Drone Deployment Algorithms for Effective Communication Establishment in Disaster Affected Areas

Bivin Varkey Varghese [†], Paravurumbel Sreedharan Kannan [†], Ravilal Soni Jayanth [†], Johns Thomas [†] and Kavum Muriyil Balachandran Shibu Kumar *,[†]

Department of Computer Science and Engineering, Rajiv Gandhi Institute of Technology, Kottayam 686501, India
* Correspondence: shibukumar@rit.ac.in
† These authors contributed equally to this work.

Abstract: Communication establishment is crucial for rescue operations in disaster affected areas. A standard tool for communication is the use of cell phones. However, they can be useless in situations where the cellular network's base stations are damaged in a disaster. A contemporary approach to re-establishing a communication network is by hosting base stations in drones. However, low battery life and difficulty in calculating the number of drones needed in different terrains are limitations of the above approach. This paper introduces a novel terrain-aware algorithm that calculates the minimum number of drones needed to cover an area with no voids in the network coverage. Our method ensures that the drones are deployed at optimal heights to maximize the average leftover energy in the network. We apply the algorithm for an actual location in *Pettimudi, India* and find the optimal number and positions of the drones to cover the area effectively without voids. In addition, we provide a simulation of the the communication establishment using above drones, and our experiments yield an average network efficiency of 98%, showing the effectiveness of our method.

Keywords: drone networks; drone-based disaster recovery; drone-based communication establishment; drone coverage filling algorithm

1. Introduction

The human race has always shown immense potential to manage and cope with natural disasters. As humans, we have developed various techniques to prevent and mitigate these life-threatening disasters. Setting up a communication infrastructure is widely discussed among such techniques, as normal means of communication might be devastated during disasters [1]. One of the most critical parts of disaster response is effective information exchange between information sources, emergency managers, and those impacted by the disaster event. During disaster events, the terrestrial communications infrastructure, such as mobile phone networks, TV networks, and radio networks, are frequently affected by the disaster impacts. This situation has a considerable impact on all preparedness, response, and recovery processes in disaster conditions [2]. Hence, there is always a need to maintain the entire disaster affected region under communication coverage throughout the rescue operations.

The communication between disaster management organizations, the general public, and other actors is a critical and fundamental factor in providing a quick and opportune response to all aspects of the disaster event. Many practical experiences show the importance of communication during a disaster [3].

One such case is the *Pettimudi Landslide* [4] that occurred in India on 6 August 2020. The disaster resulted in the snapping of communication lines and blockage of roads due to the uprooting of trees. This made it challenging for the local search groups to contact the base station. It was reported in the leading dailies that rescue groups had to stay near dead bodies for nearly three hours [5]. They needed to wait for other teams to arrive with a

vehicle or wireless device before they were able to share messages. The rescue operations were effective only after the restoration of mobile towers.

Another case is the *Sichuan Earthquake* [6] that occurred in China on 12 May 2008. During the disaster, telecommunication systems were affected. The Chinese government activated the use of the national communications satellite network to recover communication services in all affected areas and support other solution implementations for different types of data exchange between the entities in charge of disaster management. The importance of communication during a disaster can be seen in other events as well. Recent examples include Typhoon Faxai in 2019 in Chiba, Japan [7] and bush fires of New South Wales in 2019–2020 [8]. In the former case, the damage mainly happened to the roofs of buildings, and prompt damage assessment was able to quickly facilitate disaster insurance relief. The latter case required aerial monitoring of areas in order to understand the status and severity of the bush fires.

Drones are increasingly becoming a viable alternative to conventional cellular infrastructure in disaster-struck regions thanks to their low cost and availability [9]. Their easy availability and low weight make them potential candidates as communication support in any terrain [10]. An example of such a communication network establishment is shown in Figure 1. Drones can easily be deployed to the affected region from one or more base stations, although performance in terms of flight time due to battery power restrictions, the number of drones required, and terrain constraints all remain matters of concern [11].
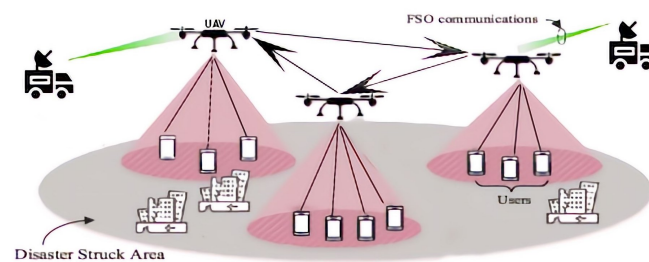
Figure 1. An example of a drone-based communication system.

This paper considers the idea of deploying drones to restore the communication system in a post-disaster area. As proposed here, a swarm of drones carrying network receptors is deployed in the disaster struck location. The drones can be deployed quickly and conveniently for temporary coverage until more permanent solutions are deployed, allowing recovery operations to continue smoothly. Using this method, the mobile phone network is restored and people can use their mobile phones to notify the concerned authorities. This enables fast establishment of reliable communications in any environment.

A significant concern with the above approach is the large number of drones required to cover the affected region. In addition, it is necessary to ensure that the deployed drones have sufficient leftover energy to serve the region long enough to conduct the rescue operation. Hence, we formulate a novel region-filling algorithm that brings the entire disaster affected area under drone network coverage without creating voids. Our algorithm considers the terrain data (altitude of various points in the region). We deploy the drones at optimal heights in order to maximize the system's average leftover energy. To the best of our knowledge, this approach towards communication establishment in a disaster affected area using stationary drones with maximum average leftover energy while minimizing the number of drones required is the first of its kind to be reported. The main research contributions of the paper can be summarized as follows:

1. Development of a terrain-aware region-filling algorithm that covers the entire disaster affected area under a cellular network.
2. Minimization of the required number of drones while maximizing the average leftover energy of the system.
3. A proof that our method uses the minimum number of drones possible.

4. Application and evaluation of the above methods using the parameters of a real location in *Pettimudi, India*.
5. A simulation of the region-filling algorithm for the terrain of Pettimudi.

The rest of this paper is organized as follows. Section 2 describes the state-of-the-art of the literature on the topic. Section 3 details the various algorithms we formulated. Section 4 explains the experiments and simulations we conducted and their results. Finally, the paper ends with a conclusion section describing our future work.

2. Related Works

There have been trials in establishing communication in a post-disaster region. AT&T is an American telecommunication company that provides AT&T Network Disaster Recovery (NDR) [12] units that restore their network in a disaster area by bringing in portable cellphone towers on trucks called *Cell on Wheels (COW)* [13]. The NDR team was a great aid during the California wildfires (2017), Hurricane Irene (2011), Hurricane Irma (2017), and Hurricane Maria (2017) [14], although transporting the COW is difficult when roads are damaged due to disaster impact. Furthermore, COW is unsuitable for disasters such as floods, landslides etc.

Another method is the use of satellite phones or two-way radios as communication devices [15]; however, these devices are not affordable for the general public, and government regulations may prevent the general public from using these devices. In addition, it is not easy to track their locations.

Using drones for communication is a growing research field due to their on-demand deployment and high flexibility [16]. Park et al. [17] proposed a formation control algorithm for a drone-based network in a planar region. The algorithm increases the network throughput by improving the network infrastructure. Zhang et al. [18] proposed deployment algorithms for a drone network that maximize the leftover energy of all drones. The algorithm takes account of any no-fly-zone constraints in the operational region. Ruan et al. [19] proposed a multi-drone coverage deployment model that focuses on efficient network coverage and energy conservation. Pankaj Kumar et al. [20] provided a framework for a network that uses drones as a relay node to achieve good quality of service, better spectral efficiency, and improved diversity. Mozaffari et al. [21] proposed a novel framework for a drone-enabled antenna array system that provides better service to users. It considers each drone as an element in the antenna array system and optimizes the drone positions in the array to maximize the antenna gain and minimize the service time for the ground users. Namvar et al. [22] developed an algorithm for deploying a heterogeneous set of drones that provide maximum wireless coverage for ground users in a planar area.

There have been attempts to improve the performance and efficiency of drone-based networks. Thibbotuwawa et al. [23] derived the energy consumption of drones and determined the influence of various parameters that affect the energy of the drones. Theys and Schutter [24] provided experimental results on the relationship between various factors of a quadcopter. Sundaresan et al. [25] developed the designs of low-altitude drone networks that provide Long-Term Evolution (LTE) connectivity. Panda et al. [26] proposed a drone-based network that uses Wi-Fi as an emergency communication network for disaster management. Li et al. [27] proposed a deployment algorithm for drone-based networks that improves network throughput and coverage in the disaster area. Ishigami and Sugiyama [28] proposed a method to improve drone network throughput by controlling the drones' height with respect to the number of users. Shakhatreh et al. [29] proposed an algorithm for drone deployment that minimizes the number of drones required for the network.

Lyu et al. [30] introduced a spiral algorithm that aims to minimize the number of base stations mounted on grounded vehicles. However, in situations where such vehicle access is limited, the technology is not applicable at all. A research paper by Plachy et al. [31] discussed the positioning of base stations in unmanned aerial vehicles. They developed an algorithm to find the optimal positions of the base stations through association of the

Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) evolutionary strategies. However, GA is computationally complex and PSO tends to fall into local optima. A paper by Ruzicka et al. [32] introduced a conditional generative adversarial neural network-based algorithm to optimize network coverage in similar situations. Their simulations showed that the algorithm converges to a quasi-optimal solution.

Recently, drone network-based disaster recovery has attained even wider acceptance. Furutani and Minami [33] discussed the various aspects of using drones for disaster risk mitigation and crisis response, especially in Japanese contexts. They explain the need for drones in disaster risk reduction, the different technologies applicable to this use case, and a variety of such cases in Japan. There have been studies focusing on the communication aspects as well; Chen and Cheng [34] proposed Reconfigurable Intelligent Surface (RIS) technology to support communications in post-disaster areas. In [35], the authors explained how a drone equipped with RIS can be used to cover users with an mmWave base station. An energy-aware multi-armed bandit algorithm is used to cover several high-capacity network hotspots while minimizing the energy consumption. In [36], the authors proposed the use of access and gateway drones to support communication over disaster-affected areas. An access drone collects information from its assigned area and flies to the nearby gateway drone to transfer the information. In order to avoid two access drones targeting the same gateway, the authors employ two multi-armed bandit-based algorithms, namely, the Kullback–Leibler upper confidence bound (KLUCB) and minimax optimal stochastic strategy (MOSS) algorithms. However, frequent flying can drain the energy of drones very quickly in this case.

The above works show that a drone-based network is a good option for establishing communication in a post-disaster area. However, there are gaps in the current research with respect to ensuring complete coverage while the minimum number of drones. In the above-mentioned drone deployment algorithms, the terrain is assumed to be a planar region, which can create bottlenecks in real-world implementation scenarios. Furthermore, maximization of average leftover energy in the system is another area that needs to be addressed in order to enable system operations over sufficiently long durations. Hence, in this study we concentrate on the development of a terrain-aware, deterministic, and energy efficient region-filling algorithm that uses the minimum number of drones to cover the entire disaster-affected area with no network voids.

3. Proposed Methods

3.1. Problem Statement

We aim to formulate a drone deployment algorithm to establish a mobile network for disaster management. The main considerations can be summarized as follows:

1. To develop a strategy for deploying drones to maintain the maximum possible average leftover energy and thereby increase the overall hovering time of all drones deployed.
2. To develop a region-filling method that guarantees coverage over the whole region with the minimum possible number of drones.
3. To deploy drones for communication establishment based on terrain and altitude analysis.
4. To determine the number and positions of base stations required to facilitate the network.

3.2. Methodology

Our aim is to formulate a swarm of drones in a disaster affected area to facilitate establishment of communication network without forming voids. A first-hand strategy to this end is the use of an AI-based method such as Particle Swarm Optimization (PSO) [37,38]. However, PSO-based algorithms are prone to falling into local optima, and the rate of convergence in the iterative process may be low [39]. Other AI-based algorithms, such as Genetic Algorithms and Simulated Annealing, face the same issues, apart from being too slow. As concerns to neural networks and deep learning, they are computationally expensive and often require frequent training to produce accurate results [40].

In our case, we have to produce deterministic positions for the drones in order to fully ensure network coverage in the disaster affected location while at the same time ensuring maximum average leftover energy to service the network. This needs to happen in a time-bound way using available computational resources. Therefore, we aim to develop a terrain-aware region-filling algorithm which is less computationally intensive. Here, we find the number and positions of drones and base stations required to create a robust temporary network in the target region. We accomplish this by considering the terrain, intending to cover the entire region without voids while minimizing the number of drones and maximizing their average leftover energy.

The drone deployment algorithm we propose consists of a *preprocessing module* and a *region-filling module*. Preprocessing involves obtaining terrain data on the disaster-struck region and then converting this into a suitable data structure for the region-filling algorithm. The region-filling algorithm provides a powerful solution to the above problem. The output of the region-filling algorithm contains a list of three-dimensional coordinates of the positions of drones, with base station indication, leftover energy, and hovering time. Finally, we simulate the deployment of drones based on the positional information obtained from the Region-Filling algorithm. An outline of the process is shown in Figure 2.
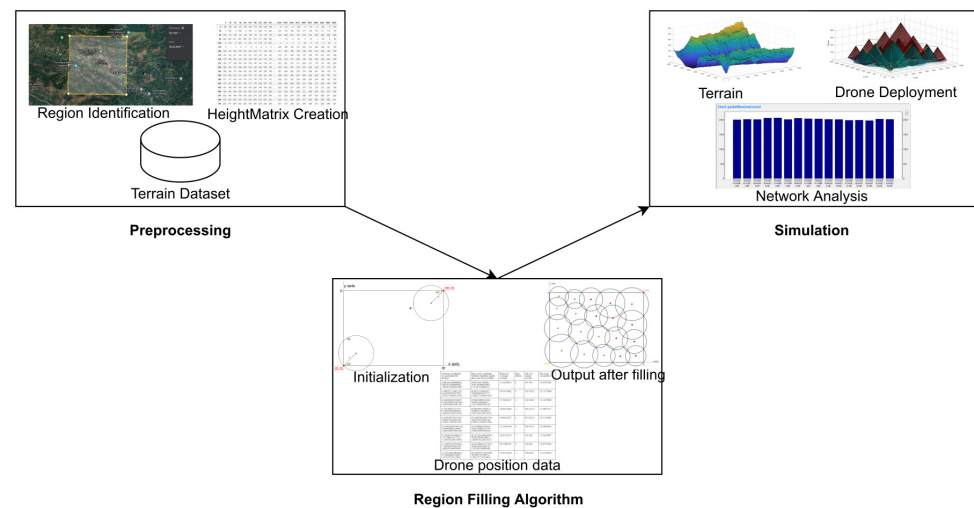
Figure 2. An outline of the process.

