between individuals if the time between deployments is adjusted appropriately, or of prioritizing a parallel search for shorter intervals. This is a simple strategy to implement and requires no special hardware, making it applicable to any type of drone.

We do not consider the SID strategy since it does not allow more than one active robot at a time, preventing a parallel deployment of the swarm. The AGS deployment requires specific sensors for its implementation, so we propose a hybrid strategy of SID and AGS, which we call fixed group incremental deployment (FGID).

The FGID strategy allows the existence of multiple active drones in the system, as the SID strategy does, i.e., instead of a single UAV, several can coexist simultaneously. This allows us to substantially improve the time required for a SID strategy by working with groups of active robots, just as an AGS strategy would allow. However, it requires the choice of group size for a given environment, at the expense of not requiring any vehicle density sensors. In fact, given this strategy, it is easy to see that both LTID and SID are special cases of the same strategy: in SID, the maximum number of drones that can coexist is equal to one, and in LTID, it is equal to the number of drones available.

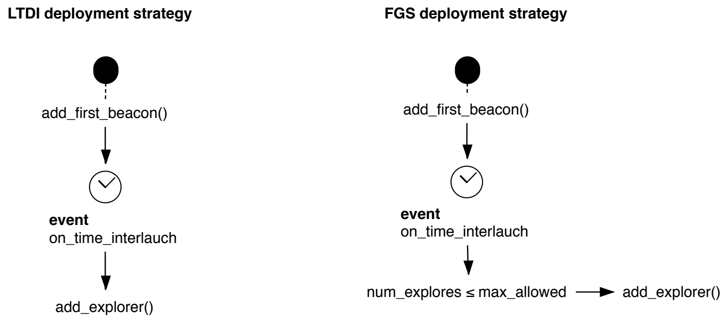


Figure 3. State diagram for each of the deployment strategies used in this article. Note that the linear-temporal incremental deployment (LTID) strategy is a special case of fixed group incremental deployment (FGID) when *max_allowed* equals the number of swarm agents.

3.2.3. Role Description

This section will detail the behavior of *slant beacon deployment* (SLABE) and the roles required to develop it. Initially, the role of *active* robots, which are responsible for the reconnaissance phase of the environment, will be described to review the role of *repeaters* or *beacon* robots, which are mainly responsible for providing the communication and stigmergy mechanisms required by the swarm.

Active

The main purpose of active robots is to explore the environment. Therefore, this behavior has a high energy cost as it requires robots to fly over the environment to develop the deployment behavior. The behavior of these individuals is defined in Figure 4.

Initially, the active robots (which we will also call explorers) will proceed to determine, through their communication beacons, which drones are close to them (i.e., which drones are within their range of communication). As previously mentioned, the support technology can be Bluetooth Low Energy. Once all the nearby robots are identified, the nearest neighbor will be calculated (for example, using the RSSI signal strength, or for more accuracy, the BLE location protocol). To implement this behavior, we require that our communication system allows the determination of at least three ranges of distance or signal reliabilities: maximum efficiency in communication (near), stable communication (medium), and communication limit (far).

If the vehicle is within a close range (*in_range* in Figure 4), then it will proceed to include the angle recommended by that neighbor *beacon* to the path to be performed by the robot. However, the vehicle may not detect a nearest neighbor within the limit of communication or, for any particular failure, it may not detect any nearby neighbor. In either case, the robot will change roles and become a *repeater*. Before reaching the limit communication margin, a communication distance is output (*exit_range* is established in the diagram). If the nearest neighbor is close to that distance, the *active* robot communicates its current angle (obtained by its onboard compass), which will be stored in the neighbor *repeater* robot. As we will see later, the purpose of communicating this angle is to determine which angles of exit in this area of the environment make it possible to achieve better deployments. A robot may also become a repeater if it is at a certain battery threshold, in which case, it does not have to fulfill the above conditions.

The movement of the *active* robot will be determined mainly by its onboard collision sensors (for example, using visual navigation or range sensors) and by the angle recommendations of its nearest neighbor. If deployment strategies that allow for the coexistence of several *active* robots are tested, the robots can have collision-avoidance mechanisms, which should also be taken into account when planning their movement.

$$\tilde{\mathbf{v}} = \mathbf{vangle}(\mathrm{draw}(P(\Lambda^{\hat{r}}_{B_t})))) \tag{1}$$

$$\mathbf{f}^{r}_{t+1} = -\alpha_1 \sum_{I=1}^{|O^{r}_{t}|} (o_i - \mathbf{pos}(r_t)) + \alpha_2 \cdot \tilde{\mathbf{v}}\Lambda^{\hat{r}}_{h_t} + \alpha_3 \cdot \mathbf{f}^{r}_{t}\mathcal{N}(\mu, \sigma) \tag{2}$$

Equation (1) shows how the velocity vector $\mathbf{f}^{r}_{t+1}$ of a vehicle r is established by executing the *SLABE* behavior at instant $t+1$. The position of the robot $\mathbf{pos}(r_t)$ and the position of the obstacles detected by the drone are taken into account, where each detected position i is a coordinate vector $\mathbf{o}_i$ that is stored in a set of obstacles detected by robot r at time t ($\mathbf{o}_i \in O^{r}_{t}$). In addition, the angle recommended by the *beacon* closest to the robot, which we will call $\Lambda^{\hat{r}}$, and its intensity should also be obtained. $P(\Lambda^{\hat{r}}_{B_t})$ is a probability distribution that stores the most promising angles in the area of influence of a *beacon*. $\Lambda^{\hat{r}}_{h_t} \in [0, 1]$ represents the recommendation intensity of the drone. In this way, the rotation is included in the drone's movement function by extracting a random angle following this distribution ($\mathbf{draw}(P(\Lambda^{\hat{r}}_{B_t}))$) and generating a unit vector $\tilde{\mathbf{v}}$ with this angle by means of the function $\mathbf{vangle}$. This unit vector is rectified by the recommendation intensity of *beacon* $\Lambda^{\hat{r}}_{h_t}$.

Speed in a previous instant is taken into account to produce fluid behavior. This velocity will be slightly altered by a random term extracted from a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ to include a noise term that accounts for possible modeling errors in the system. We will assume that the robot is stopped at the initial instant $\|\mathbf{f}^{r}_{0}\| = 0$. The normalizing terms α_i allows the performance of the behavior to be adjusted based on the environment and the available sensors and actuators. For this article, we will assume constant values of these parameters.

Repeater

The purpose of the vehicles *beacon* or *repeaters* is multiple. On the one hand, given that the behavior guarantees that there will be at least one communication channel between two vehicles, they are in charge of providing a communication platform between the drones. This platform could be useful, for example, in case we want to manually inspect a specific area of the environment where the drones have had access. It will also be used, when calculating the angular recommendation intensity of each *beacon*, as we will see later. The behavior of these individuals is defined in Figure 5.

When a robot performs the role of *repeater* or *beacon*, it must move to a state of low consumption and therefore stop its engines, setting its speed to 0. As discussed in [23], in the case of a linkage system, it is preferable for the system to be located in areas as high as possible to maximize the perception and coverage. The robot can land in the nearby area that best achieves these effects.

After stopped the robot becomes part of the swarm, the communication packages that it receives must be communicated to the rest of the drones with which it communicates.This resending process is conducted automatically by XBee devices using the appropriate protocol. However, the main goal was to provide a distributed pheromone system.

Therefore, we wish to develop this behavior with simple drones (microdrones) without requiring specific sensors for the communication of pheromones: we propose the use of *repeater* robots as sematectonic deposits. In this way, each *active* drone that leaves the action range of the *beacon* will recommend its exit angle to the *repeater*. The *repeater* will store the angles recommended by each drone and will also take into account how many times the drones visit it.

The number of visits will be used by the *repeater* to calculate how good the vehicle's recommendation is. If a vehicle is displayed only once, this probably means that the drone was able to move away from the repeating area in the direction it took. However, if the robot reappears many times, it is likely that its starting angle will not be appropriate.

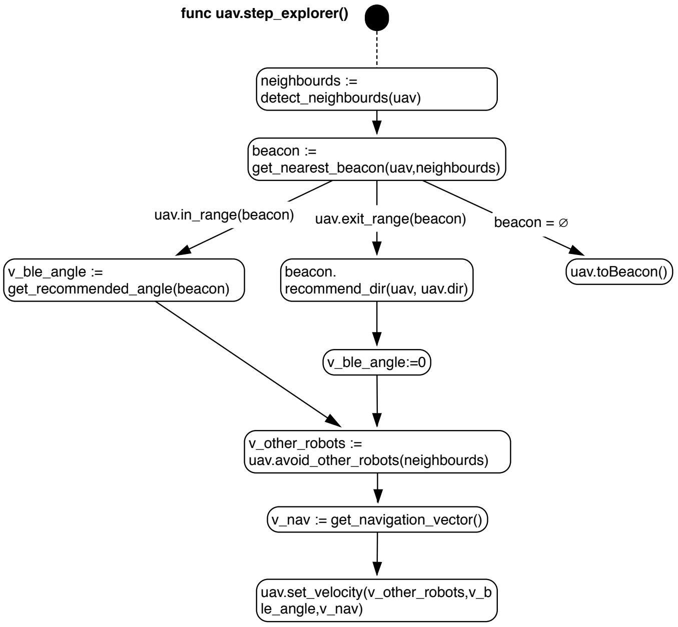

func uav.step_explorer()
neighbourds := detect_neighbourds(uav)
beacon := get_nearest_beacon(uav,neighbourds)
uav.in_range(beacon)
uav.exit_range(beacon)
beacon = ∅
v_ble_angle := get_recommended_angle(beacon)
beacon. recommend_dir(uav, uav.dir)
uav.toBeacon()
v_ble_angle:=0
v_other_robots := uav.avoid_other_robots(neighbourds)
v_nav := get_navigation_vector()
uav.set_velocity(v_other_robots,v_b le_angle,v_nav)

Figure 4. State diagram for the slant beacon deployment (SLABE) behavior of individuals of type *active*. For each processing step, any individual in this role will execute the steps listed in the diagram.

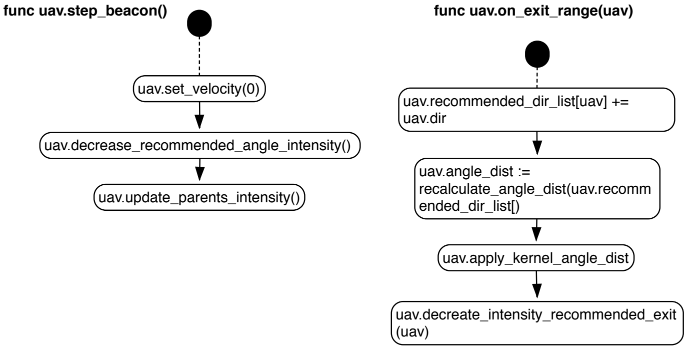

func uav.step_beacon()
uav.set_velocity(0)
uav.decrease_recommended_angle_intensity()
uav.update_parents_intensity()
func uav.on_exit_range(uav)
uav.recommended_dir_list[uav] += uav.dir
uav.angle_dist := recalculate_angle_dist(uav.recomm ended_dir_list[)
uav.apply_kernel_angle_dist
uav.decreate_intensity_recommended_exit (uav)

Figure 5. State diagram for the SLABE behavior of individuals of type *repeater*, also called *beacon*. For each processing step, any individual in this role will execute the steps listed by the diagram.

Specifically, each vehicle b in the *beacon* state will use a probability distribution $P(\Lambda^b_{B_t})$ at time t. This distribution will store the information about which angles are most promising for the entire swarm. In addition, the recommendation intensity $\Lambda^b_{h_t}$ will determine the weight with which it takes into account the recommendations it makes.

The distribution $P(\Lambda^b_{B_t})$ will be updated at each instant with the information from the environment and will be filtered through the convolution filter presented in Equation (5), which will enable a new distribution to be obtained for the instant $t+1$. The previous filtering process makes it possible to consider not only the discrete angles through which other vehicles passed but also the angles close to their area of influence. More specifically, the distribution $P(\Lambda^b_{B_t})$ will be implemented discretely using a histogram of 360 values (one value for each degree). In Figure 6, the discrete kernel $P(\Lambda_K)$ used for the convolution process is shown.

The process of calculating the distribution for each instant t is presented in Equaiton (3), where $\mathbf{\Lambda}^b_{\Gamma^r}$ is the set of all the angles recommended to *beacon* b for a drone r, and β_1 is a distribution normalization term. The variable a_i represents the angle recommended to the *beacon* by robot r, which is calculated from the sum of all the angles recommended by that robot to the *beacon*, where $\Lambda^b_{\Gamma^r_k}$ is each of the angles.

$$a_i = \sum_{k=0}^{|\mathbf{\Lambda}^b_{\Gamma^r}|} (\Lambda^b_{\Gamma^r_k}) \cdot 1/|\mathbf{\Lambda}^b_{\Gamma^r}| \tag{3}$$

$$P(\Lambda^b_{B_t} = a_i) = \frac{\sqrt{1/|\mathbf{\Lambda}^b_{\Gamma^r}|}}{\beta_1} \tag{4}$$

$$P(\Lambda^b_{B_{t+1}} = z) = \sum_{m=-\infty}^{\infty} P(\Lambda^b_{B_t} = m) P(\Lambda_K = z - m) \tag{5}$$

Note that when an *active* drone is about to leave the area of a *beacon* drone, it will communicate its angle, and the intensity of its recommended angle will decrease $\Lambda^b_{h_t}$ proportionally to a value $\beta_2 \in [0,1]$ so that each *beacon* will gradually reduce its influence over time.

Finally, the *beacon* drones will form a structure of parents/children so that when a drone transitions from *active* to *beacon*, it will consider the last *beacon* node that it visited (which, according to the behavior definition, should be in contact with it). As mentioned above, the recommendation intensity of a drone $\Lambda^b_{h_t}$ decreases with each visit by a factor of β_2. However, there is a way to increase the intensity of the influence. For each instant of time t, the recommendation intensity of a drone is calculated as presented in Equation (6). In this way, the most promising roads, where a priori more *beacon* drones are located, are enhanced. Finally, as time passes, the intensity of the pheromones will decrease by a factor of β_3.

$$\Lambda^b_{h_{t+1}} = \max(\Lambda^b_{h_t} \cdot \beta_2, \text{parent}(\Lambda^b)_{h_t}) \cdot \beta_3 \tag{6}$$

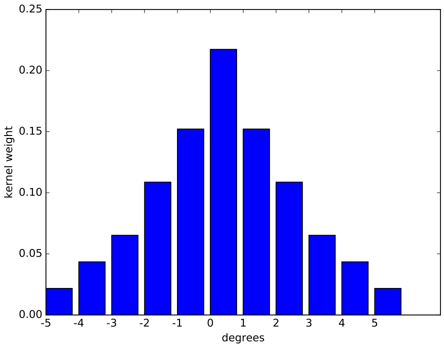


Figure 6. Discrete kernel $P(\Lambda_K)$ used for the convolution process of the distribution of *beacon* robots.

3.2.4. Example of Operation

As noted above, several parameters must be established for the performance of the presented behavior. Most of the parameters can be established in a simple way based on the environment or the type of sensors and actuators available on the robots. In all the experiments that we will perform, we will use the fixed parameters that we present next, which are chosen for their versatility and good performance in various environments. It is important to emphasize that these parameters will not be altered, even for different types of maps (simple or complex) on which the behavior will be tested.

The tuple of behaviors that define the $SLABE_a$ behavior used for all the experiments in this article is defined in Equation (7).

$$SLABE_a = \begin{cases} \alpha_1 = 0.26 \\ \alpha_2 = 0.52 \\ \alpha_3 = 0.21 \\ \Lambda_{h_0} = 1 \\ \mu = 1 \\ \sigma = 0.25 \\ \beta_1 : \text{obtained for each } t \text{ from } P(\Lambda^b_{B_t}) \\ \beta_2 = 0.7 \\ \beta_3 = 0.999 \\ P(\Lambda_K) \\ \|\mathbf{f}_0\| = 0 \end{cases} \tag{7}$$

It should be mentioned that the normalizing term β_1 will be calculated for each instant of time t, in a simple way from the current data of the distribution $P(\Lambda^b_{B_t})$ for each beacon b. As discussed earlier, the kernel $P(\Lambda_K)$ is discretely specified in Figure 6.

$$Robot_{SLABE} = \begin{cases} RF_{rmax} = 6 \text{ m} = \text{uav.exit\_range} \\ RF_{ibeacon} = 3 \text{ m} = \text{uav.in\_range} \\ Obs_{rmax} = 2.5 \text{ m} \\ vel_{max} = 1 \text{ m/s} \\ r_{coverage} = 10 \text{ m} \end{cases} \tag{8}$$

A robot that implements SLABE behavior must also define its sensory and motor characteristics, as established in Equation (8). The maximum coverage range RF_{rmax} to guarantee our radio frequency system is set to 6 m, and a robot is considered to be within the margin of influence of a beacon $RF_{ibeacon}$ if the robot is within a 3 m radius of the beacon's position. The detection range of the obstacle detection system of the robot Obs_{rmax}, its maximum velocity vel_{max}, and the area that is considered to be covered by its position $r_{coverage}$ are detailed in Figure 6.

Finally, we will use an energy model based on [27], where a common energy consumption for both types of drones will be used to facilitate the comparisons. A consumption of 120 W will be assumed for a drone in the *active* state, and 1 W will be assumed for a drone in the *beacon* state. It should be noted that microdrones with an appropriate configuration can achieve a much lower consumption (mainly due to their reduced weight and therefore lower energy requirements for takeoff).

An example of how this behavior works for a simple-type map is provided in Figure 7. Various states have been selected, from the initial step to 95% deployment of the map area. Each robot has a viewing area that is used to calculate the percentage coverage of the map (area represented in blue). This figure shows how the behavior and intensity of the recommendation of each *beacon* robot evolve (the higher the yellow tone of the drone is, the greater intensity of the recommendation). The communication area RF_{rmax} of the drone can vary from white to yellow to represent this intensity. Finally, the recommended angles for each *beacon* robot are also displayed.

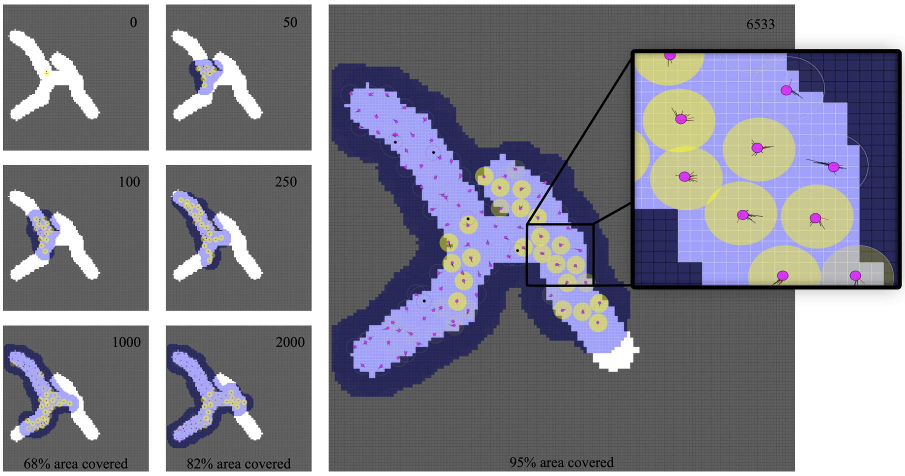


Figure 7. Behavioral trace for different simulation steps. Each simulation step is equivalent to 1/2 s. An enlarged area of the simulation is shown when the target case (95% coverage of the map area) is reached. The figure shows the robots (in red), their communication area RF_{rmax} (yellow), the angles recommended by the *beacon* robots (black concentric lines around the robot) and their visual or coverage area (blue).