3.3. Preprocessing

The first step in preprocessing is to obtain the terrain data of the disaster struck region. This information is obtained in JSON format and then converted to the Cartesian Coordinate system required by the region-filling algorithm. The elevation profile of a region is commonly available in *Degree Minute Second (DMS): Altitude* format [41].

In our work, the Universal Transverse Mercator (UTM) coordinate system is used instead of the latitude–longitude coordinate system in order to precisely determine the distance between two points. If the two points are inside a UTM zone, the distance between the two points can be easily determined using the UTM coordinates [42].

3.3.1. The Preprocessor Module

The actions of the preprocessor module include:

- Construction of the *Height Matrix(H)* data structure.
- Calculation of the environment parameters required by the region-filling algorithm.

Height Matrix Data Structure

The *Height Matrix (H)* data structure stores the normalized altitudes of different locations in the disaster affected area. The altitude data are obtained as a JSON file (*Coordinate: Altitude* format). A normalization factor is calculated by finding the altitude of the

lowest-lying point in the terrain. The normalized altitudes are obtained by subtracting the normalization factor from each altitude, then populated to the Height Matrix data structure.

Environmental Parameters

We define the various environmental parameters specific to the drones and the terrain in Table 1. We note the average altitude of the terrain from *H*, then add a threshold value based on the terrain elevation profile. The sum of the average normalized altitudes and threshold value forms H_{min}.

Table 1. The various parameters of a drone.

Parameter	Description
m	Length of the region
n	Breadth of the region
H_{min}	Minimum altitude of a drone
H_{max}	Maximum permissible height of a drone
h_{td}	Threshold height for drone deployment
$SH_{x,y,r}$	Height of the lowest point in a circular region with center at (x, y) and radius r (SH is a sub Height Matrix, a subset of H)
B	Initial energy of a drone
W	Ratio of horizontal lift to drag ratio to vertical lift to drag ratio
C	Energy dissipation rate per kilometer
α	Coverage angle of network drone in degrees

3.4. Region Filling

Region filling is the core part of drone deployment; here, we implement a strategy to cover a rectangular region without any voids using circles of different radii. Each circle represents the coverage area of each drone. A drone deployed at a particular height is considered to have a cone-shaped coverage region, as shown in Figure 3. Usually, when a rectangular region is filled using circles, there will be voids in between. However, we use overlapping circles to avoid leaving any regions without network coverage. As this leads to wastage or redundancy of coverage, we aim to minimizing this wastage, and thereby the required number of drones.
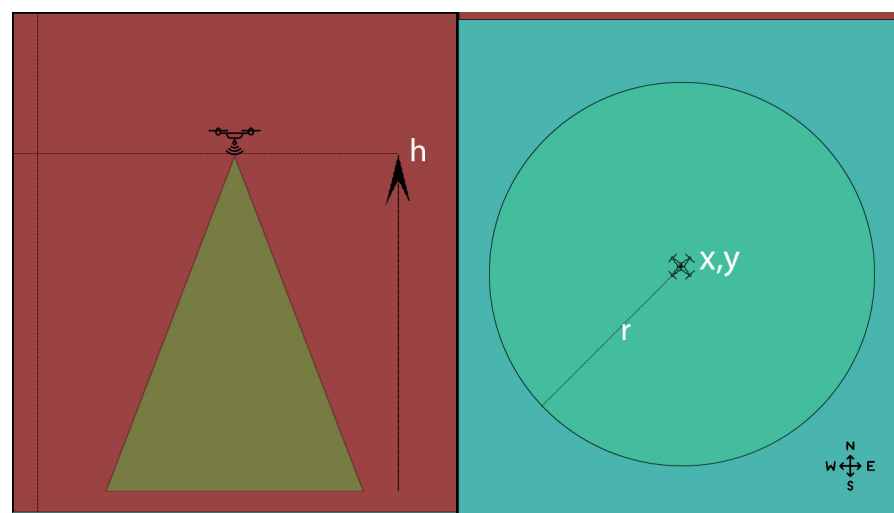
Figure 3. Front view and top view of network coverage of a drone deployed at a particular height.

We define H_{max} as the maximum altitude (in km) where a drone can be positioned such that any mobile device connected to it has sufficient network strength. Furthermore, H_{max} must be greater than the tallest obstacle within the whole region. While the region-

filling algorithm performs filling coverage in two dimensions, it incorporates terrain data at various points for the practical deployment of drones.

3.4.1. Determining the Number and Positions of Base Stations

Drones are deployed from corresponding base stations. It is necessary to determine both the number of base stations required and their positions. In order to fix the base stations in a deterministic way, we adopt a strategy whereby they are always placed at the corners of the rectangular region. Therefore, one, two, three, or four base stations are possible for a rectangular region. Increasing the number of base stations reduces the required number of drones, as the drones close to base stations can be deployed at greater heights, thereby increasing the coverage area. However, our experiments (detailed in Section 4) show that increasing the number of base stations beyond two does not provide any clear advantage; hence, we fix the number of base stations at two, located at opposite corners. One base station is fixed as the origin $(0, 0)$ and the other as (m, n), as shown in Figure 4.
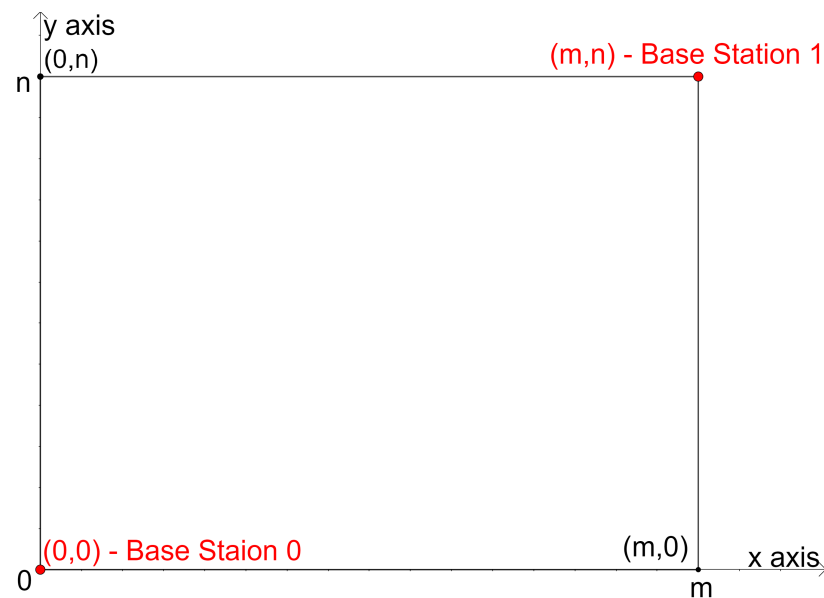
Figure 4. The base stations are located at diagonally opposite corners.

3.4.2. Outline of the Region-Filling Algorithm

The algorithm begins to fill the area with drone coverage, specifically, circles, from the origin towards the other corner. This is achieved by filling each *x*-directional and *y*-directional inner layer at each iteration until the whole region is covered. The algorithm considers the following attributes of a drone object *d*:

1. (x, y)—*x* and *y* coordinates of the position of a drone
2. *r*—radius of the cone-shaped network coverage
3. *h*—height of the cone-shaped network coverage
4. *b*—an indication of the base station; a value of 0 means the base station of the drone is at $(0, 0)$ and 1 means the base station of the drone is at (m, n)
5. *e*—leftover energy of the drone when the drone reaches its assigned location starting from its base station
6. h_{ref}—the maximum value among the heights of reference points associated with each drone, which is needed in order to cope with the terrain of the affected region
7. *north, south, west, east*—the northern, southern, western, and eastern neighbouring drones of the network drone

We use h_{ref} to denote the height of the reference point of a drone. There are many reference points associated with each drone object. These points can be the corner points

where a circle intersects other circles, the *x*-axis, or the *y*-axis; thus, h_{ref} is the maximum value of the heights of all reference points of a particular drone. A view of reference points and h_{ref} is shown in Figure 5.
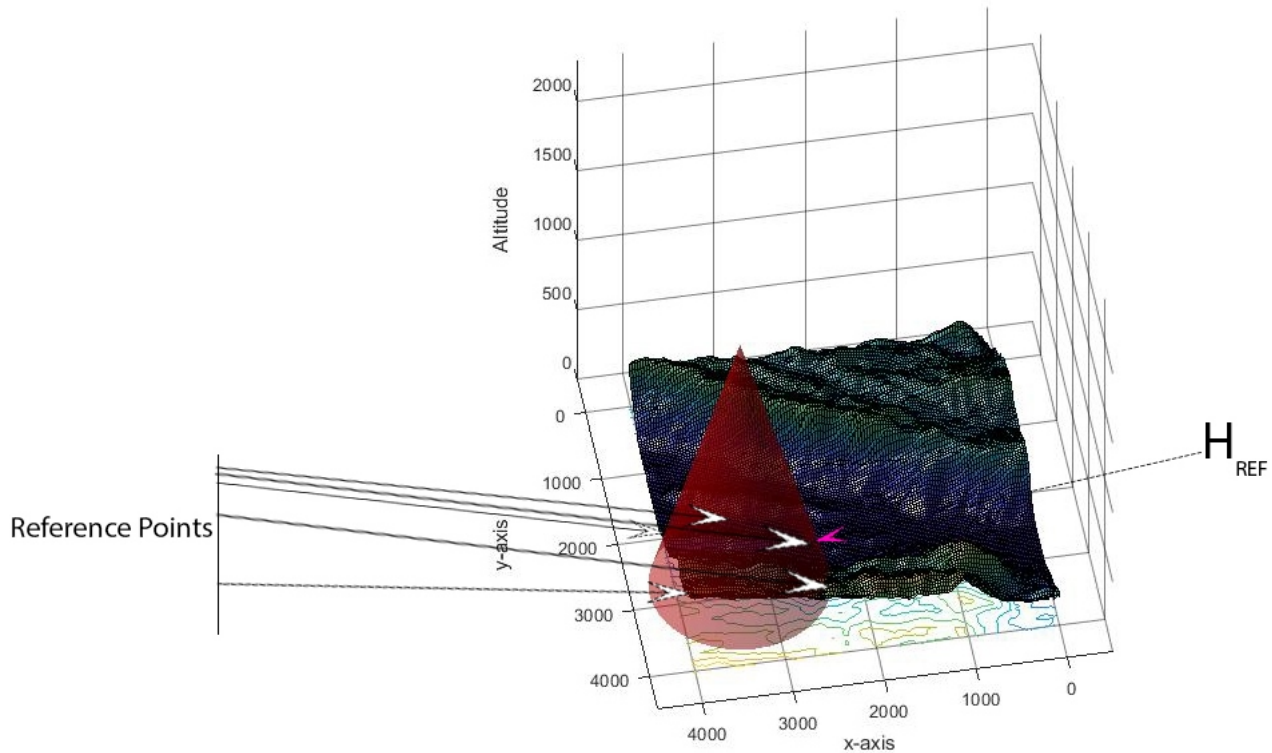
Figure 5. A view of reference points and h_{ref} for a drone in example terrain.

The algorithm fills the rectangular region layer by layer, starting from the base station at (0,0) and moving towards the one at (*m*, *n*) while assigning each drone to the nearest base station. We follow the approach of filling the circles along the *x*-axis first, followed by the *y*-axis. The algorithm maintains a few more variables, which are provided below:

1. *D*—an ordered collection of drone objects (the output of the algorithm)
2. X_{ref}—a reference *x*-coordinate for *y*-directional filling
3. Y_{ref}—a reference *y*-coordinate for *x*-directional filling

An outline of the method is shown in Figure 6.

3.4.3. Calculating Various Drone Parameters

One of the prime considerations in drone deployment is that every drone should have the same leftover energy on deployment [18]. In this way, drones nearer to the base stations can be deployed at greater heights to cover the maximum area on the surface.