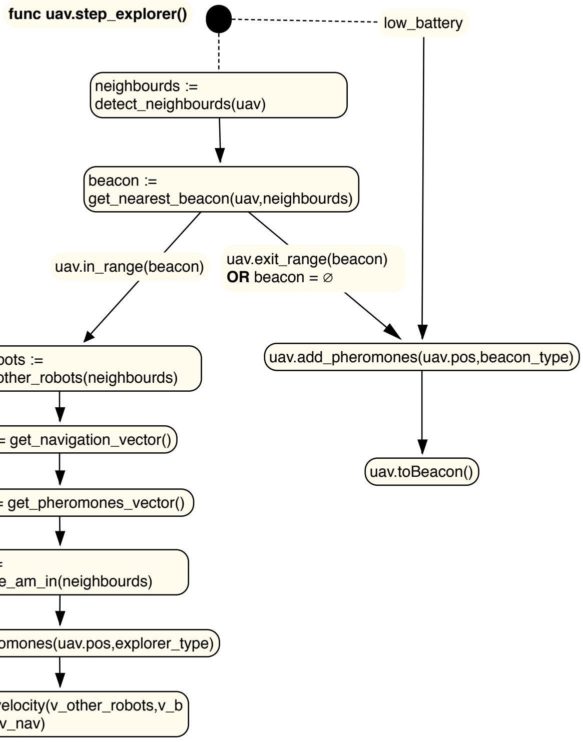

func uav.step_explorer()
low_battery
neighbourds :=
detect_neighbourds(uav)
beacon :=
get_nearest_beacon(uav,neighbourds)
uav.in_range(beacon)
uav.exit_range(beacon)
OR beacon = ∅
v_other_robots :=
uav.avoid_other_robots(neighbourds)
uav.add_pheromones(uav.pos,beacon_type)
v_nav := get_navigation_vector()
uav.toBeacon()
v_phero := get_pheromones_vector()
uav.pos =
uav.where_am_in(neighbourds)
uav.add_pheromones(uav.pos,explorer_type)
uav.set_velocity(v_other_robots,v_b
le_angle,v_nav)

Figure 8. State diagram for the Sematectonic Pheromone Deployment (SEPHE) behavior of individuals of type *active*. In each processing step, any individual in this role will execute the steps listed in the diagram.

3.3. Sematectonic Pheromone Deployment (SEPHE)

3.3.1. Behavior Description

As observed in the previous section, the slant beacon deployment behavior requires not special capacity for implementation. Any current UAV can fulfill all the hardware requirements, including communication. Therefore, it is appropriate behavior for implementation in large swarms, where individuals are simple. This type of behavior is therefore especially suitable for a swarm of microdrones, such as those presented above. If we have individuals with more sensory and processing capabilities, we can take greater advantage of the benefits of the use of stigmergy for swarm communication and the emergence of useful behaviors.

In this section, we will present a behavior called sematectonic pheromone deployment (SEPHE) that has the same purpose as that of SLABE, that is, to cover an unstructured and a priori complex environment. For this behavior, the drones will be required to be able to perform local localization among themselves. In [3], the authors present a sensor that can be used to effectively perform local localization among agents. Like SLABE, there will be two differentiated states. *Active* agents will take off and navigate via an obstacle-avoidance algorithm until the signal from the farthest *beacon* vehicle is within the safety limit. Then, the *active* vehicle will transition to the *beacon* state, as in the previous case.

The fundamental difference with SLABE behavior is that while an *active* drone navigates, it will emit pheromones that will be collected by the *beacon* vehicles through which it travels. The *beacon* drones will therefore serve as the local location mechanism of the swarm as well as stores of local pheromone maps. These local maps will be communicated to any drone within range.

The repeater drones will therefore be responsible for the communication, both direct and indirect, of the swarm. They must also maintain their virtual pheromone map, communicate it to their nearby robots, and adjust it, if necessary. The repeater drones should also apply the intensity decrement to the map, so that, among other tasks, areas that have not been explored for a long time can be re-verified.

This behavior is therefore a variation of the use of quantitative sematectonic stigmergy, which requires more communication than that required by the SLABE behavior but which is more similar to the standard implementations of community-based pheromone robotic systems, in addition to providing much more information to the active robots for making decisions in flight. The following describes the behavior in detail.

3.3.2. Role Description

Active

Active drones are drones that are in flight exploring the environment. The behavior of these individuals is defined in Figure 8. Like the SLABE behavior, active drones determine their movement based on several factors. On the one hand, they determine their position with respect to the repeater vehicles, but in this case, the positioning is much more accurate (not only do they know that they are in rank with a repeater, but they know their position relative to it). On the other hand, they also take into account, in the event that two or more active robots coexist, their relative location, as long as they are within the detection range. We will assume that they also use an onboard collision sensor that will allow them to obtain a repulsion vector for areas that can generate collisions.

The main novelty with respect to SLABE is that a pheromone map sent by the nearest *repeater* is now available, which will enable them to make more informed decisions. A pheromone type of variable intensity that will lose intensity over time will be used. While *active* robots navigate through the environment, they will leave a path of pheromones, which will be stored in the *repeater* robots. The navigation process will continue until either the robot does not have sufficient battery power to continue developing its active task or it is in the limit communication range (*exit_range*). The pheromone emission of an *active* robot is characterized by its intensity Ψ_{ha} and its radius Ψ_{ra}.

Specifically, the motion of an active robot will be determined mainly by its onboard collision sensors and the local pheromone map. If deployment strategies that allow the coexistence of several active robots are tested, the local location of those robots within a certain detection range will also be used as planning information. All these parameters will be combined to obtain the robot speed.

$$C = \left\{ (x, y) | \sqrt{x^2 + y^2} \leq \Psi_{ra} \right\} \tag{9}$$

$$\mathbf{f}^r_{t+1} = \gamma_1 \sum_{I=1}^{|O^r_t|} (o_i - \text{pos}(r_t)) + \gamma_2 \cdot \sum_{c \in C} \left(\frac{\langle y, x \rangle}{\| \langle y, x \rangle \|^2} \cdot \Psi_{field_{y,x}} \right) + \gamma_3 \cdot \mathbf{f}^r_t \mathcal{N}(\mu, \sigma) \tag{10}$$

where C is the set of local Cartesian coordinates (x, y) that are within the detection radius of the pheromone sensor Ψ_{rb}. O^r_t is the set of obstacles detected by robot r at time t, $\langle y, x \rangle$ is a vector formed by the coordinates y, x normalized by its squared norm. $\Psi_{field_{y,x}}$ is a matrix that stores the reading of the corresponding pheromone sensor in position y, x. The speed of the drone $\mathbf{f}^r_{t+1}$ at instant $t + 1$, which is defined by the sum of terms in the previous equation, is adjusted by $\gamma_i \in [0, 1]$, whose overall sum is one.

Repeater

As in the SLABE behavior, the repeaters are responsible for maintaining the communication channel. The behavior of these individuals is defined in Figure 9. In this case, this channel is of special importance since we are not considering sporadic communications with very little bandwidth (as in the previous case) but a channel that is used to update the pheromone map and the local location of the swarm individuals. Repeaters will be responsible for forwarding the required packages to all drones in their range of action.

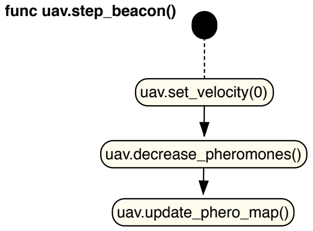


Figure 9. State diagram for the SEPHE behavior of *beacon* individuals. In each processing step, any individual in this role will execute the steps listed in the diagram.

As soon as a drone transitions from the *active* state to the *repeater* state, it will emit a pheromone halo of Ψ_{hb} intensity and radius Ψ_{rb}. The repeater robot will enter a state of low power consumption and should therefore be located in an area of the environment that maximizes its perception and coverage, if possible.

In this behavior, repeaters must provide local positioning mechanisms and must have sufficient memory to store their local pheromone map. This map will be constantly updated, either by the navigation process of the *active* drones or by the process of the constant reduction in pheromones intensity as time passes. Specifically, each second, the intensity will decrease by a factor of Ψ_{decay}.

3.3.3. Example of Operation

As noted above, several parameters must be established to achieve the presented behavior. Most of the parameters depend mainly on the type of sensors and actuators available on the robots. In all the experiments that we will conduct, we will use the following fixed parameters. It is important to emphasize that these parameters will not be altered, even for different types of maps (simple or complex) on which the behavior will be tested. The tuple of behaviors that define the behavior $SEPHE_a$ that will be used for all experiments in this article is defined in Equation (11).

$$SEPHE_a = \begin{cases} \gamma_1 = 0.57 \\ \gamma_2 = 0.28 \\ \gamma_3 = 0.14 \\ \mu = 4 \\ \sigma = 1 \\ \Psi_{ha} = 0.09 \\ \Psi_{hb} = 0.1 \\ \Psi_{ra} = 5\text{ m} \\ \Psi_{rb} = 5\text{ m} \\ \Psi_{error} = 5\text{ m} \\ \Psi_{decay} = 0.99 \\ \|\mathbf{f}_0\| = 0 \end{cases} \tag{11}$$

In addition, in our simulator, to obtain a more realistic approach with respect to local location systems, the pheromone layout events have a positioning error of $\Psi_{error} = 5$ m around the actual disposition point.

A robot that implements SEPHE behavior must also define its sensory and motor characteristics, as established in Equation (12). The parameters are consistent with the previous behavior, and the same energy model will be used (120 W consumption per drone in the *active* state and 1 W consumption per drone in the *beacon* state, so we will assume that the same drones will execute both behaviors; although, as previously shown, SLABE behavior has lower hardware requirements).

$$Robot_{SEPHE} = \begin{cases} RF_{rmax} = 5\text{ m} = \text{uav.exit\_range} \\ RF_{ibeacon} = 3\text{ m} = \text{uav.in\_range} \\ Obs_{rmax} = 2.5\text{ m} \\ vel_{max} = 1\text{ m/s} \\ r_{coverage} = 10\text{ m} \end{cases} \tag{12}$$

In Figure 10, we present an example of how this behavior works for a simple-type map. Various states have been selected, from the initial step to 95% coverage of the map area. The distributed pheromone map maintained by *beacon* robots is shown in blue. Along the route, the most promising roads are prioritized to achieve greater coverage.

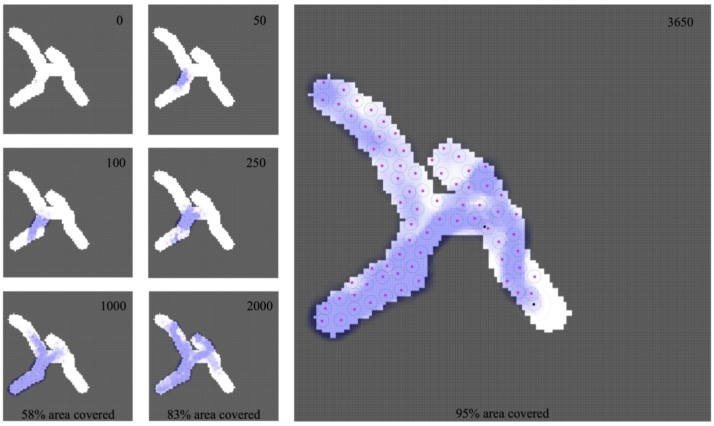


Figure 10. Behavioral trace for different simulation steps. Each simulation step is equivalent to 1/2 s. An enlarged area of the simulation is shown when the target case (95% coverage of the map area) is reached. The figure shows robots (in red), their detection area (concentric gray line with the robot), and the pheromone map maintained by the *beacon* robots (blue).

4. Results

4.1. Purpose and Design of Experiments

In this section, several tests will be conducted to verify the behaviors presented in the previous section. Several experiments will be performed to analyze the number of collisions (if a collision occurs, even if it is recoverable, it is counted for each of the agents involved), the time required to cover 95% of the area to be explored, the energy consumed and the total flight time of the swarm.

We will use two main types of maps to compare, in a general way, the deployment strategy used as well as the functioning of unstructured behaviors. We will add a type of map that is more oriented to the comparison of the behaviors in complex environments of very difficult access.

For each test, a random map will be generated with the characteristics listed below using cellular automata (as briefly described in the discussion of the simulator). In this way, we will check the robustness of the behaviors in unstructured environments. The free points from which vehicles could be introduced into the system will also be calculated. In this way, random behavior start points will also be used.

The first type of maps will be called *easy* and will be maps with a walkable zone of approximately 75 m^2. Easy maps will be unstructured maps with few ramifications and obstacles. The second type will be called *hard*. Hard maps will be much more extensive maps with an approximately 200 m^2 walkable zone. These maps will also have much higher entropy, with obstacles that will make it difficult to navigate and with a multitude of paths and variants that will make it difficult to perform the desired behavior. To verify exhaustively the behaviors presented in complex environments, we will perform some tests in *hard* environments with very difficult access, where there will be a central wall on the map that will divide it into two parts with a single access point of 10 m. To completely cover these maps, at least half of the swarm must pass through the door. These maps will be called *wall*. Figure 11 shows an example of each of type of map.

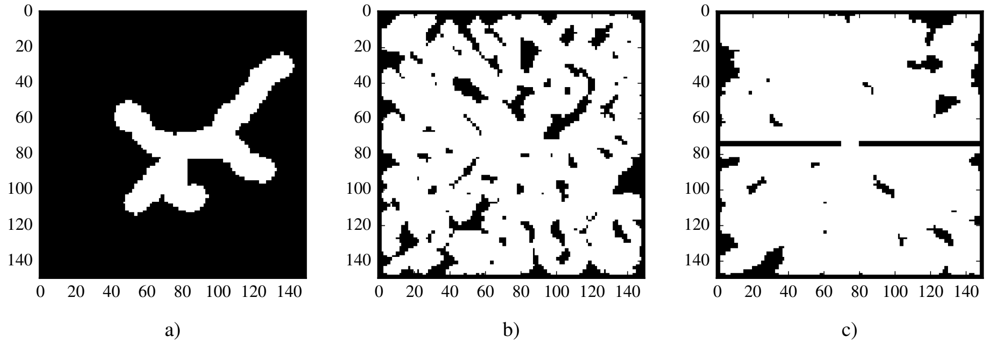


Figure 11. Example of maps types. A 15 m × 15 m area is shown. (**a**) *Easy*, with approximately 75 m^2 of walkable space; (**b**) *hard*, with approximately 200 m^2 of walkable space and with a greater entropy; (**c**) *wall*, the same as *hard*, but with a wall that divides the map into two zones.

In this section, we want to provide a set of tests that allow us to verify several aspects. On the one hand, the take-off time interval is important in terms of the number of collisions and the total energy required by the swarm. We are interested in validating two strategies (LTID and FGID), taking into account a new parameter, the maximum number of active vehicles that may be navigating at a specific moment. In addition, we want to validate the importance of the take-off time interval is maintained in all types of maps.

We also want to test the energy efficiency of our two behavior patterns. The implementation requirements are the same, so we assume that there will be variation in both with respect to the energy required, deployment times and number of collisions. It is also important to verify how different types of maps affect the behavior. We will verify the performance of both types of behavior with all three types of maps.

To test the LTID deployment strategy, 50 tests will be developed for different take-off time intervals (6, 8, 10, 12, 18, and 24 s). Each test will have a randomly generated map of the type to be tested, as well as a different starting point. By not considering redeployment, we will assume that we have a sufficient number of individuals to completely cover the environment (given their area of perception, in our case, set to $r_{coverage}$). In these tests, we will assume an upper limit of 100 individuals, which allows us to cover a maximum area of 1000 m^2.

In the case of the FGID deployment strategy, each of the previous take-off time intervals will be tested for a maximum number of 2, 5, 10, and 20 individuals in flight. To reduce the influence of the random generation of maps, because an increase in the number of tests per map is not feasible, we will assume a single map of each type and a common starting point. In this way, for each behavior and each type of map, 1200 simulations will be conducted (50 simulations on the same map for each combination of take-off time interval and maximum number of vehicles in flight).

4.2. LTID Tests

Figure 12 presents the results of 600 simulations performed for SLABE and SEPHE using the LTID strategy for different take-off time intervals on *easy* maps. For each take-off value, 50 random maps and random take-off positions are generated. For both behavior types, the figure shows the average and standard deviation of the number of collisions, the time required to cover the 95%of the environment, the energy consumed and the total flight time required by the active members of the swarm.

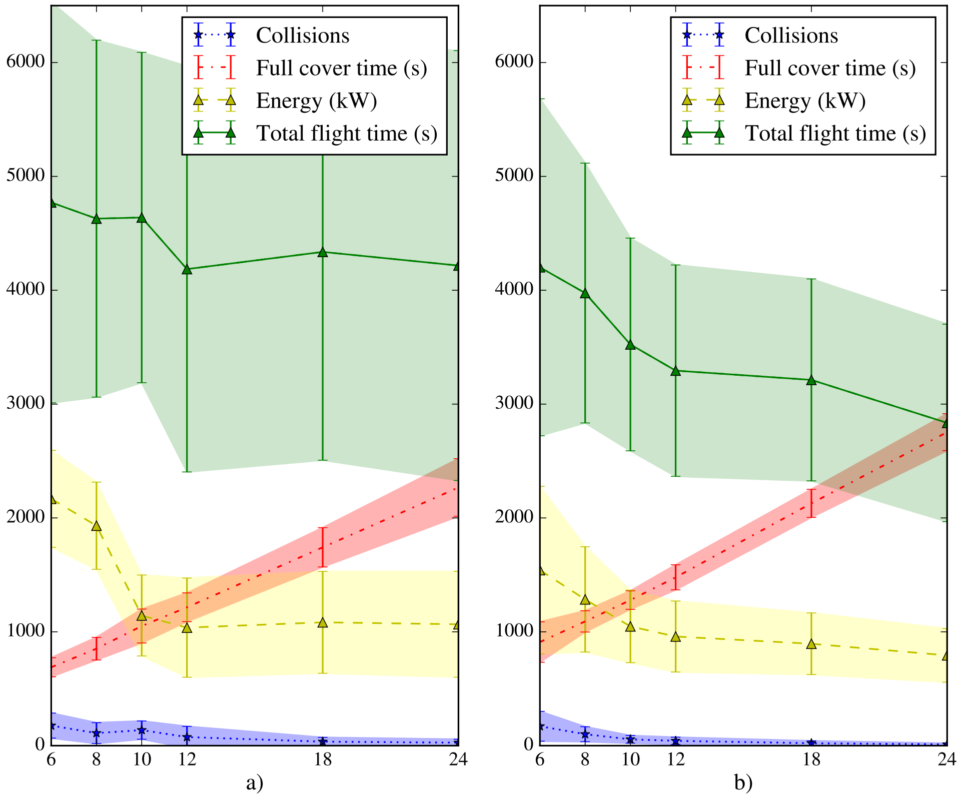


Figure 12. Results of the SLABE (**a**) and SEPHE (**b**) behavior simulations performed on 50 random *easy* maps. In each simulation, a random start position has been chosen among all feasible locations. Six different take-off time intervals have been tested for each map (6, 8, 10, 12, 18, and 24 s). The average and standard deviation of the number of collisions, the time required to cover 95% of the environment, the energy consumed by the swarm, and the total flight time are shown.