We use drones with the same battery capacity and initial energy in the proposed method. After a drone is deployed, the leftover energy can be calculated as follows [18]:

$$e_i = B - C \times (dist_i + W \times (h_i - h_b))$$
$$e_i = e_j$$

(1)

for $1 \leq i \neq j \leq n$, where:

- h_i—the height of the *i*th drone
- e_i—the leftover energy of the *i*th drone
- h_b—the height of the base station
- *B*—the initial energy of all drones
- *C*—the energy parameter (same for all drones)

- *W*—the ratio of horizontal lift to drag ratio to vertical lift to drag ratio
- $dist_i$—the horizontal distance travelled by the *i*th drone

One of our research objectives is to maximize the network coverage of the region. From Equation (1), the maximum leftover energy is obtained when $dist_i = 0$ and $h_i = h_{max}$. Hence,

$$e = B - C \times W(h_{max} - h_b) \tag{2}$$

therefore, the maximum leftover energy is obtained when the drone is placed directly above the base station. From Equations (1) and (2), we obtain

$$B - C \times W(h_{max} - h_b) = B - C \times (dist_i + W \times (h_i - h_b))$$
$$\Rightarrow h_i = \frac{W \times H_{max} - dist_i}{W} \tag{3}$$
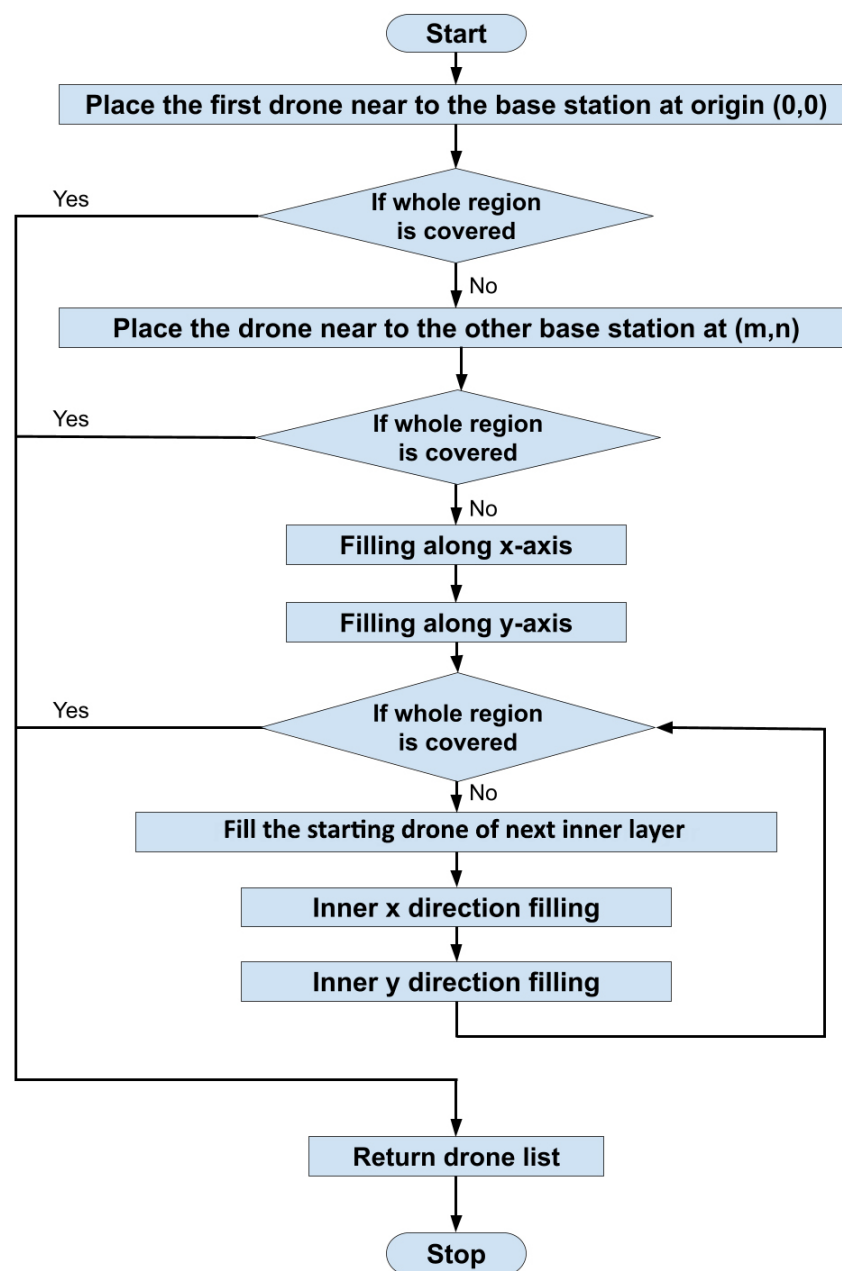
Figure 6. An outline of the Region-Filling algorithm.

For the first drone, $dist_1 = r_1$, where r_1 is its radius of coverage. Because the coverage region is cone-shaped, $r_i = h_i \times tan\alpha$; therefore,

$$B - C \times W(h_{max} - h_b) = B - C \times (h_1 \times tan\alpha + W \times (h_1 - h_b))$$
$$\Rightarrow h_1 = \frac{W \times H_{max}}{W + tan\alpha} \tag{4}$$

The calculated height value h is adjusted to be within H_{min} and H_{max}. The radius of coverage is calculated as

$$r = h \times tan\,\alpha \tag{5}$$

where α is the coverage angle of the drone.

Finally, in [18] the authors suggest an equation for finding the leftover energy e in one dimension:

$$e = B - C \times d_i,$$
$$d_i = |x_i - x'_i| + W_i \times h_i \tag{6}$$

where B is the initial energy of the drone, C is the energy dissipation rate, and x_i and x'_i are the positions of the base station and the drone, respectively. In our case, we have to consider both the Euclidean distance between the base station and the drone and the vertical distance that the drone has traveled $(h - h_b)$. Hence, the above formula converges to

$$e = B - C \times (dist + W \times (h - h_b)) \tag{7}$$

where h_b is the height of the base station. The height of a deployed drone and its coverage region are shown in Figure 7.
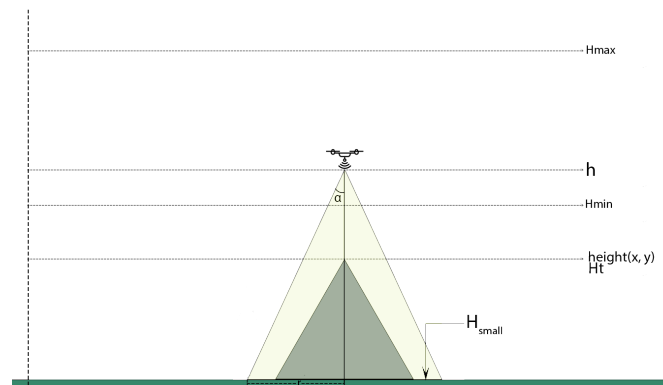
Figure 7. Visualization of a deployed drone and its coverage.

In the above equation, the leftover energy is calculated following the assumption that the terrain is a plane. However, we use a threshold height H_{min} (defined in Section 3.3) for practical situations, below which the drone is not deployed. Obviously, this may not guarantee equal leftover energy for every drone, however, the average leftover energy for the entire system is greater than that for a system with drones deployed at equal heights. The procedure is described in Algorithm 1. Note that we fix the height of a drone to be the value of h_{fixed} (for the first two drones where $h_{fixed} \neq 0$) in lines 11 and 12. Otherwise, we check whether the difference between the actual height $(h + h_{ref})$ computed for the drone and the height of the obstacle (h_t) is less than h_{td}; if it is, we place the drone at a height h_{td} above the obstacle (line numbers 19 and 20).

We try to cover the entire rectangular region with coverage circles of drones from the first base station to the second. We use a layered approach in which each layer consists of a starting circle, a set of circles along the x-direction, and a set of circles along the y-direction. The initial starting circle is placed near the base station at $(0, 0)$.

Algorithm 1 FindDroneParameters—Calculating the base station (b), height (h), radius of coverage (r), and leftover energy (e) of the drone

---

Input: (x, y)—position of the drone, h_{ref}—height of reference point, $h_{fixed} = 0$
Output: b, h, r, e

1: $dist_1 \leftarrow \sqrt{x^2 + y^2}$
2: $dist_2 \leftarrow \sqrt{(x - m)^2 + (y - n)^2}$
3: $dist \leftarrow Minimum(dist_1, dist_2)$
4: **if** $dist = dist_1$ **then**
5:      $b \leftarrow 0$
6:      $h_b \leftarrow H_{0,0}$
7: **else**
8:      $b \leftarrow 1$
9:      $h_b \leftarrow H_{m,n}$
10: **end if**
11: **if** $h_{fixed} \neq 0$ **then**
12:      $h \leftarrow h_{fixed}$
13: **else**
14:      $h \leftarrow \dfrac{W \times H_{max} - dist}{W}$
15:      **if** $h < H_{min}$ **then**
16:          $h \leftarrow H_{min}$
17:      **end if**
18:      $h_t \leftarrow H_{x,y}$
19:      **if** $(h + h_{ref} - h_t) < h_{td}$ **then**
20:          $h \leftarrow h_t + h_{td} - h_{ref}$
21:      **end if**
22: **end if**
23: $r \leftarrow \tan(\alpha) \times h$
24: $h_{small} \leftarrow SH_{x,y,r}$
25: **if** $h + h_{ref} - h_{small} > H_{max}$ **then**
26:      $h \leftarrow H_{max} + h_{small} - h_{ref}$
27:      $Perform$ 16 to 18
28: **end if**
29: $r \leftarrow \tan(\alpha) \times h$
30: $e \leftarrow B - C \times (dist + W \times (h - h_b))$
31: **return** (b, h, r, e)

3.4.4. Filling the Coverage Circles Along the X and Y Axes

Figure 8 shows how we find circles to be filled along the x-axis. The initial circle near the base station $(0, 0)$ is placed to pass through $(0, 0)$. The value of d for the initial drone is the coverage radius.

The center of the initial circle $(d.x, d.y)$ is placed such that the angle made by the line between $(d.x, d.y)$ and $(0, 0)$ and the x-axis is $45°$, meaning that any wasted coverage is equally distributed along both the x and y directions. The point at which the circle meets the x-axis is marked as (x_1, y_1). To find the next circle in the x-direction, we assume that it has the same radius (r) as the initial circle and grazes the x-axis. In the figure, its center is (x_2, y_2). In order to distribute the wastage equally in both directions, we fix the center of the new circle at (x, y) at a distance r from (x_1, y_1) along the angle (θ), obtaining a new center and radius for the new circle at (x, y). We refine the new circle again with respect to the new center and new radius. This procedure continues until the center in the previous round and that in the current round match. This is analogous to the rounds of the *k-means* clustering algorithm [43]. The detailed method used for filling circles along the x-axis is provided in Algorithm 2.

The algorithm uses two functions, *ArbitraryDistant* and *EquiDistant*. The former finds a point (x, y) at a distance $r_1 + r$ from (x', y') and at a distance $r_2 + r$ from (x'', y''), where

(x', y') and (x'', y'') are any two points and r_1, r_2, and r are arbitrary. There may be two different points for (x, y) on a plane; in order to always choose the point inside the rectangle, we use a separate variable *special* in the algorithm. The latter (*EquiDistant*) fixes the position of a drone object at an equal distance from the positions provided by the parameter points.
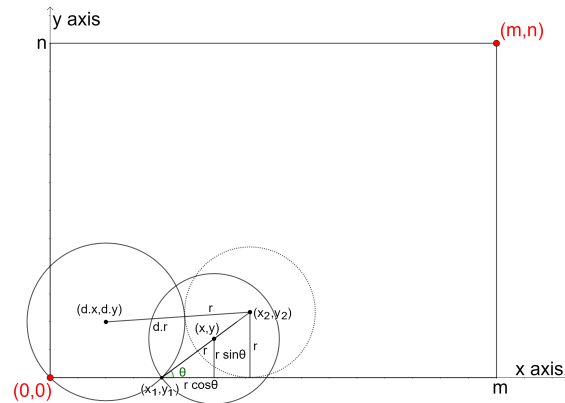
Figure 8. Finding the next circle along the x-axis (*FillAlongXAxis*).

We continue the process of finding new circles as long as it is possible to set the center of the new circle equidistant from the points (x_1, y_1) and $(m, 0)$. The final output of filling along the x-axis is shown in Figure 9.
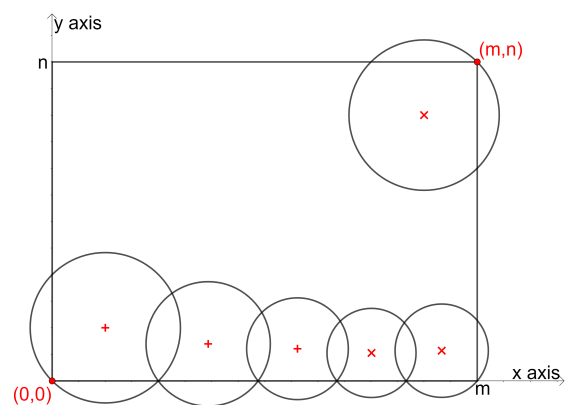
Figure 9. The output of filling circles along the x-axis.

Filling the circles along the y-axis (*FillAlongYAxis*) is quite similar to the procedure described above. The only difference is the change in direction, as shown in Figure 10.
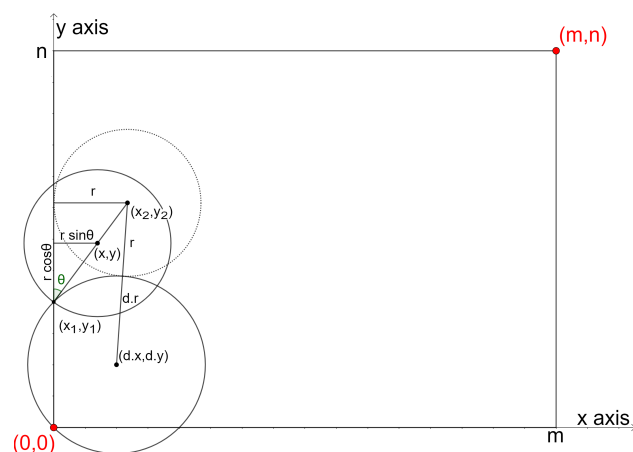
Figure 10. Finding the next circle along y-axis (*FillAlongXAxis*).