At a general level, it is easy to see how the minimum time required to reach the desired coverage (approximately 16 min) is achieved with a take-off time interval of 6 s (the shortest used). However, the shorter interval also increases the number of collisions between vehicles. The energy required by the swarm starts at 2000 kW for a take-off time interval of 6 s and decreases to 1000 kW as the time interval increases. This decrease is significant for the initial interval and becomes much less pronounced for an interval of 12 s. The same is not true for the deployment time, which increases continuously from the initial 6 s take-off interval (and 16 min deployment) to the 24 s take-off interval (and 30 min deployment) for the SLABE behavior.

The general tendencies are the same for both types of behavior: less energy is consumed and fewer collisions occur as the take-off time interval increases. There is much greater variability in the total flight time required and in the energy consumed for the SLABE strategy. Figure 13 presents the results of 600 simulations for both types of behavior (SLABE and SEPHE) using the LTID strategy for different take-off time intervals on *hard* maps. The parameters used in the tests are the same as in the previous case.

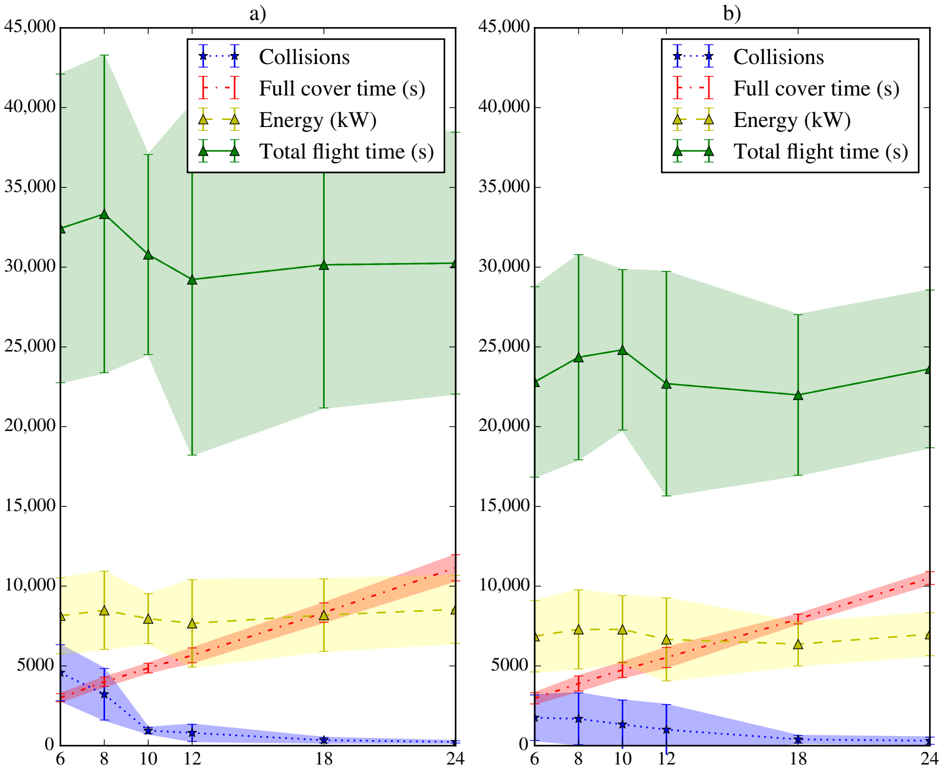


Figure 13. Results of the SLABE (**a**) and SEPHE (**b**) behavior simulations performed on 50 random *hard* maps. In each simulation, a random start position has been chosen among all feasible locations. Six different take-off time intervals (6, 8, 10, 12, 18, and 24 s) have been tested for each map. The average and standard deviation of the number of collisions, the time required to achieve 95% coverage, the energy consumed by the swarm and the total flight time are shown.

The time required for deployment is much longer than that of the previous case (a minimum of approximately 40 min), which is a consequence of the size of the map and the greater difficulty of navigation. The general tendencies observed in the previous case are maintained: the shorter the take-off time interval, the greater the number of collisions and the shorter the time require to achieve the desired coverage. However, the decrease in energy is not as substantial as that in the previous case. Energy consumptions remains almost constant, even for smaller take-off time intervals. The coverage times of the two approaches are similar, although there is still much more variability in the flight time required by the SLABE strategy.

Although the behaviors have been executed on the same type of agents, the SLABE behavior has much lower hardware requirements for its implementation; therefore, the agents that support it do not require higher energy consumption. Figure 14 presents a simulation of the SLABE and SEPHE behaviors equivalent to the previous one but with a different energy model for the SLABE agents. In this case, the agents executing this behavior require 15 W in the active state (equivalent to the consumption of vehicles of the ARDrone type) and 0.5 W in the beacon state. The vehicles that execute the SEPHE behavior follow general energy model based on [27]. With this assumption, the swarm SLABE behavior requires less energy, even for a longer flight time.

Finally, the performance of the behaviors for *wall*-type maps has been checked. Figure 15 presents the results of 600 simulations for the both types of behavior (SLABE and SEPHE) using the LTID strategy for different take-off time intervals.

In this case, the time required is much longer (a deployment time or approximately 4 h) due to the difficulty of the environment. The energy and time required, as well as the number of collisions, are substantially increased compared to those of the previous tests. In this extreme case, there is an increase in the variability of the SLABE approach due mainly to the low granularity of the local location system.

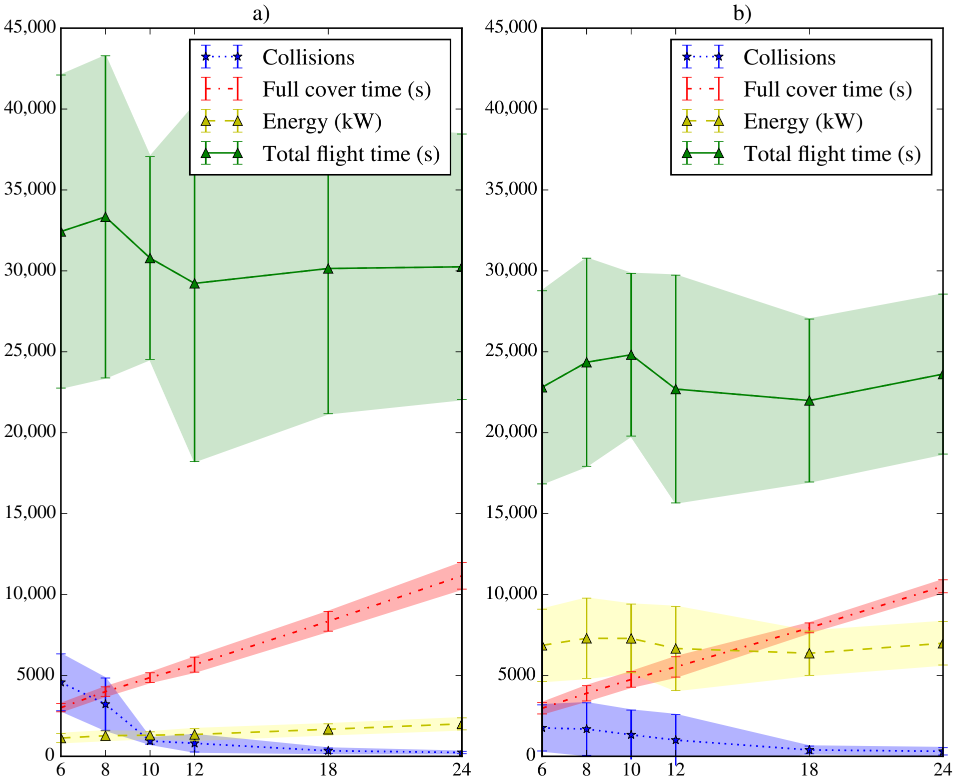


Figure 14. Results of the SLABE (**a**) and SEPHE (**b**) behavior simulations, equivalent to those presented in Figure 13, with an energy model for SLABE agents with lower consumption, where active agents consume 12 W and repeater agents consume 0.5 W.

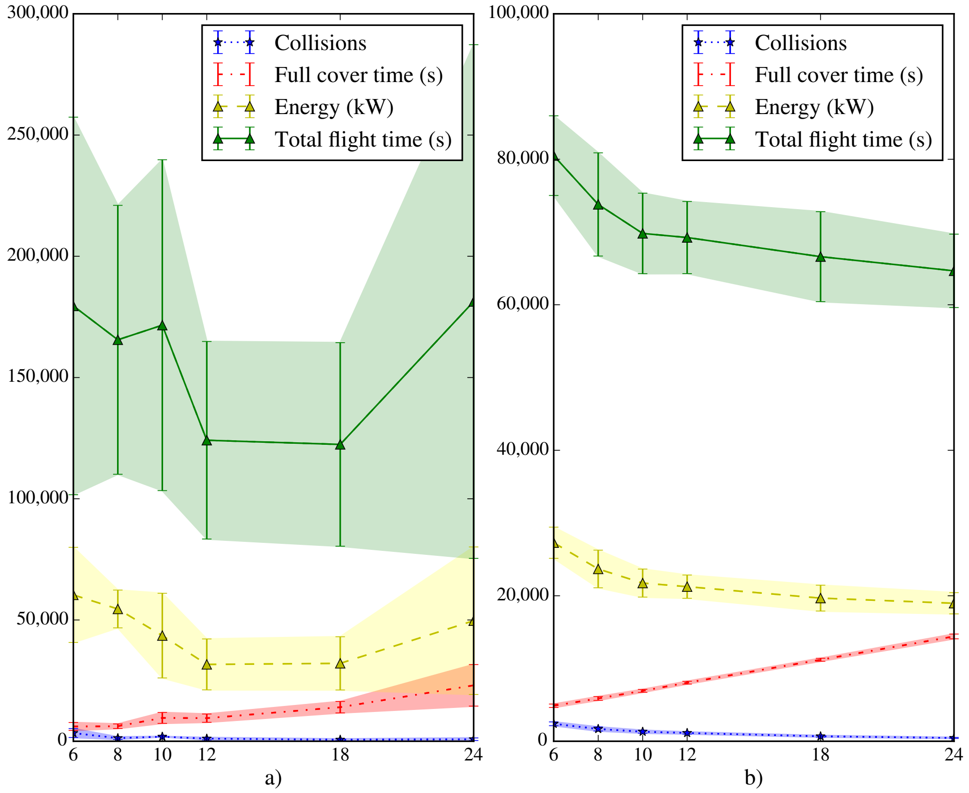


Figure 15. Results of the SLABE (**a**) and SEPHE (**b**) behavior simulations performed on 50 random *wall* maps. In each simulation, a random start position has been chosen among all feasible locations. Six different take-off time intervals (6, 8, 10, 12, 18, and 24 s) have been tested for each map. The average and standard deviation of the number of collisions, the time required to achieve 95% coverage, the energy consumed by the swarm, and total flight time are shown.

4.3. FGID Tests

Figures 16 and 17 show the tests performed for the SLABE and SEPHE behaviors, respectively, using the FGID deployment strategy for different take-off time intervals and maximum number of vehicles on *easy* maps.

In this case, we observe a great variability in the energy consumption and total flight time of the swarm with a reduced maximum number of vehicles, which is important for SLABE behavior. This variability is substantially reduced as the number of vehicles used increases.The SLABE behavior is more constant with respect to energy consumption, which is lower and has less variability in the case of SLABE, even with a limited number of vehicles. The number of collisions increases as the maximum number of vehicles flying in parallel increases. The take-off time interval, which affects the maximum number of active vehicles increases, also affects collisions, as in the previous case.

When the maximum number of active vehicles in the swarm increases, the energy consumption remains constant or even decreases slightly. These results indicate that the energy cost of adding new active agents is balanced by the contribution to swarm deployment. It is important to emphasize that the variability of the energy use decreases as the maximum number of active vehicles increases. However, beyond 5 active drones, the deployment time remains almost constant, regardless of the number of active agents.

It is interesting to compare the previous results with those presented in Figures 18 and 19 for complex maps. In this case, the time required to complete the deployment, the energy consumed and the number of collisions increase substantially. The SLABE behavior shows greater variability in all the observed variables. However, we can see that for a sufficient number of active individuals (greater than or equal to 10), both the time and energy consumption are comparable to those of the SEPHE behavior. In complex environments, the decrease in energy used due to increasing the take-off time interval is not appreciable, which indicates that shorter take-off time intervals are better when the deployment time decreases. However, collisions between devices will increase.

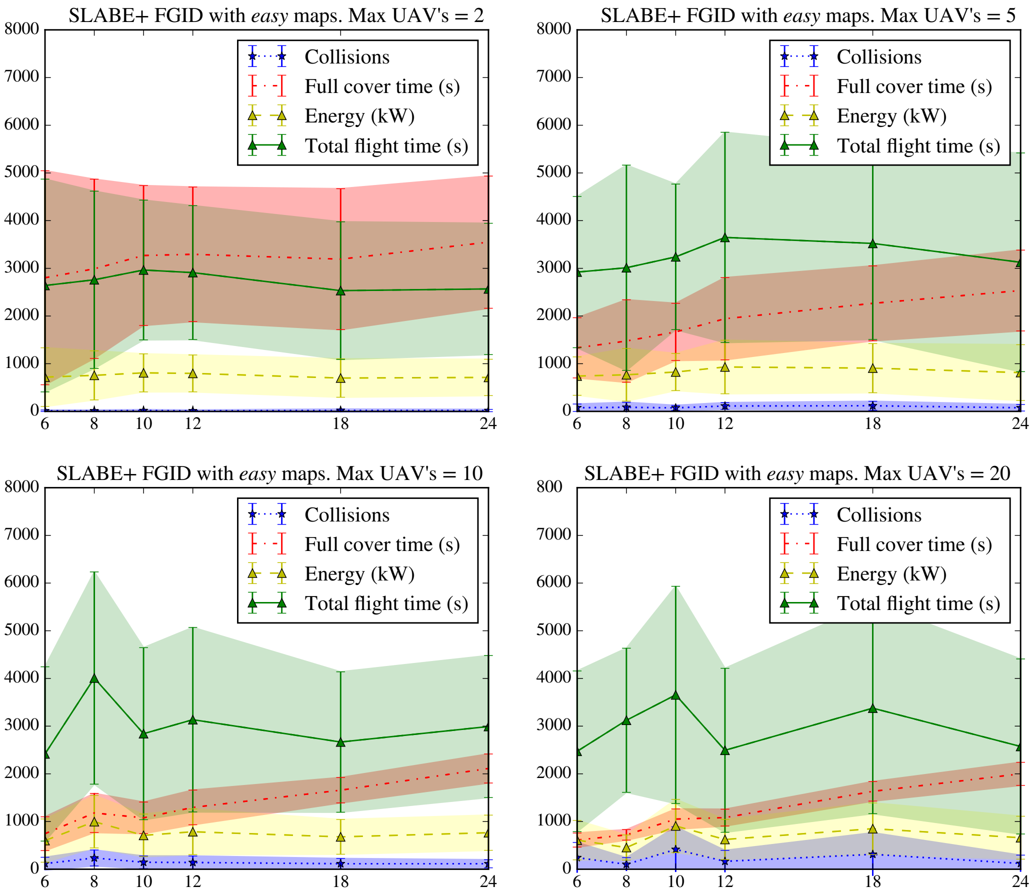


Figure 16. Results of the SLABE behavior simulations for the FGID take-off strategy in an *easy* environment. The graph shows the average evolution and variation of the number of collisions, the time required to achieve 95% coverage of the environment, the energy consumed by the swarm, and the total flight time for 600 simulations (50 simulations on the same map for each combination of take-off time interval and maximum number of vehicles in flight).

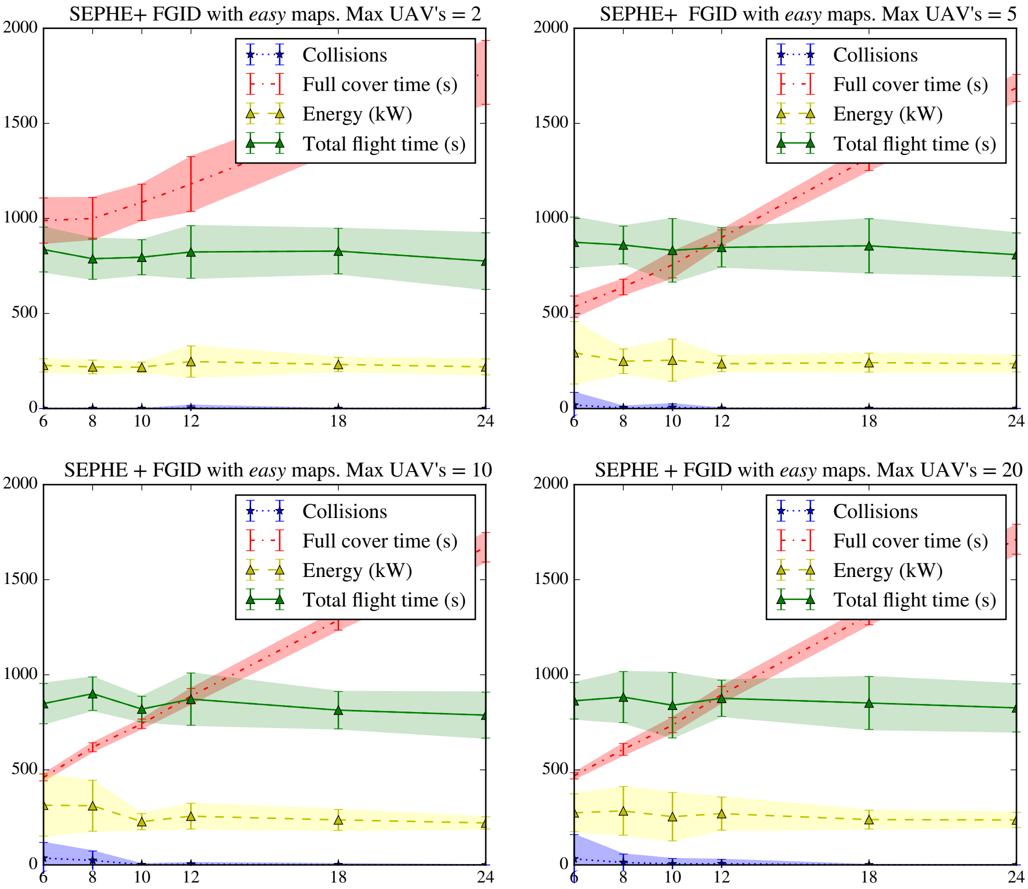


Figure 17. Results of the SEPHE behavior simulations for the FGID take-off strategy in an *easy* environment. The graph shows the average evolution and variation of the number of collisions, the time required to achieve 95% coverage, the energy consumed by the swarm, and the total flight time for 600 simulations (50 simulations on the same map for each combination of take-off time interval and maximum number of vehicles in flight).