Algorithm 2 FillAlongXAxis—Filling drone coverage circles along x-axis

---

    Input: d—a drone object representing the starting circle of the outer layer
    Output: updated D—collection of drones, updated y_{ref}—reference y-coordinate for x **direction filling.**

  1: $indication \leftarrow EquiDistant(d, \{(0,0),(m,0)\}, 0)$
  2: **while** $indication = -1$ **do**
  3:     $r \leftarrow d.r$
  4:     $x_1 \leftarrow d.x + \sqrt{d.r^2 - d.y^2}, y_1 \leftarrow 0$
  5:     $h_{ref} \leftarrow H_{x_1, y_1}$
  6:     $d.h_{ref} \leftarrow Maximum(d.h_{ref}, h_{ref})$
  7:     $(d.b, d.h, d.r, d.e) \leftarrow FindDroneParameters(d.x, d.y, d.h_{ref})$
  8:     $x_2 \leftarrow d.x + \sqrt{(d.r + r)^2 - (r - d.y)^2}, y_2 \leftarrow r$
  9:     $\theta \leftarrow tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right)$
10:     $x \leftarrow x_1 + r \times cos\theta, y \leftarrow y_1 + r \times sin\theta$
11:     $(b, h, r, e) \leftarrow FindDroneParameters(x, y, h_{ref})$
12:     $Perform\ 8\ to\ 10$
13:     $coordinates \leftarrow \{(x, y)\}$
14:     **while** $true$ **do**
15:         $(b, h, r, e) \leftarrow FindDroneParameters(x, y, h_{ref})$
16:         $Perform\ steps\ 8\ to\ 10$
17:         **if** $(x, y) \in coordinates$ **then**
18:             $(b_1, h_1, r_1, e_1) \leftarrow FindDroneParameters(x, y, h_{ref})$
19:             **while** $h_1 > h$ **do**
20:                 $(b, h, r, e) \leftarrow (b_1, h_1, r_1, e_1)$
21:                 $Perform\ steps\ 8\ to\ 10$
22:                 $(b_1, h_1, r_1, e_1) \leftarrow FindDroneParameters(x, y, h_{ref})$
23:             **end while**
24:             **break**
25:         **end if**
26:         $coordinates \leftarrow coordinates \cup \{(x, y)\}$
27:     **end while**
28:     $d_1 \leftarrow Drone(x, y, r, h, b, e, h_{ref})$
29:     $d.east \leftarrow d_1, d_1.west \leftarrow d$
30:     $D \leftarrow D \cup \{d_1\}$
31:     $d \leftarrow d_1$
32:     $(x, y) \leftarrow ArbitraryDistant(x_1, y_1, m, 0, 0, 0, r)$
33:     **if** (x, y) $are\ defined$ **then**
34:         $indication \leftarrow 0$
35:     **else**
36:         $indication \leftarrow -1$
37:     **end if**
38: **end while**
39: $indication \leftarrow EquiDistant(d, \{(x_1, y_1),(m,0)\}, 0)$
40: $y_{ref} \leftarrow d.y + \sqrt{d.r^2 - (m - d.x)^2}$

3.4.5. Inner Layer Filling

Having obtained an outer layer of coverage circles along the x and y axes, we now concentrate on the inner layers. We introduce the concept of *TwoNeighbours* and *ThreeNeighbours*. *TwoNeighbours* is the procedure by which we find the center and radius of a circle based on its two neighbours, as shown in Figure 11a , where we find (x, y) and r based on the parameters of its neighbouring two circles.
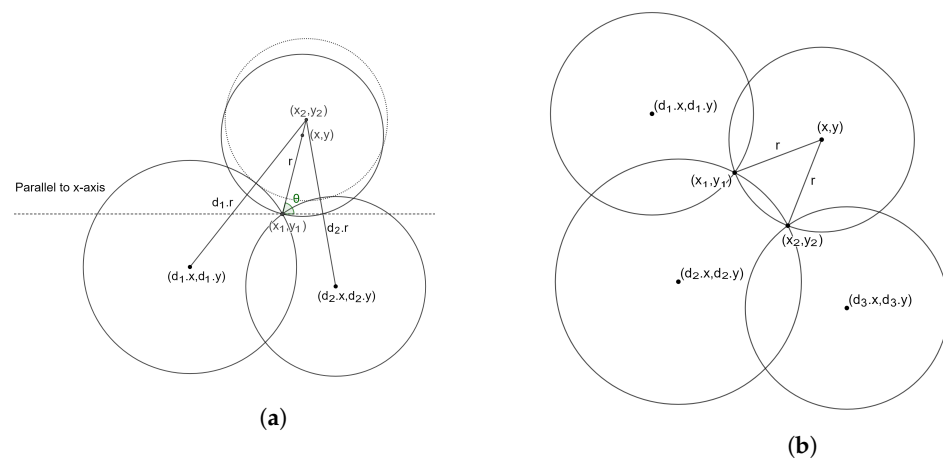
Figure 11. Principles behind: (**a**) *TwoNeighbours* and (**b**) *ThreeNeighbours*.

The working of *TwoNeighbours* is similar to that of the algorithms for filling circles along the x and y axes. We choose a point (x_2, y_2) at a distance $d_1.r + r$ from $(d_1.x, d_1.y)$ and $d_2.r + r$ from $(d_2.x, d_2.y)$. Because the value of r is initially unknown, it is chosen as the average of $d_1.r$ and $d_2.r$. We find the center of the new circle (x, y) at an angle θ along the line between (x_1, y_1) and (x_2, y_2) at a distance r from (x_1, y_1). We continue to change the values of (x_2, y_2) and (x, y) (a similar situation as in algorithm *FillAlongXAxis*) until we do not obtain a new point for (x, y). The detailed algorithm is shown in Algorithm 3.

The concept of *ThreeNeighbours* is simpler than that of *TwoNeighbours*, as we have a specific point (x, y), as shown in Figure 11b; here, (x_1, y_1) is a point at a distance r from both (x_1, y_1) and (x_2, y_2). The value of r is the average radii of all three neighbouring circles.

We find each circle in the inner layers based on the *TwoNeighbours* and *ThreeNeighbours* concepts. Wherever possible, we use *ThreeNeighbours*, as it is easier to process. The detailed algorithm used for inner x-direction filling is provided in Algorithm 4 (Filling along the y-axis follows a similar procedure). Figure 12 shows the output after filling the inner layers in both the x and y directions.
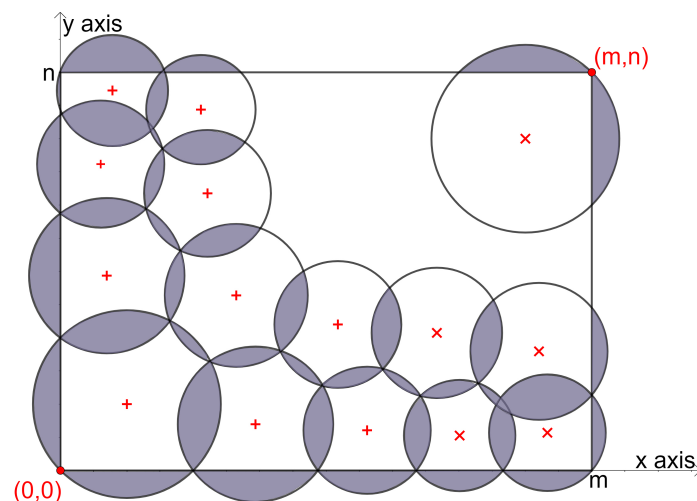
Figure 12. The output after filling an inner layer in both the x and y directions. The areas marked in grey show the wastage of coverage.

Algorithm 3 TwoNeighbours—Calculating the position of a third drone object based on the positions of its two neighbours

---

    Input: d_1, d_2—**two neighbouring drone objects**
    Output: d—**new drone object**

1: **if** $d_1 \,||\, d_2$ *not defined* **then**
2:     $d \leftarrow not\ defined$
3: **else**
4:     $r \leftarrow Average(d_1.r, d_2.r)$
5:     $(x_1, y_1) \leftarrow ArbitraryDistant(d_1.x, d_1.y, d_2.x, d_2.y, d_1.r, d_2.r, 0)$
6:     $h_{ref} \leftarrow H_{x_1,y_1}$
7:     $d_1.h_{ref} \leftarrow Maximum(d_1.h_{ref}, h_{ref})$
8:     $(d_1.b, d_1.h, d_1.r, d_1.e) \leftarrow FindDroneParameters(d_1.x, d_1.y, d_1.h_{ref})$
9:     $d_2.h_{ref} \leftarrow Maximum(d_2.h_{ref}, h_{ref})$
10:    $(d_2.b, d_2.h, d_2.r, d_2.e) \leftarrow FindDroneParameters(d_2.x, d_2.y, d_2.h_{ref})$
11:    *Perform steps 4 and 5*
12:    $(x_2, y_2) \leftarrow ArbitraryDistant(d_1.x, d_1.y, d_2.x, d_2.y, d_1.r, d_2.r, r)$
13:    **if** $x_2 = x_1$ **then**
14:       $\theta \leftarrow 90°$
15:    **else**
16:       $\theta \leftarrow tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right)$
17:    **end if**
18:    $x \leftarrow x_1 + r \times cos\theta, y \leftarrow y_1 + r \times sin\theta$
19:    $(b, h, r, e) \leftarrow FindDroneParameters(x, y, h_{ref})$
20:    *Perform steps 12 to 18*
21:    $coordinates \leftarrow \{(x, y)\}$
22:    **while** *true* **do**
23:       *Perform steps 12 to 19*
24:       **if** $(x, y) \; \epsilon \; coordinates$ **then**
25:          $b_1, h_1, r_1, e_1 \leftarrow FindDroneParameters(x, y, h_{ref})$
26:          **while** $h_1 > h$ **do**
27:             $(b, h, r, e) \leftarrow b_1, h_1, r_1, e_1$
28:             *Perform steps 12 to 18*
29:             $(b_1, h_1, r_1, e_1) \leftarrow FindDroneParameters(x, y, h_{ref})$
30:          **end while**
31:          **break**
32:       **end if**
33:       $coordinates \leftarrow coordinates \cup \{(x, y)\}$
34:    **end while**
35:    $d \leftarrow Drone(x, y, r, h, b, e, h_{ref})$
36:    $d_1.east \leftarrow d_2.north \leftarrow d$
37:    $d.west \leftarrow d_1, d.south \leftarrow d_2$
38: **end if**
39: **return** d

Algorithm 4 InnerXDirectionFilling—Filling inner layer along the x direction

---

Input: d—a drone object that represents the starting circle of the inner layer
Output: updated D—collection of drones, updated y_{ref}—reference y—coordinate for x direction filling.

1: $indication \leftarrow -1$
2: **while** $indication = -1$ **do**
3:     $d_1 \leftarrow d.south, d_2 \leftarrow d_1.east$
4:     $(x_1, y_1) \leftarrow ArbitraryDistant(d.x, d.y, d_1.x, d_1.y, d.r, d_1.r, 0)$
5:     $(x_2, y_2) \leftarrow not\ defined$
6:     **if** d_2 *is not defined* **then**
7:         $d_3 \leftarrow TwoNeighbours(d, d_1)$
8:     **else**
9:         **if** $\sqrt{(d_2.x - d.x)^2 + (d_2.y - d.y)^2} \leq d_2.r + d.r$ **then**
10:             $d_1 \leftarrow d_2, d_2 \leftarrow d_1.east$
11:             $(x_1, y_1) \leftarrow ArbitraryDistant(d.x, d.y, d_1.x, d_1.y, d.r, d_1.r, 0)$
12:         **end if**
13:         $d_3 \leftarrow ThreeNeighbours(d, d_1, d_2)$
14:         **if** d_3 *is not defined* **then**
15:             $d_3 \leftarrow TwoNeighbours(d, d_1)$
16:         **else**
17:             $(x_2, y_2) \leftarrow ArbitraryDistant(d_1.x, d_1.y, d_2.x, d_2.y, d_1.r, d_2.r, 0)$
18:         **end if**
19:     **end if**
20:     $D \leftarrow D \cup \{d_3\}, d \leftarrow d_3$
21:     $(x, y) \leftarrow ArbitraryDistant(x_1, y_1, m, y_{ref}, 0, 0, d.r)$
22:     **if** (x, y) *and* (x_2, y_2) *are defined* **then**
23:         **if** $(x_2 - x)^2 + (y_2 - y)^2 > d.r^2$ **then**
24:             $(x, y) \leftarrow ArbitraryDistant(x_2, y_2, m, y_{ref}, 0, 0, d.r)$
25:         **end if**
26:     **end if**
27:     **if** (x, y) *are defined* **then**
28:         $indication \leftarrow 0$
29:     **else**
30:         $indication \leftarrow -1$
31:     **end if**
32: **end while**
33: **if** (x_2, y_2) *are defined* **then**
34:     $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (m, y_{ref}), (x_2, y_2)\}, 0)$
35: **else**
36:     $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (m, y_{ref})\}, 0)$
37: **end if**
38: $d_1 \leftarrow D[2]$
39: **if** $\sqrt{(d_1.x - d.x)^2 + (d_1.y - d.y)^2} \leq d_1.r + d.r$ **then**
40:     $x_3 \leftarrow m, y_3 \leftarrow d_1.y - \sqrt{d_1.r^2 - (m - d_1.x)^2}$
41:     $(x, y, r, h, b, e, h_{ref}) \leftarrow (d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref})$
42:     $indication \leftarrow EquiDistant(d, \{(x_3, y_3), (m, y_{ref})\}, 1)$
43:     **if** $indication = 0$ **then**
44:         **if** $(x_1 - d.x)^2 + (y_1 - d.y)^2 > d.r^2$ or $((x_2, y_2)$ are defined and $(x_2 - d.x)^2 + (y_2 - d.y)^2) > d.r^2)$ **then**
45:             $(d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref}) \leftarrow (x, y, r, h, b, e, h_{ref})$
46:         **end if**
47:     **end if**
48: **end if**
49: $y_{ref} \leftarrow d.y + \sqrt{d.r^2 - (m - d.x)^2}$

3.4.6. The Region-Filling Algorithm

The algorithm fixes coverage circles for the base stations at $(0, 0)$ and (m, n), as shown in Figure 13, in order to utilize the maximum possible heights for the drones near the base stations. We proceed by filling the circles along the x and y axes. We then move to the inner layer filling to cover the entire region. The detailed procedure is provided in Algorithm 5. The above method uses the minimum number of drones possible.
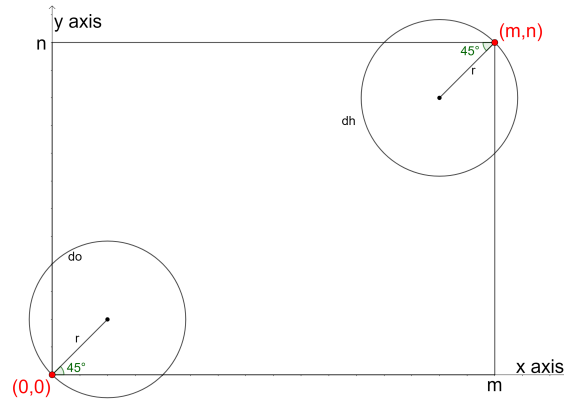
Figure 13. The region-filling algorithm initially places the circles at the two base stations.

Algorithm 5 Region Filling

Input: $m, n, H, SH, \alpha, B, W, C, H_{max}, H_{min}, h_{td}$
Output: *Ordered collection of drone objects D*

1: $D \leftarrow \{\}$, $h \leftarrow \frac{W \times H_{max} - d}{W + \tan(\alpha)}$, $r \leftarrow h \times \tan(\alpha)$
2: $x \leftarrow \frac{r}{\sqrt{2}}, y \leftarrow \frac{r}{\sqrt{2}}$
3: $h_{b_1} \leftarrow H_{0,0}, h_{b_2} \leftarrow H_{m,n}$
4: $(b, h_1, r, e) \leftarrow FindDroneParameters(x, y, h_{b_1}, h)$
5: $x \leftarrow \frac{r}{\sqrt{2}}, y \leftarrow \frac{r}{\sqrt{2}}$
6: $d \leftarrow Drone(x, y, r, h_1, b, e, h_{b_1})$
7: $coordinates \leftarrow \{\}$
8: **while** $(d.x, d.y) \notin coordinates$ **do**
9:      $coordinates \leftarrow coordinates \cup (d.x, d.y)$
10:      $(d.b, d.h, d.r, d.e) \leftarrow FindDroneParameters(d.x, d.y, d.h, d.h_{ref})$
11:      $d.x \leftarrow \frac{r}{\sqrt{2}}, d.y \leftarrow \frac{r}{\sqrt{2}}$
12: **end while**
13: $D \leftarrow D \cup \{d\}$
14: **if** $(d.x - m)^2 + (d.y - n)^2 > d.r^2$ **then**
15:      $(b, h_1, r, e) \leftarrow FindDroneParameters(m - d.x, n - d.y, h_{b_2}, h)$
16:      $x \leftarrow m - \frac{r}{\sqrt{2}}, y \leftarrow n - \frac{r}{\sqrt{2}}$
17:      $d_1 \leftarrow Drone(x, y, r, h_1, b, e, h_{b_2})$
18:      $coordinates \leftarrow \{\}$
19:      **while** $(d_1.x, d_1.y) \notin coordinates$ **do**
20:          $coordinates \leftarrow coordinates \cup \{(d_1.x, d_1.y)\}$
21:          $(d_1.b, d_1.h, d_1.r, d_1.e) \leftarrow FindDroneParameters(d_1.x, d_1.y, d_1.h, d_1.h_{ref})$
22:          $d_1.x \leftarrow m - \frac{d_1.r}{\sqrt{2}}, d_1.y \leftarrow n - \frac{d_1.r}{\sqrt{2}}$
23:      **end while**
24:      $D \leftarrow D \cup \{d_1\}$
25:      $(x_1, y_1) \leftarrow ArbitraryDistant(d.x, d.y, d_1.x, d_1.y, d.r, d_1.r, 0)$
26:      $(x_2, y_2) \leftarrow ArbitraryDistant(d.x, d.y, d_1.x, d_1.y, d.r, d_1.r, 0, 1)$
27:      **if** (x_1, y_1) *are not defined or* , (x_1, y_1) *are defined and* $Maximum(y_1, y_2) < n$ *or* $Minimum(y_1, y_2) > 0$ **then**

28:       $FillAlongXaxis(d)$
29:       $FillAlongYaxis(d)$
30:       **while** *true* **do**
31:           $d_1 \leftarrow d.north, d_2 \leftarrow d.east$
32:           **if** d_1 *or* d_2 *is not defined* **then**
33:               **break**
34:           **end if**
35:           $(x_1, y_1) \leftarrow (x_2, y_2) \leftarrow$ *not defined*
36:           **if** $\sqrt{(d_1.x - d_2.x)^2 + (d_1.y - d_2.y)^2} \leq d_1.r + d_2.r$ **then**
37:               $d_3 \leftarrow TwoNeighbours(d_1, d_2)$
38:               $(x_1, y_1) \leftarrow ArbitraryDistant(d_1.x, d_1.y, d_2.x, d_2.y, d_1.r, d_2.r, 0)$
39:           **else**
40:               $d_3 \leftarrow ThreeNeighbours(d_1, d, d_2)$
41:               **if** d_3 *is not defined* **then**
42:                   $d_4 \leftarrow TwoNeighbours(d_1, d_2)$
43:                   $d_5 \leftarrow TwoNeighbours(d, d_2)$
44:                   $D \leftarrow D \cup \{d_4, d_5\}$
45:                   $InnerXDirectionFilling(d_5)$
46:                   $InnerYDirectionFilling(d_4)$
47:                   **continue**
48:               **else**
49:                   $(x_1, y_1) \leftarrow ArbitraryDistant(d_1.x, d_1.y, d.x, d.y, d_1.r, d.r, 0)$
50:                   $(x_2, y_2) \leftarrow ArbitraryDistant(d.x, d.y, d_2.x, d_2.y, d.r, d_2.r, 0)$
51:               **end if**
52:           **end if**
53:           **if** $(d_3.x - m)^2 + (d_3.y - n)^2 \leq d_3.r_2$ **then**
54:               **break**
55:           **end if**
56:           $D \leftarrow D \cup \{d_3\}$
57:           $d \leftarrow d_3$
58:           $d_1 \leftarrow D\{2\}, d_2 \leftarrow d.south, d_3 \leftarrow d.west$
59:           **if** $(\sqrt{(d_1.x - d.x)^2 + (d_1.y - d.y)^2} \leq d_1.r + d.r)$ *and* $(d_2.x + d_2.r \geq m$ *or* $d_3.y + d_3.r \geq n)$ **then**
60:               **if** $d_2.x + d_2.r \geq m$ **then**
61:                   $x_3 \leftarrow x_4 \leftarrow m$
62:                   $y_3 \leftarrow d_2.y + \sqrt{d_2.r^2 - (m - d_2.x)^2}$
63:                   $y_4 \leftarrow d_1.y + \sqrt{d_1.r^2 - (m - d_1.x)^2}$
64:                   $(x, y, r, h, b, e, h_{ref}) \leftarrow (d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref})$
65:                   $indication \leftarrow EquiDistant(d, \{(x_3, y_3), (x_4, y_4)\}, 1)$
66:                   **if** $indication = 0$ **then**
67:                       **if** $((x_1 - d.x)^2 + (y_1 - d.y)^2 > d.r_2)$ *or* (x_2, y_2) *are defined and* $(x_2 - d.x)^2 + (y_2 - d.y)^2 > d.r_2)$ **then**
68:                           $(d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref}) \leftarrow (x, y, r, h, b, e, h_{ref})$
69:                       **end if**
70:                   **else**
71:                       **break**
72:                   **end if**
73:               **end if**
74:           **else if** $d_3.y + d_3.r \geq n$ **then**
75:               $x_3 \leftarrow d_3.x + \sqrt{d_3.r^2 - (n - d_3.y)^2}$
76:               $x_4 \leftarrow d_1.x + \sqrt{d_1.r^2 - (n - d_1.y)^2}$

77:            $y_3 \leftarrow y_4 \leftarrow n$
78:            $(x, y, r, h, b, e, h_{ref}) \leftarrow (d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref})$
79:            $indication \leftarrow EquiDistant(d, \{(x_3, y_3), (x_4, y_4)\}, 1)$
80:            **if** $indication = 0$ **then**
81:                **if** $((x_1 - d.x)^2 + (y_1 - d.y)^2 > d.r_2)$ or $(x_2, y_2$ are defined and $(x_2 - d.x)^2 + (y_2 - d.y)^2 > d.r_2)$ **then**
82:                    $(d.x, d.y, d.r, d.h, d.b, d.e, d.h_{ref}) \leftarrow (x, y, r, h, b, e, h_{ref})$
83:                **end if**
84:            **else**
85:                **break**
86:            **end if**
87:            **else if** $d_2.x + d_2.r \geq m$ **then**
88:            $x_3 \leftarrow m$
89:            $y_3 \leftarrow d_2.y + \sqrt{d_2.r^2 - (m - d_2.x)^2}$
90:            **if** (x_2, y_2) are defined **then**
91:                $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (x_3, y_3), (x_2, y_2)\}, 0)$
92:            **else**
93:                $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (x_3, y_3)\}, 0)$
94:            **end if**
95:            **if** $indication = 0$ **then**
96:                $InnerYDirectionFilling(d)$
97:                **break**
98:            **end if**
99:            **else if** $d_3.y + d_3.r \geq n$ **then**
100:            $x_3 \leftarrow d_2.x + \sqrt{d_2.r^2 - (n - d_2.y)^2}$
101:            $y_3 \leftarrow n$
102:            **if** (x_2, y_2) are defined **then**
103:                $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (x_3, y_3), (x_2, y_2)\}, 0)$
104:            **else**
105:                $indication \leftarrow EquiDistant(d, \{(x_1, y_1), (x_3, y_3)\}, 0)$
106:            **end if**
107:            **if** $indication = 0$ **then**
108:                $InnerXDirectionFilling(d)$
109:                **break**
110:            **end if**
111:            **end if**
112:            $InnerXDirectionFilling(d)$
113:            $InnerYDirectionFilling(d)$
114:            **if** $(\sqrt{(d_1 - d.x)^2 + (d_1.y - d.y)^2} \leq d_1.r + d.r)$ or $(d.x + d.r \geq m$ or $d.y + d.r \geq n)$ **then**
115:                **break**
116:            **end if**
117:        **end while**
118:    **end if**
119: **end if**
120: **return** D

3.5. Complexity Analysis of the Region Filling Algorithm

The time complexity of the region-filling algorithm is directly proportional to the number of drones required to cover the region, which is $\approx \dfrac{m \times n}{\pi r^2 - \mathscr{W}}$, where $\mathscr{W}$ is the average wastage (defined as $\mathscr{W} = \frac{Total\ area\ of\ wastage}{Number\ of\ circles}$) and r is the average radius of a circle. Therefore, the time complexity is $T(n) = O(\dfrac{m \times n}{\pi r^2 - \mathscr{W}})$

In the best case,

$$r = H_{max} \times tan\alpha$$

$$\Rightarrow T(n) = O(\frac{m \times n}{\pi(H_{max} \times tan\alpha)^2 - \mathcal{W}})$$

In the worst case,

$$r = H_{min} \times tan\alpha$$

$$\Rightarrow T(n) = O(\frac{m \times n}{\pi(H_{min} \times tan\alpha)^2 - \mathcal{W}})$$

In the average case,

$$r = (\frac{H_{max} + H_{min}}{2}) \times tan\alpha$$

$$\Rightarrow T(n) = O(\frac{m \times n}{\pi((\frac{H_{max} + H_{min}}{2}) \times tan\alpha)^2 - \mathcal{W}})$$

where

- α is the coverage angle
- H_{max} is the maximum possible altitude
- H_{min} is the minimum possible altitude

3.6. Proof That Our Method Uses the Minimum Number of Drones

To prove that the region-filling algorithm uses the minimum number of drones to cover the whole region, we show that the various stages in the algorithm ensure the minimum waste of coverage, and hence the minimum number of drones. The minimum wastage happens when the wastage is distributed equally in all directions.

3.6.1. Deployment of the Initial Drone

A straightforward idea in placing the first (or last) drone is that its coverage boundary should pass through the origin. In order to attain minimum wastage, we use the idea of dividing wastage equally along both axes. Consider a circle with a center C and radius r, as in Figure 14:

$$\angle ACB = \angle BCD = 90°$$
$$Area(Segment AB) = r^2(\frac{\theta\pi}{360} - \frac{sin\theta}{2})$$
$$= Area(Segment BD)$$

hence, the wastage is equally distributed along both axes.
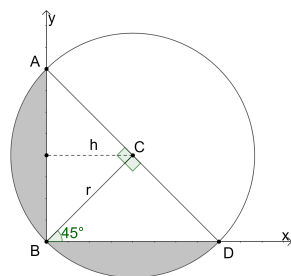
Figure 14. Placing the first drone at C.

3.6.2. Wastage of Coverage in *FillAlongXAxis* and *FillAlongYAxis*

As given in the methods *FillAlongXAxis* and *FillAlongYAxis*, the radii of the different circles filling the rectangular region are different when deploying drones along the *x* and *y* axes. The boundaries of the rectangular region are then $(0, 0)$ and (m, n).