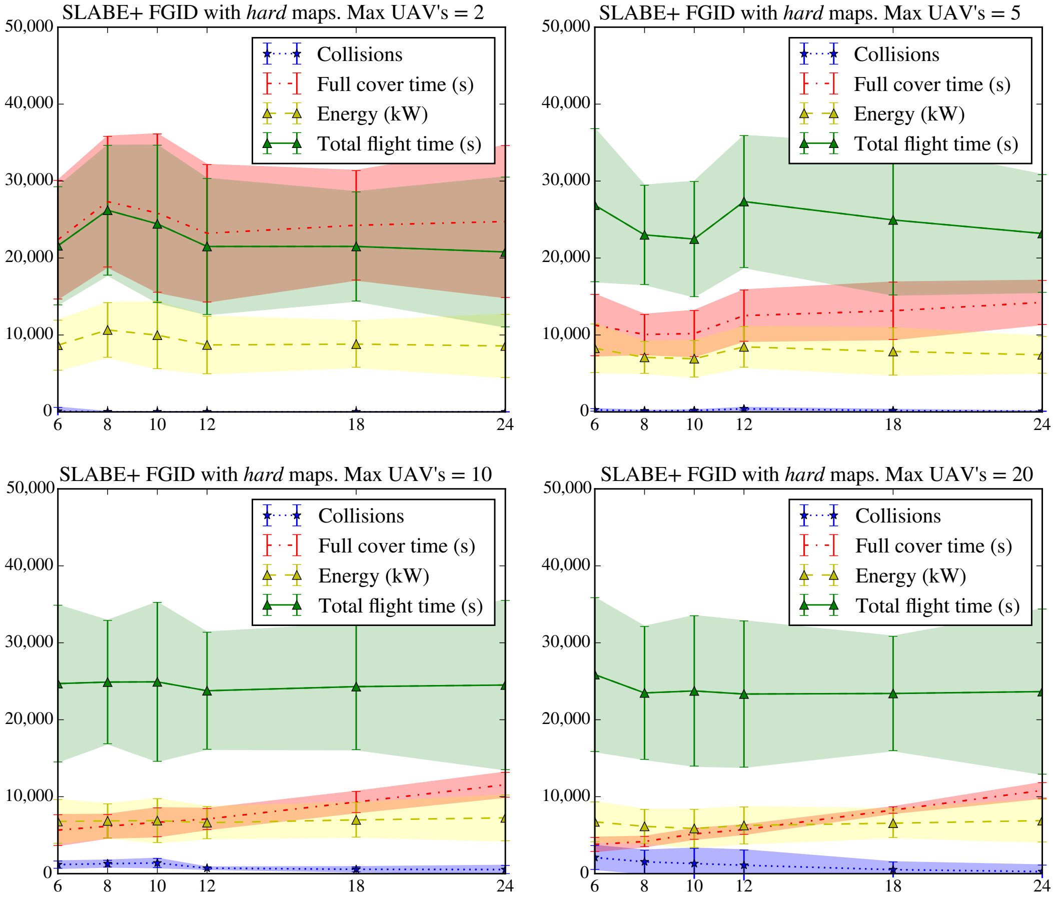


Figure 18. Results of the SLABE behavior simulations for the FGID take-off strategy in a *hard* environment. The graph shows the average evolution and variation of the number of collisions, the time required to achieve 95% coverage of the environment, the energy consumed by the swarm, and the total flight time for 600 simulations (50 simulations on the same map for each combination of take-off time interval and maximum number of vehicles in flight).

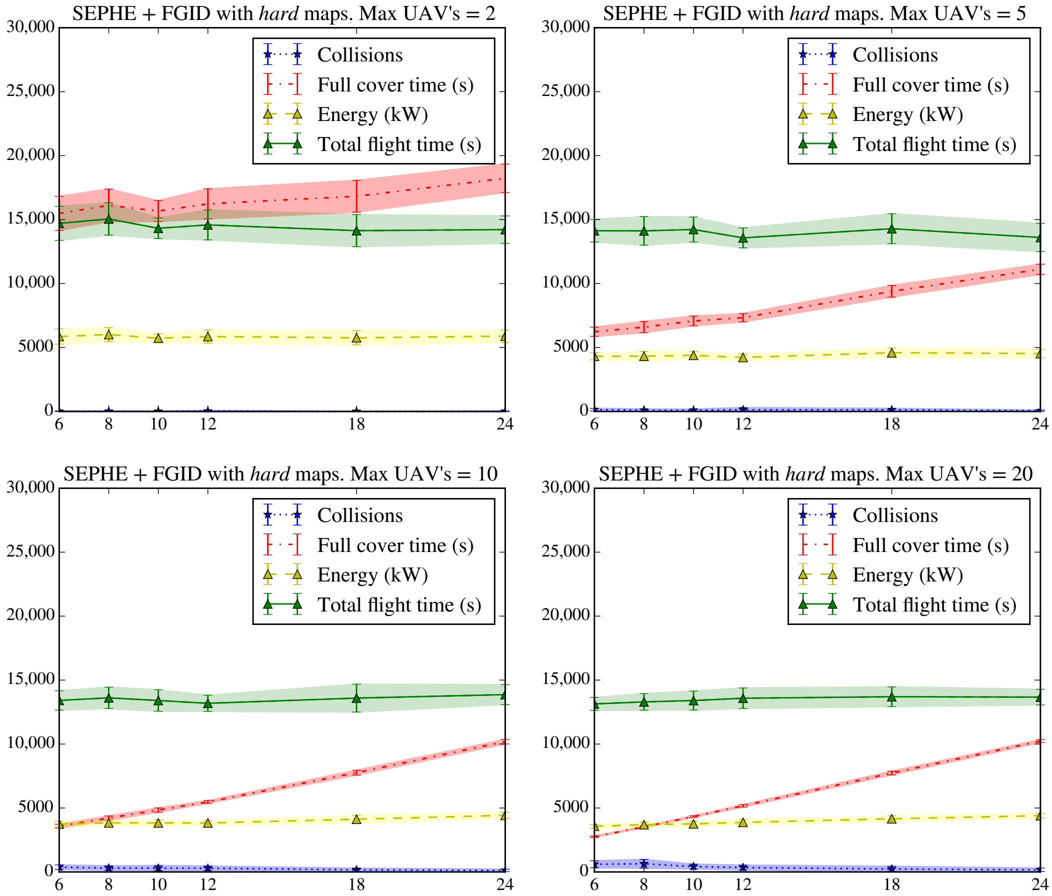


Figure 19. Results of the SEPHE behavior simulations for the FGID take-off strategy in a *hard* environment. The graph shows the average evolution and variation of the number of collisions, the time required to achieve 95% coverage of the environment, the energy consumed by the swarm, and the total flight time for 600 simulations (50 simulations on the same map for each combination of take-off time interval and maximum number of vehicles in flight).

5. Discussion

From the above data, we can draw some interesting conclusions regarding several of the aspects analyzed. Below, we will analyze the two types of proposed behavior, as well as the different deployment strategies.

5.1. Comparison of SLABE and SEPHE Behaviors

Two behaviors based on pheromones have been proposed to solve a common deployment task in unstructured environments. The behaviors differ mainly in the sensory needs required for their implementation. The SLABE strategy is much simpler and does not require a system for local positioning, which is required by the SEPHE strategy.

In all the tests, both behaviors have shown that they are capable of achieving satisfactory coverage of the environment, taking into account the energy limitations imposed by the dynamics of the UAVs of the swarm.

The data appear to confirm that a lower resolution of the location sensor corresponds to a higher collision rate and higher energy consumption of the swarm. We believe that this is due to the low resolution of the stigmergy sensor used in the SLABE approach. This sensor is not continuous and does not produce guidance information at all points in space, so the behavior is more subject to variation.

The increase in energy cost is determined by the granularity of the stigmergy sensor and by the indirect increase in the number of collisions.

However, it is important to note that the average deployment times in both cases are comparable. We believe this to be an important conclusion since it reinforces the thesis that a swarm with a greater number of less-informed individuals can perform the same task as a swarm with a smaller number of more-informed individuals.

If a system of stigmergy is available, as required by the SEPHE behavior, we can expect lower variability in all the parameters analyzed and a shorter time to achieve complete deployment. The systems required by each individual will increase the cost of the swarm. However, a swarm of low-cost drones with basic sensors can be used to develop a strategy with an average deployment time very similar to that of the SEPHE behavior at the cost of a higher probability of collision, greater energy needs (assuming that the drones that execute the behaviors have the same individual consumption) and a greater variability (but a much lower total cost).

However, the devices that can execute the SLABE behavior can weigh less, thereby drastically reducing the energy consumption. Therefore, the SLABE behavior may require less energy for its implementation (as observed in Figure 14, where the drones develop a realistic energy model based on the requirements of ARDrone commercial drones).

5.2. Take-Off Time Interval and Deployment Strategies

We considered how the take-off time interval and the different strategies used in the operation of the behaviors presented affected the results.

In the LTID strategy, the longer the take-off time interval is, the longer the deployment. However, for short take-off time intervals, the number of collisions between individuals increases considerably because the density of vehicles in a given area is not limited and therefore increases the probability of collision in the most active areas.

In general, the greater the number of active agents (shorter take-off time interval), the more energy is required. However, in some cases, a longer take-off time interval can increase, rather than decrease, the energy consumed by the swarm.

This result could be seen in *wall* maps with SLABE behavior because in some areas of the environment, there is not sufficient information about the beacons that provide robots with navigation information. Therefore, active robots develop a more random search behavior, increasing the flight time for some maps. The results indicate that for the LTID strategy, the best take-off time interval for the unstructured maps is between 12 and 18 s.

The FGID strategy, limits the maximum number of active vehicles that can circulate in the environment. The time required for deployment increases for low values of the maximum number of vehicles (2 or 5 vehicles maximum). However, for more than 10 vehicles, no substantial improvements and a greater probability of collision are observed, regardless of the behavior used. This probability stabilizes the results obtained for the LTID strategy, which has no limit on the number of active vehicles. This evidence suggests that between 10 and 20 active UAVs in parallel is optimal for large maps of the *hard* typology and between 5 and 10 active UAVs is optimal for small maps of the *easy* typology.

In general, it is convenient to limit the number of active individuals; therefore, the FGID strategy produces better results than those of the LTID strategy, at the cost of requiring mesh-type point-to-point communication (which allows the number of active individuals available at a given time to be known) and requiring to determine for a certain environment the maximum number of active agents.

6. Conclusions

In this article, two types of swarm behavior that use stigmergy as an indirect communication mechanism have been discussed. A microscopic model of both has been presented, considering the requirements for the final implementation. Specifically, SLABE behavior can be implemented with current hardware using drones and BLE beacons. The behavior has been tested via simulations;

this behavior covers the environment and manages to develop a simple but effective stigmergy. In the same way, the SEPHE behavior manages to develop an environmental deployment with a more complex stigmergy sensor that requires local location.