Consider a circle with a center E and radius r_1, as in Figure 15. We want to find the center C such that the wastage is distributed as equally as possible. For this, we find a point C_1 which is $r_1 + r$ distant from E and at an r vertical distance from the *x*-axis. Let B be $(x_1, 0)$ and C_1 be (x_2, r); now, C can be calculated as $x = x_1 + r\cos\theta$ and $y = r\sin\theta$. To prove the efficiency of this approach, we can consider $r_1 = r$, meaning that E is $(\frac{r}{\sqrt{2}}, \frac{r}{\sqrt{2}})$. Therefore,

$$(x_1 - \frac{r}{\sqrt{2}})^2 + (\frac{r}{\sqrt{2}})^2 = r^2$$
$$\Rightarrow x_1 = \sqrt{2}r \tag{8}$$

The distance between E and C_1 is $r_1 + r = 2r$. Therefore,

$$(x_2 - \frac{r}{\sqrt{2}})^2 + (r - \frac{r}{\sqrt{2}})^2 = 4r^2$$
$$\Rightarrow x_2 = r(\frac{1}{\sqrt{2}} + \sqrt{4 - (1 - \frac{1}{\sqrt{2}})^2})$$
$$\theta = tan^{-1}(\frac{y_2 - y_1}{x_2 - x_1})$$
$$= tan^{-1}(\frac{r}{r(\sqrt{4 - (1 - \frac{1}{\sqrt{2}})^2}) - \frac{1}{\sqrt{2}}})$$
$$\simeq 37.008°$$
$$\angle ACB = 45° + \theta = 82.008° (ACBE \text{ is a rhombus})$$
$$\angle BCD = 180° - 2\theta = 105.984° (\triangle BCD \text{ is an isosceles triangle})$$

$$2(Area(Segment AB)) = 2(r^2(\frac{\angle ACB\pi}{360°} - \frac{sin(\angle ACB)}{2}))$$
$$\simeq 0.46r^2$$
$$Area(Segment BD) = r^2(\frac{\angle BCD\pi}{360°} - \frac{sin(\angle BCD)}{2})$$
$$\simeq 0.42r^2$$
$$\Rightarrow 2(Area(Segment AB)) \simeq Area(Segment BD)$$

Here, the wastage of coverage (between the *x*-axis and the circles and between two circles) is distributed almost equally. A similar proof can be shown for the wastage with respect to the *y*-axis.
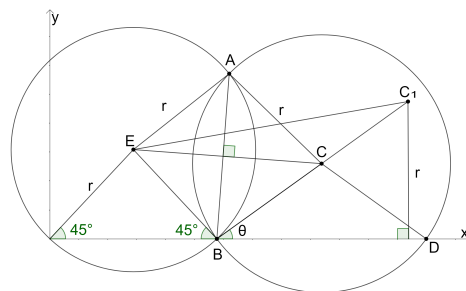
Figure 15. Placing drones along the *x*-axis.

3.6.3. Wastage of Coverage in *TwoNeighbours*

Considering two circles with centers C_1 and C_2 and radii r_1 and r_2, we want to find the center C and radius r of the third circle such that the wastage is distributed among the two circles proportionally to their areas. To this end, our approach is to find a point C_3 which is $r_1 + r$ distant from C_1 and $r_2 + r$ distant from C_2, as in Figure 16a.

(a)	(b)
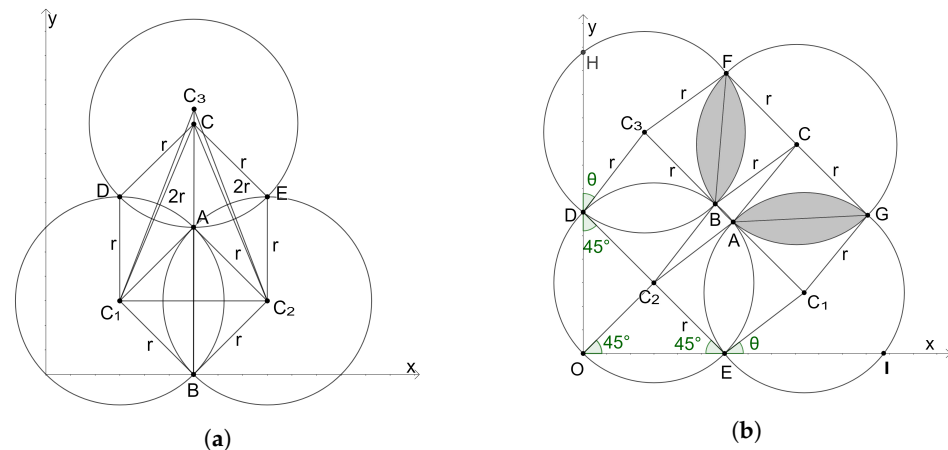
Figure 16. Wastage of coverage in (**a**) *TwoNeighbours* and (**b**) *ThreeNeighbours*.

Let A be the intersecting point of circles with centers C_1 and C_2 near to point C_3, and let $r_1 = r_2 =$ r. The locus of all points equidistant at an arbitrary distance from points C_1 and C_2 is then a straight line passing through the point of the line joining C_1 and C_2. Thus, $\overline{CAB}$ is a straight line. Here, AC_2BC_1, CDC_1A, and CEC_2A are each in the shape of a rhombus, while $\triangle CAC_1$ and $\triangle CAC_2$ are congruent.

$$\angle C_1CA = \angle C_2AC \qquad\qquad\qquad\qquad \text{(CPCT)}$$
$$\angle ACD = 2\angle C_1CA = 2\angle C_2CA$$
$$= \angle ACE = \theta$$
$$Area(Segment\,AD) = r^2\left(\frac{\pi\theta}{360} - \frac{sin\theta}{2}\right)$$
$$= (Area(Segment\,AE))$$
$$2(Area(Segment\,AD)) = 2(Area(Segment\,AE))$$

Hence, the wastage is equally distributed among the two neighbours.

3.6.4. Wastage of Coverage in *ThreeNeighbours*

Considering three circles with centers C_1, C_2, and C_3 and radii r_1, r_2, and r_3, respectively, we want to find the center of the next circle C with radius r. Let A be the intersecting point of circles with centers C_1 and C_2 and let B be that of circles with centers C_2 and C_3, as shown in Figure 16b; furthermore, let $r_1 = r_2 = r_3 = r$.

$$\angle DC_2O = \angle EC_2O = 90° \Rightarrow \overline{DC_2E} \text{ is a straight line.}$$

it is evident that

$$\overline{DC_2} \parallel \overline{C_2E} \parallel \overline{C_3B} \parallel \overline{AC_1} \text{ and } \overline{C_3BAC_1} \text{ is a straight line.}$$

$$\angle C_3 DH = \angle C_1 EI = \theta$$
$$\angle C_3 DC_2 = \angle C_2 EC_1 = 135° - \theta$$
$$2\angle C_3 DC_2 + 2\angle DC_3 B = 360°$$
$$\angle DC_3 B = \frac{360° - 2(135° - \theta)}{2} = 45° + \theta$$
$$\angle C_2 BA = \angle DC_3 B$$
$$\text{(Corresponding angles of } \overline{DC_3} \& \overline{C_2 B})$$
$$\angle ABC = \angle C_2 BA \quad \text{(Property of rhombus } C_2 BCA)$$
$$\angle BC_3 F = \angle ABC$$
$$\text{(Corresponding angles of } \overline{BC} \& \overline{C_3 F})$$
$$= 45° + \theta$$
$$2(Area(segmentBF)) = 2r^2 \left(\frac{(45° + \theta)\pi}{360°} - \frac{sin(45° + \theta)}{2} \right)$$

Similarly,

$$2(Area(segmentGA)) = 2r^2 \left(\frac{(45° + \theta)\pi}{360°} - \frac{sin(45° + \theta)}{2} \right)$$
$$2(Area(segmentBF)) = 2(Area(segmentGA))$$

hence, the wastage of the area of the circle centered at C is equally distributed among those centered at C_1 and C_3.

3.6.5. Wastage of Coverage in *InnerXDirectionFilling* and *InnerYDirectionFilling*

Both *InnerXDirectionFilling* and *InnerYDirectionFilling* make use of *ThreeNeighbours* and *TwoNeighbours*. The last circle's center along each round of *InnerXDirectionFilling* is set to be equidistant from (m, y_{ref}) and another reference point of the circle. Similarly, for *InnerYDirectionFilling*, the last circle's center is set to be equidistant from (x_{ref}, n) and another reference point of the circle. The reason for this is to focus on the circular area within the target region. Because *TwoNeighbours* and *ThreeNeighbours* are proven to equally distribute the wastage area, their combination minimizes the required number of drones.

3.6.6. The Region Filling-Algorithm Minimizes the Number of Drones Required

After filling the first layer, the starting circle of the next layer is filled using *ThreeNeighbours*, followed by filling along the inner x-direction and y-direction. This process continues until the stopping condition is met. As it is proven that the procedures used by the region-filling algorithm distribute wastage of coverage equally and minimize the required number of drones, the combination of these procedures results in the minimum number of drones.

4. Experiments and Results

We conducted various experiments using the terrain data of the landslide-affected region of Pettimudi, Kerala, India, for drone deployment to establish communication. We considered homogeneous drones with similar parameters. The various parameters we used (environmental and drone specific) are shown in Tables 2 and 3.

We fix H_{max} as 2 km. The disaster affected area of Pettimudi is obtained from Google Maps. We consider base stations at diagonally opposite corners, with the first at (9.9913888888888, 76.9316666666667) and the second at (10.0275000000002, 76.96833333333339). We fetch the altitude data of the region using the Google Elevation API. The DMS format is first converted to DD format, then the coordinates are converted to UTM. The altitudes from the API are measured from sea level. To convert the altitude data to heights with reference to the lowest point in the terrain under consideration, we subtract the altitude of the lowest point from each point in the altitude data in order to obtain the normalized altitudes, which are used to populate

the data structure *HeightMatrix (H)*. We use the altitude of the nearest point in the matrix for missing altitude data of any point.

Table 2. Various environmental parameters for the Pettimudi region used in the experiments.

Parameter	Value
Length	4 km
Width	4 km
Normalized H_{min}	0.185 km
Height threshold	0.05 km
Air density	1.293 kg/m^3

Table 3. Various parameters of the drones used for the experiments.

Parameter	Value
Coverage angle	30°
Weight	5 kg
Diameter (distance between two opposite rotors)	0.4 m
Battery Capacity	20,000 mAh
Battery voltage	22.2 V
Battery energy	444 Wh
Battery efficiency	0.9

We apply the region-filling algorithm for the area and obtain the coverage and drone positions, as shown in Figure 17. It is found that sixteen drones are needed to cover the region effectively. The various parameters, specific to Pettimudi, found are shown in Table A1 of Appendix A.
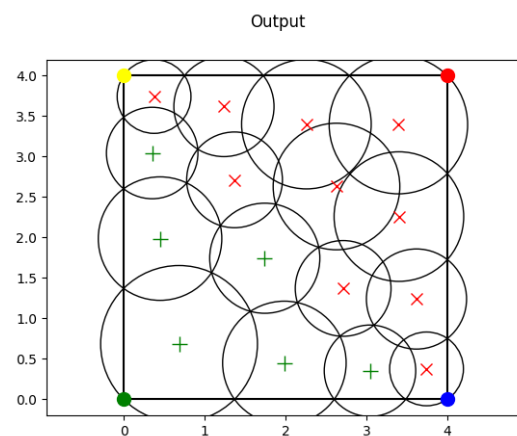
Figure 17. Coverage of drones for the Pettimudi region as obtained by the region-filling algorithm.

4.1. Analysis of Results

In this section, we show how the double optimization problem (minimizing the number of drones and maximizing the average leftover energy) is addressed. We plot the relation between various drone parameters, as in Figure 18.
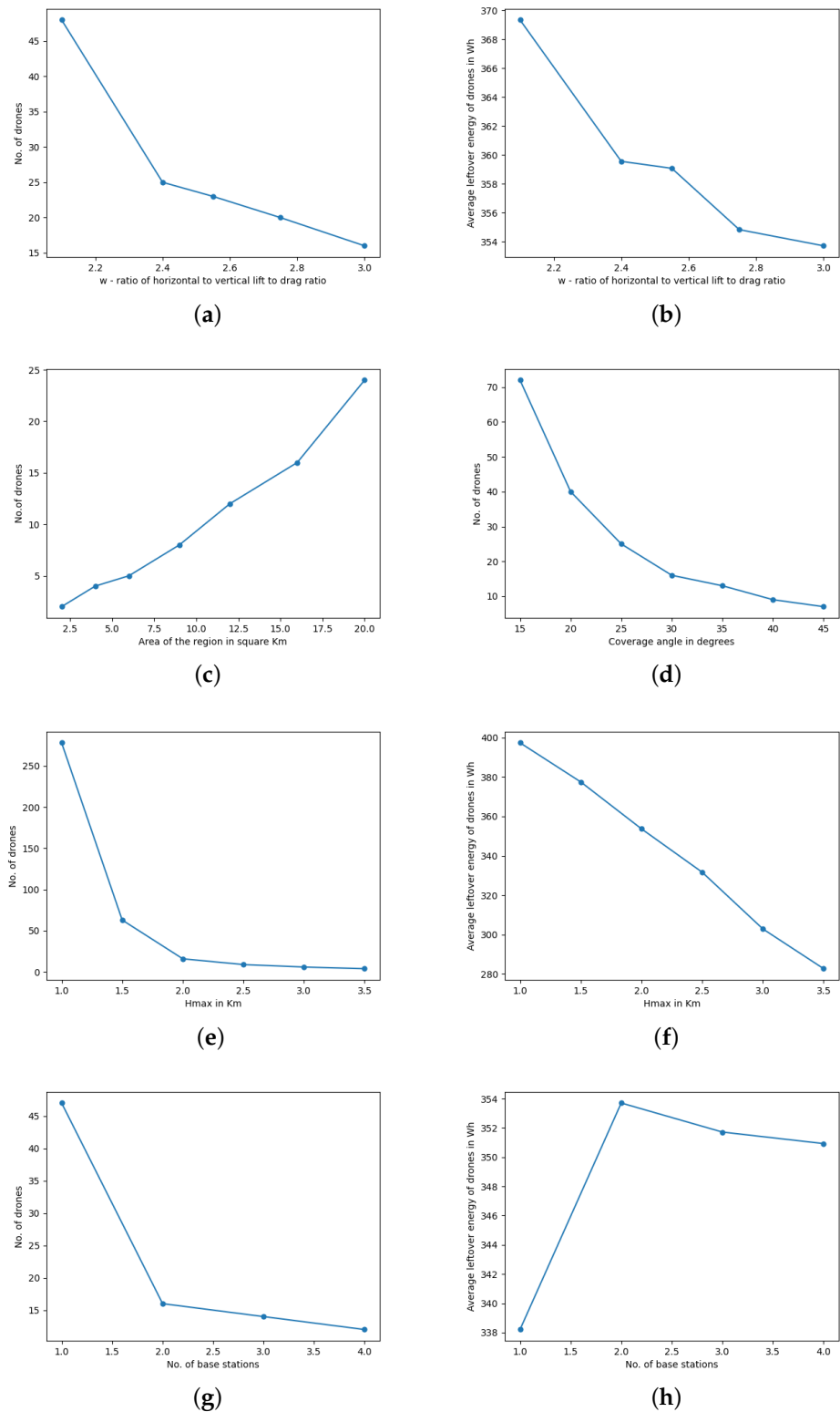
Figure 18. Analysis of various drone parameters: (**a**) number of drones required vs. *W*; (**b**) *W* vs. average leftover energy; (**c**) area vs. number of drones required; (**d**) coverage angle vs. number of drones required; (**e**) H_{max} vs. number of drones required; (**f**) H_{max} vs. average leftover energy; (**g**) number of base stations vs. number of drones required; (**h**) number of base stations vs. average leftover energy.

4.1.1. Ratio of Horizontal Lift to Drag Ratio to Vertical Lift to Drag Ratio (W)

We attempt to obtain the change in the number of drones required for covering the area with varying values of ratio of horizontal lift to drag ratio to vertical lift to drag ratio (W). Our experiments show the graph as in Figure 18a. We obtain another plot of W vs. average leftover energy, shown in Figure 18b. We observe that the number of drones required decreases with an increase in the value of W. However, the average leftover energy decreases with increasing W. Hence, choosing a value of W is a double optimization problem, and the result can be obtained using the *elbow method*. A value of $W = 2.4$ is the most optimal in our experiments.

4.1.2. Area of the Region

It is interesting to see how the area of the region affects the number of drones required. Our experiments yield the plot shown in Figure 18c. The graph shows a linear relationship with the parameters.

4.1.3. Coverage Angle

The coverage angle (α) is a parameter of the drone that determines the area where a drone can facilitate effective communication. Our analysis shows that an increase in coverage angle reduces the number of drones required. However, this relationship is only sublinear, as shown in Figure 18d.

4.1.4. Maximum Height (H_{Max})

H_{max} is the maximum attainable height for a drone with sufficient network strength. The graph in Figure 18e suggests that the number of drones decreases exponentially with increasing H_{max}, while the plot in Figure 18f shows the linear relationship between H_{max} and the average leftover energy. These two plots again lead to a double optimization and, in our case, the optimal value for H_{max} is 2 km.

4.1.5. Number of Base Stations

For a rectangular disaster affected area, the base stations can be placed only in the corners, as ground-level access inside the terrain may not be possible. Therefore, the number of base stations can be one, two, three, or four. Figure 18g shows the relationship between the number of drones required to cover the region and the number of base stations used. From the graph, we can see that the number of drones required decreases with an increasing number of base stations. We then compare the average leftover energy with the number of base stations; the graph is shown in Figure 18h. The curves in these two plots lead towards the solution of a double optimization problem (maximizing the average leftover energy while at the same time minimizing the number of drones required), and in our case, a value of two for the number of base stations is found to be optimal.

We further consider the corners at which the base stations can be placed. They can be placed either in adjacent corners or diagonally opposite to each other. As explained in the description of the region-filling algorithm, the first base station is always placed at (0, 0). A plot showing H_{max} vs. the number of drones required when the second base station is placed at various corners is shown in Figure 19a. A similar plot for the average leftover energy is shown in Figure 19b. These plots suggest that placing the second base station diagonally opposite to the first yields the maximum efficiency.

(**a**)   (**b**)
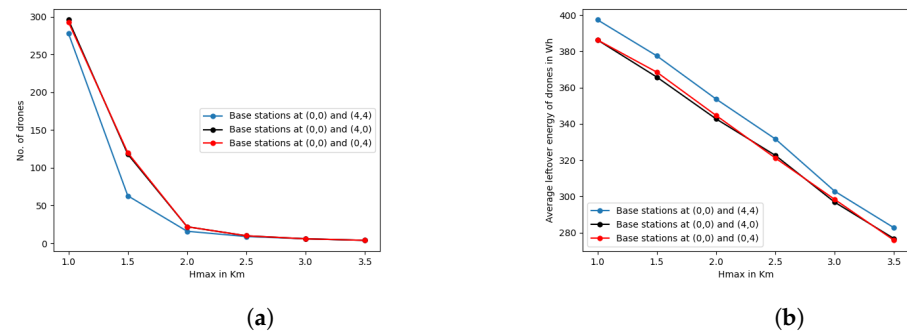
Figure 19. The effect of placing the second base station at different corners: (**a**) H_{max} vs. number of drones required and (**b**) H_{max} vs. average leftover energy.

4.2. Simulation

We simulated drone deployment for the Pettimudi region using Matlab and OMNet++ [44]. OMNeT++ is a discrete event simulator mainly used for network simulation. We used the *INET* Framework in OMNet++ to simulate the wireless networks.

The terrain data of Pettimudi region (4 km × 4 km) were obtained using the Google Elevation API. A simulation of the terrain is shown in Figure 20a. After construction of the Height Matrix data structure, the region-filling algorithm produces the positions of the sixteen drones deployed to cover the entire area. The drones are deployed using two base stations placed at two diagonally opposite corners of the rectangular region. The trajectories of the drones deployed from the base stations is shown in Figure 20b. After the drones reach their specified coordinates based on the output from the region-filling algorithm, the communications network is established. Here, we take 30° as the coverage angle (α). The network coverage obtained is shown in Figure 20b. Figure 21a shows the cone-shaped network coverage of the sixteen deployed drones. Figure 21b shows a top view of the network coverage. It can be observed that the region is fully covered, with no voids in between. The altitudes at which different drones are placed is shown in Figure A1 of Appendix A.

(**a**)   (**b**)
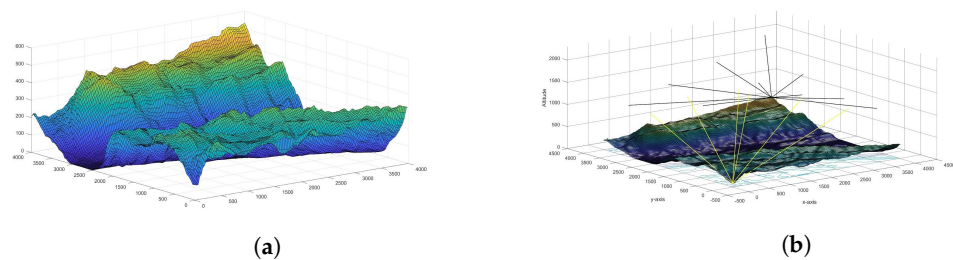
Figure 20. Positions of drones for Pettimudi region: (**a**) terrain simulation of Pettimudi and (**b**) trajectories of the sixteen drones deployed from two base stations placed diagonally opposite to one another.

The simulation process proceeds as follows. The terrain map of *Pettimudi* is generated using the *OsgEarth* package provided in OMNeT++. The entire 4 km x 4 km is divided into sixteen equal-area rectangles. A cell phone is placed at each of these rectangles and each cell phone is constrained to move inside these rectangles randomly to cover different points in the region. During these movements, the cell phone can send a UDP packet to the nearest drone host. Our simulation shows an average network efficiency (percentage ratio of the number of packets received to the number of packets sent) of 98%.

The simulation videos are available in supplementary materials. The simulation results confirm that the terrain-aware nature of the region-filling algorithm ensures that every cell phone inside the region is covered by the network.
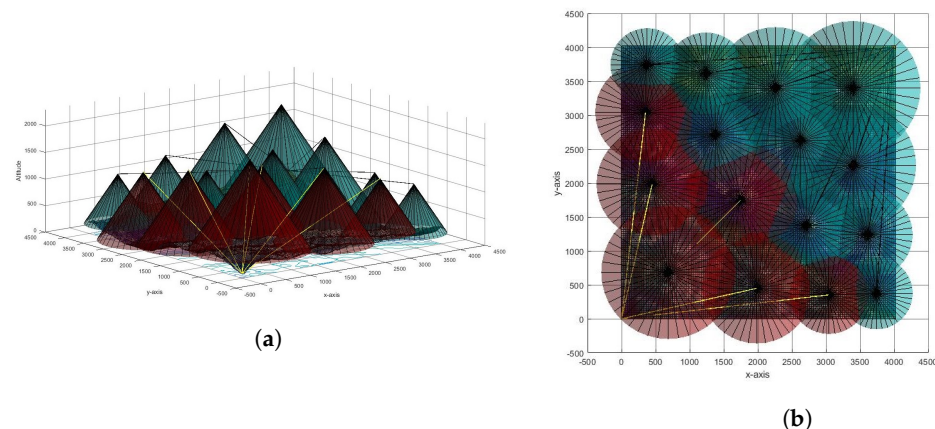
Figure 21. View of the network coverage obtained in simulation for a coverage angle $\alpha = 30°$: (**a**) network coverage front view and (**b**) network coverage top view.

5. Discussions

The strategy of establishing a communications network without any voids by deploying drones in disaster affected areas at optimal heights with a view to minimizing the number of drones and maximizing the average leftover energy is a unique approach. In this section, we compare our approach with other methods available in the literature which are closely related.

Ullah et al. [8] presented an idea for a bush fire mitigation system using drones, specifically, for the 2019–2020 bush fires in New South Wales, Australia. Using a swarm of drones to monitor the bush fire, they developed a method based on PSO to obtain the shortest path a drone has to travel to the target area, thereby ensuring that it uses the maximum hovering time and the minimum travel time. The drone travels back to the sink station after the monitoring is over. Here, our objective is different, as we want to continuously provide communication facilities until the rescue operation is over. Furthermore, we have to ensure that every single point in the affected region is under network coverage, which cannot be guaranteed by the above approach.

Kurt et al. [45] proposed the idea of establishing in-network connectivity between drones used for intelligent traffic management and damage assessment in post-disaster scenarios. They use a connectivity maintenance heuristic that causes changes in the formation of drones in response to the service requests. It focuses on a planar surface, whereas in our case the algorithm considers terrain types with different obstacles at different heights. Furthermore, the above paper does not pay very much attention to energy conservation, while our method focuses on maximizing the average leftover energy by placing the drones at different heights. Moreover, the above method concentrates on connectivity maintenance between drones, while our technique relies on quick establishment of communications between the base stations and end users, with the drones in between, which ensures that there are no network voids in the disaster-struck terrain.

6. Conclusions

Natural calamities such as floods, earthquakes, and landslides are inevitable, and can cause severe damage to human life. With the advent of new technologies, sophisticated mechanisms are available for rescue operations in such situations. One of the major threats that rescue teams face is damage to communication infrastructure, which hinders the progress of rescue due to the difficulties faced in coordination. The easiest method of communication in the present era is by cell phone. However, cell phones are unusable if the base stations providing network coverage are damaged in the calamity. An easy way to establish network coverage for cell phone communications is by hosting base stations in drones. However, drones have limited battery life and may require a substantial amount of energy to fly to their destinations. They have to frequently return to the nearest base

station to be recharged, then fly back to their destination. This may introduce voids in network coverage, making rescue operations inefficient. In this paper, we introduce a novel algorithm (the region-filling algorithm) for deploying drones in disaster-affected areas in a way that ensures that there are no voids in coverage and that maximum average leftover energy is available for the network. In addition, we ensure that the algorithm finds a deployment pattern that uses the minimum possible number of drones. We selected *Pettimudi*, India, a disaster affected location, for simulation experiments to find the drone deployment pattern for the region based on our algorithm. Our experiments show that the algorithm is able to find a drone deployment pattern that provides coverage to the entire area. Furthermore, our approach ensures that the maximum average leftover energy is available for the drones via deployment at different heights while taking the local terrain into consideration. We prove our claim that this approach uses the minimum number of drones, and present a simulation of our algorithm using Matlab and *OMNet++*.

7. Future Work

Our future work will include experiments with real hardware drones instead of simulations. In addition, we intend to incorporate the following areas into our research:

- Here, we have considered only perfect circular-shaped coverage for drones; we intend to engage in a detailed exploration of other shapes as well.
- The present work concentrates on the algorithm used to fill the area without voids using stationary drones. We intend to study the effects of achieving the same using moving drones. In addition, we may explore different technologies available for establishing and maintaining communication between the drones and base stations.
- A detailed comparison between the region-filling algorithm and other network coverage management methods should be explored.