Moreover, the information available under both behavior types affects several parameters. The average deployment time in both cases is comparable, even though there is greater variability in the least-informed option.

However, we consider it to be remarkable that for complex environments, the less-informed approach can complete the deployment process in a similar way to that of the informed approach but with a lower cost (both economic and energetic). This result reinforces the thesis of this type of system: many simple agents can perform the same task as complex agents when they perform appropriate behavior.

Finally, we have shown how the FGID strategy gives better results than those of the LTID strategy. It is advisable to use mesh-type point-to-point communication (where the number of active individuals available at any given time is known) whenever possible. Otherwise, LTID is an effective deployment process that does not require a communications infrastructure.

Now that the performance of both behaviors has been demonstrated, our future line of research will test these behaviors on real robots and adjust the microscopic model used on physical robots.

Author Contributions: The authors have contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the Ministerio de Ciencia, Innovación y Universidades (Spain), project RTI2018-096219-B-I00. Project co-financed with FEDER funds.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Kuswadi, S.; Adji, S.I.; Sigit, R.; Tamara, M.N.; Nuh, M. Disaster swarm robot development: On going project. In Proceedings of the 2017 International Conference on Electrical Engineering and Informatics (ICELTICs), MDPI AG, Banda Aceh, Indonesia, 18–20 October 2017; Volume 20, pp. 45–50. [CrossRef]
2. Chung, S.; Paranjape, A.A.; Dames, P.; Shen, S.; Kumar, V. A Survey on Aerial Swarm Robotics. *IEEE Trans. Robot.* **2018**, *34*, 837–855. [CrossRef]
3. Roberts, J.F.; Stirling, T.S.; Zufferey, J.C.; Floreano, D. 2.5 D infrared range and bearing system for collective robotics. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 3659–3664.
4. Coppola, M.; Mcguire, K.; De Wagter, C.; Croon, G. A Survey on Swarming With Micro Air Vehicles: Fundamental Challenges and Constraints. *Front. Robot. AI* **2020**, *7*, 18. [CrossRef]
5. Li, Q.; De Rosa, M.; Rus, D. Distributed algorithms for guiding navigation across a sensor network. In Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, San Diego, CA, USA, 14–19 September 2003; ACM: New York, NY, USA, 2003; pp. 313–325.
6. Paradzik, M.; İnce, G. Multi-agent search strategy based on digital pheromones for UAVs. In Proceedings of the 2016 24th Signal Processing and Communication Application Conference (SIU), Zonguldak, Turkey, 16–19 May 2016; pp. 233–236. [CrossRef]
7. Khan, A.; Munir, A.; Kaleem, Z.; Ullah, F.; Bilal, M.; Nkenyereye, L.; Shah, S.; Nguyen, L.D.; Islam, S.M.R.; Kwak, K.S. RDSP: Rapidly Deployable Wireless Ad Hoc System for Post-Disaster Management. *Sensors* **2020**, *20*, 548. [CrossRef]
8. Pezeshkian, N.; Nguyen, H.G.; Burmeister, A. *Unmanned Ground Vehicle Radio Relay Deployment System For Non-Line-of-Sight Operations*; Technical Report; DTIC Document: Fort Belvoir, VA, USA, 2007 .
9. Miranda, K.; Molinaro, A.; Razafindralambo, T. A Survey on Rapidly Deployable Solutions for Post-disaster Networks. *IEEE Commun. Mag.* **2016**, *54*, 117–123. [CrossRef]
10. Nouyan, S.; Campo, A.; Dorigo, M. Path formation in a robot swarm. *Swarm Intell.* **2008**, *2*, 1–23. [CrossRef]

11. Nouyan, S.; Groß, R.; Bonani, M.; Mondada, F.; Dorigo, M. Teamwork in self-organized robot colonies. *Evol. Comput. IEEE Trans.* **2009**, *13*, 695–711. [CrossRef]
12. De Nardi, R.; Holland, O. Swarmav: A Swarm of Miniature Aerial Vehicles. Proceedings of the 21st Bristol UAV Systems Conference, April 2006. Available online: http://cogprints.org/5569/1/bristoluav21.pdf (accessed on 27 October 2020).
13. Gao, N.; Jin, S.; Li, X. 3D Deployment of UAV Swarm for Massive MIMO Communications. In Proceedings of the ACM MobiArch 2020 the 15th Workshop on Mobility in the Evolving Internet Architecture, London, UK, 21 September 2020; ACM: New York, NY, USA, 2020. [CrossRef]
14. Rudol, P.; Wzorek, M.; Conte, G.; Doherty, P. Micro unmanned aerial vehicle visual servoing for cooperative indoor exploration. In Proceedings of the Aerospace Conference, Big Sky, MT, USA, 1–8 March 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 1–10.
15. Rosalie, M.; Danoy, G.; Bouvry, P.; Chaumette, S. UAV Multilevel Swarms for Situation Management. In Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use—DroNet 16, Singapore, 26 June 2016; ACM Press: New York, NY, USA, 2016. [CrossRef]
16. Zhang, H.; Da, X.; Hu, H. Multi-UAV cooperative spectrum sensing in cognitive UAV network. In Proceedings of the 5th International Conference on Communication and Information Processing, Chongqing, China, 15–17 November 2019; ACM: New York, NY, USA, 2019. [CrossRef]
17. Gupte, S.; Mohandas, P.I.T.; Conrad, J.M. A survey of quadrotor Unmanned Aerial Vehicles. In Proceedings of the 2012 IEEE Southeastcon, Orlando, FL, USA, 15–18 March 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 1–6.
18. Alfeo, A.L.; Cimino, M.G.; Vaglini, G. Enhancing biologically inspired swarm behavior: Metaheuristics to foster the optimization of UAVs coordination in target search. *Comput. Oper. Res.* **2019**, *110*, 34–47. [CrossRef]
19. Zhang, G.; Hsu, L.T. A New Path Planning Algorithm Using a GNSS Localization Error Map for UAVs in an Urban Area. *J. Intell. Robot. Syst.* **2018**, *94*, 219–235. [CrossRef]
20. Rydell, J.; Tulldahl, M.; Bilock, E.; Axelsson, L.; Köhler, P. Autonomous UAV-based Forest Mapping Below the Canopy. In Proceedings of the 2020 IEEE/ION Position, Location and Navigation Symposium (PLANS), Portland, OR, USA, 20–23 April 2020; pp. 112–117.
21. Grzonka, S.; Grisetti, G.; Burgard, W. Towards a navigation system for autonomous indoor flying. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA'09), Kobe, Japan, 12–17 May 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 2878–2883.
22. Burgard, W.; Moors, M.; Stachniss, C.; Schneider, F.E. Coordinated multi-robot exploration. *Robot. IEEE Trans.* **2005**, *21*, 376–386. [CrossRef]
23. Stirling, T.; Roberts, J.; Zufferey, J.C.; Floreano, D. Indoor navigation with a swarm of flying robots. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA), Saint Paul, MN, USA, 14–18 May 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 4641–4647.
24. Corke, P.; Peterson, R.; Rus, D. Localization and navigation assisted by networked cooperating sensors and robots. *Int. J. Robot. Res.* **2005**, *24*, 771–786. [CrossRef]
25. Corke, P.; Peterson, R.; Rus, D. Coordinating aerial robots and sensor networks for localization and navigation. In *Distributed Autonomous Robotic Systems 6*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 295–304.
26. Zufferey, J.C.; Hauert, S.; Stirling, T.; Leven, S.; Roberts, J.; Floreano, D. Aerial Collective Systems. In *Handbook of Collective Robotics: Fundamentals and Challenges*; Pan Stanford Publishing: Singapore, 2013; pp. 609–660.
27. Stirling, T.; Wischmann, S.; Floreano, D. Energy-efficient indoor search by swarms of simulated flying robots without global information. *Swarm Intell.* **2010**, *4*, 117–143. [CrossRef]
28. Coppola, M.; Mcguire, K.; Scheper, K.; Croon, G. On-board communication-based relative localization for collision avoidance in Micro Air Vehicle teams. *Auton. Robot.* **2018**, *42*, 1787–1805. [CrossRef] [PubMed]
29. Stirling, T.; Floreano, D. Energy-time efficiency in aerial swarm deployment. In *Distributed Autonomous Robotic Systems*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 5–18.
30. O'Hara, K.J.; Balch, T.R. Distributed path planning for robots in dynamic environments using a pervasive embedded network. In Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, New York, NY, USA, 19–23 August 2004; IEEE Computer Society: Washington, DC, USA, 2004; Volume 3, pp. 1538–1539.

31. Batalin, M.A.; Sukhatme, G.S. Coverage, exploration and deployment by a mobile robot and communication network. *Telecommun. Syst.* **2004**, *26*, 181–196. [CrossRef]
32. Phan, T.A.; Russell, R.A. An effective collaboration algorithm for swarm robots communicating by sematectonic stigmergy. In Proceedings of the 2010 11th International Conference on Control Automation Robotics & Vision, Singapore, 7–10 December 2020; IEEE: Piscataway, NJ, USA, 2010; pp. 390–397.
33. Bamberger, R.J.; Watson, D.P.; Scheidt, D.H.; Moore, K.L. Flight demonstrations of unmanned aerial vehicle swarming concepts. *Johns Hopkins APL Tech. Dig.* **2006**, *27*, 41–55.
34. Faludi, R. *Building Wireless Sensor Networks: With ZigBee, XBee, Arduino, and Processing*; O'reilly: Springfield, MI, USA, 2010.
35. Luke, S.; Cioffi-Revilla, C.; Panait, L.; Sullivan, K.; Balan, G. Mason: A multiagent simulation environment. *Simulation* **2005**, *81*, 517–527. [CrossRef]

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.