Finally, we intend to optimize the region-filling algorithm by finding the best specific terrain locations for the base stations.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/computers11090139/s1, Video S1: 3D simulation of drone deployment in Pettimudi, created in MatLab, Video S2: Simulation of drone deployment in Pettimudi, created in OmNet++.

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

Funding: This research was funded by TEQIP Phase II of the Rajiv Gandhi Instituite of Technology, Kottayam, India.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank TEQIP Phase II of the Rajiv Gandhi Institute of Technology, Kottayam, India, for providing financial assistance for this research.

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

Appendix A. Data on Pettimudi

Table A1 shows the positions, base stations, coverage, leftover energy, and hovering time for each drone for the Pettimudi region, found using the region-filling algorithm. Figure A1 shows the simulation of drones placed at different altitudes and their coverage,

highlighting that the drones placed near the base stations are located at greater heights in order to maintain the leftover energy of all the drones as approximately equal.

Table A1. Actual drone positions and other parameters for the Pettimudi region as obtained by the region-filling algorithm.

Cartesian Coordinates (*x*, *y*) and Normalized Heights in km	Real World Coordinates (*x*, *y*) and Heights from the Sea Level in km	Radius of Coverage in km	Base Station	Leftover Energy in Wh	Hovering Time in Minutes
(0.6850012909668802, 0.6850012909668802, 1.9810872946905196)	(9.9975445782069, 76.93794986634965, 2.52708729469052)	1.144338031	0	339.294	23.38707962
(1.9863072120622192, 0.450208730417653, 1.6521173291631104)	(9.99535120808547, 76.94980395247727, 2.1981173291631104)	0.954314682	0	338.01825	23.13270969
(3.0440939462526897, 0.35293290321067433, 1.3095049307901139)	(9.994416966351103, 76.9594446404037, 1.855504930790114)	0.756411037	0	338.01825	23.13270969
(3.742788025157351, 0.3762679343689235, 1.0910503229335835)	(9.994590335836122, 76.96581724258051, 1.6370503229335835)	0.630224818	1	360.41475	27.59831517
(0.450208730417653, 1.9863072120622192, 1.5481173291631105)	(10.009320044652764, 76.93587672010516, 2.0941173291631108)	0.894241027	0	342.93225	24.11250498
(0.35293290321067433, 3.0440939462526897, 1.3035049307901139)	(10.01888693185655, 76.93504800175259, 1.849504930790114)	0.752945249	0	338.30175	23.18923634
(0.3762679343689235, 3.742788025157351, 1.1140503229335836)	(10.025201446811009, 76.93529828116637, 1.6600503229335837)	0.643510337	1	359.328	27.38162967
(1.7439562567843623, 1.7439562567843503, 1.481891136490049)	(10.007060435737923, 76.94766363480579, 2.027891136490049)	0.855986705	0	339.294	23.38707962
(2.7124246028936443, 1.3706564833549857, 1.355107375455865)	(10.003634231219198, 76.95647521198755, 1.901107375455865)	0.782752436	1	359.0445	27.32510302
(3.618532811839651, 1.2396785509335047, 1.146148156903163)	(10.00240158709933, 76.96473106479948, 1.6921481569031631)	0.662051051	1	371.1405	29.73690682
(1.37065648335483, 2.71242460289361, 1.2411073754558135)	(10.015834661655397, 76.94431150115088, 1.7871073754558136)	0.716902468	1	364.431	28.3991094
(1.239678550933715, 3.61853281183974, 1.3511481569032364)	(10.024032260677686, 76.94316581197926, 1.8971481569032365)	0.780465468	1	361.45425	27.80557956
(2.630328456811882, 2.630328456812093, 1.4423306425423421)	(10.015025145572809, 76.95579474007404, 1.9883306425423422)	0.833135326	1	370.52625	29.61443241
(3.4013205248344214, 2.2585005948062418, 1.63315625474712)	(10.011622647234047, 76.96280560529775, 2.17915625474712)	0.943362172	1	363.0135	28.11647614
(2.2585005948064087, 3.4013205248342473, 1.8341562547471535))	(10.02201430185454, 76.95244553872685, 2.3801562547471535)	1.059466064	1	353.51625	26.22283331
(3.395089978241617, 3.395089978241617, 2.0140000000000002)	(10.021896870058129, 76.96281057227907, 2.5600000000000005)	1.16334944	1	360.5490667	27.62509637
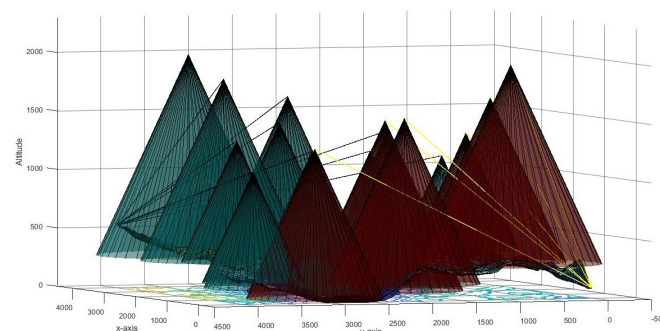
Figure A1. Figure showing the heights at which drones are placed in the simulation.

References

1. Khaled, Z.E.; Mcheick, H. Case studies of communications systems during harsh environments: A review of approaches, weaknesses, and limitations to improve quality of service. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 9960. [CrossRef]
2. Kamruzzaman, M.; Sarkar, N.I.; Gutierrez, J.; Ray, S.K. A study of IoT-based post-disaster management. In Proceedings of the 2017 International Conference on Information Networking (ICOIN), Da Nang, Vietnam, 11–13 January 2017; pp. 406–410. [CrossRef]
3. Lefeuvre, F.; Tanzi, T. About radio science contribution to disaster management. In Proceedings of the 2014 XXXIth URSI General Assembly and Scientific Symposium (URSI GASS), Beijing, China, 16–23 August 2014; pp. 1–4. [CrossRef]
4. Achu, A.L.; Joseph, S.; Aju, C.D.; Mathai, J. Preliminary analysis of a catastrophic landslide event on 6 August 2020 at Pettimudi, Kerala State, India. *Landslides* **2021**, *18*, 1459–1463. [CrossRef]
5. Krishnakumar, G.; Raman, G.K. In Idukki, Living in Fear of Landslides. The Hindu Daily, August 29, 2020. Available online: https://www.thehindu.com/news/national/kerala/in-idukki-living-in-fear-of-landslides/article32468924.ece (accessed on 27 August 2021).
6. Fatal Earthquake Hit the Chinese Province of Sichuan in May 2008 Disaster Management–How Satellite Communications and ND SatCom Technology Supported First Responders. Available online: https://www.ndsatcom.com/gfx/file/solutions/governmentCase_Study_Sichuan_earthquake_China_October_2008.pdf (accessed on 21 June 2021).
7. Xu, J.; Zeng, F.; Liu, W.; Takahashi, T. Damage Detection and Level Classification of Roof Damage after Typhoon Faxai Based on Aerial Photos and Deep Learning. *Appl. Sci.* **2022**, *12*, 4912. [CrossRef]
8. Ullah, F.; Khan, S.I.; Munawar, H.S.; Qadir, Z.; Qayyum, S. UAV Based Spatiotemporal Analysis of the 2019–2020 New South Wales Bushfires. *Sustainability* **2021**, *13*, 207. [CrossRef]
9. Nikhil, N.; Shreyas, S.M.; Vyshnavi, G.; Yadav, S. Unmanned Aerial Vehicles (UAV) in Disaster Management Applications. In Proceedings of the 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 20–22 August 2020; pp. 140–148. [CrossRef]
10. Mototolea, D. A Study On The Actual And Upcoming Drone Communication Systems. In Proceedings of the 2019 International Symposium on Signals, Circuits and Systems (ISSCS), Iasi, Romania, 11–12 July 2019; pp. 1–4. [CrossRef]
11. Liu, W.; Lis, K.; Salzmann, M.; Fua, P. Geometric and Physical Constraints for Drone-Based Head Plane Crowd Density Estimation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 244–249. [CrossRef]
12. Morrison, K.T. Rapidly recovering from the catastrophic loss of a major telecommunications office. *IEEE Commun. Mag.* **2011**, *49*, 28–35. [CrossRef]
13. Rabieekenari, L.; Sayrafian, K.; Baras, J.S. Autonomous relocation strategies for cells on wheels in public safety networks. In Proceedings of the 2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC), IEEE, Las Vegas, NV, USA, 8–11 January 2017; pp. 41–44. [CrossRef]
14. AT&T Network Disaster Recovery. Available online: https://www.business.att.com/solutions/family/network-services/network-disaster-recovery.html (accessed on 27 August 2021).
15. Takahashi, T.; Jeong, B.; Okawa, M.; Akaishi, A.; Asai, T.; Katayama, N.; Akioka, M.; Yoshimura, N.; Toyoshima, M.; Miura, R.; et al. Disaster satellite communication experiments using WINDS and wireless mesh network. In Proceedings of the 2013 16th International Symposium on Wireless Personal Multimedia Communications (WPMC), Atlantic City, NJ, USA, 24–27 June 2013; pp. 1–4.
16. Fotouhi, A.; Ding, M.; Hassan, M. Service on Demand: Drone Base Stations Cruising in the Cellular Network. In Proceedings of the 2017 IEEE Globecom Workshops (GC Wkshps), Singapore, 4–8 December 2017; pp. 1–6. [CrossRef]
17. Park, S.; Kim, K.; Kim, H.; Kim, H. Formation control algorithm of multi-UAV-based network infrastructure. *Appl. Sci.* **2018**, *8*, 1740. [CrossRef]
18. Zhang, X.; Duan, L. Energy-Saving Deployment Algorithms of UAV Swarm for Sustainable Wireless Coverage. *IEEE Trans. Veh. Technol.* **2020**, *69*, 10320–10335. [CrossRef]
19. Ruan, L.; Wang, J.; Chen, J.; Xu, Y.; Yang, Y.; Jiang, H.; Zhang, Y.; Xu, Y. Energy-efficient multi-UAV coverage deployment in UAV networks: A game-theoretic framework. *China Commun.* **2018**, *15*, 194–209. [CrossRef]

20. Kumar, P.; Singh, P.; Darshi, S.; Shailendra, S. Analysis of Drone Assisted Network Coded Cooperation for Next Generation Wireless Network. *IEEE Trans. Mob. Comput.* **2021**, *20*, 93–103. [CrossRef]

21. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Drone-Based Antenna Array for Service Time Minimization in Wireless Networks. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [CrossRef]

22. Namvar, N.; Homaifar, A.; Karimoddini, A.; Maham, B. Heterogeneous UAV Cells: An Effective Resource Allocation Scheme for Maximum Coverage Performance. *IEEE Access* **2019**, *7*, 164708–164719. [CrossRef]

23. Thibbotuwawa, A.; Nielsen, P.; Zbigniew, Z.; Bocewicz, G. Energy Consumption in Unmanned Aerial Vehicles: A Review of Energy Consumption Models and Their Relation to the UAV Routing. In *Proceedings of the 39th International Conference on Information Systems Architecture and Technology–ISAT 2018*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 853.

24. Theys, B.; Schutter, J.D. Forward flight tests of a quadcopter unmanned aerial vehicle with various spherical body diameters. *Int. J. Micro Air Veh.* **2020**, *12*. [CrossRef]

25. Sundaresan, K.; Chai, E.; Chakraborty, A.; Rangarajan, S. SkyLiTE:End-to-end design of low-altitude UAV networks for providing LTE connectivity. *arXiv* **2018**, arXiv:1802.06042.

26. Panda, K.G.; Das, S.; Sen, D.; Arif, W. Design and Deployment of UAV-Aided Post-Disaster Emergency Network. *IEEE Access* **2019**, *7*, 102985–102999. [CrossRef]

27. Li, J.; Lu, D.; Zhang, G.; Tian, J.; Pang, Y. Post-Disaster Unmanned Aerial Vehicle Base Station Deployment Method Based on Artificial Bee Colony Algorithm. *IEEE Access* **2019**, *7*, 168327–168336. [CrossRef]

28. Ishigami, M.; Sugiyama, T. A Novel Drone's Height Control Algorithm for Throughput Optimization in Disaster Resilient Network. *IEEE Trans. Veh. Technol.* **2020**, *69*, 16188–16190. [CrossRef]

29. Shakhatreh, H.; Khreishah, A.; Ji, B. UAVs to the Rescue: Prolonging the Lifetime of Wireless Devices Under Disaster Situations. *IEEE Trans. Green Commun. Netw.* **2019**, *3*, 942–954. [CrossRef]

30. Lyu, J.; Zeng, Y.; Zhang, R.; Lim, T.J. Placement Optimization of UAV-Mounted Mobile Base Stations. *IEEE Commun. Lett.* **2017**, *21*, 604–607. [CrossRef]

31. Plachy, J.; Becvar, Z.; Mach, P.; Marik, R.; Vondra, M. Joint Positioning of Flying Base Stations and Association of Users: Evolutionary-Based Approach. *IEEE Access* **2019**, *7*, 11454–11463. [CrossRef]

32. Ruzicka, M.; Volosin, M.; Gazda, J.; Maksymyuk, T.; Han, L.; Dohler, M. Fast and computationally efficient generative adversarial network algorithm for unmanned aerial vehicle–based network coverage optimization. *Int. J. Distrib. Sens. Netw.* **2022**, *18*. [CrossRef]

33. Furutani, T.; Minami, M. Drones for Disaster Risk Reduction and Crisis Response. In *Emerging Technologies for Disaster Resilience: Practical Cases and Theories*; Sakurai, M., Shaw, R., Eds.; Springer: Singapore, 2021; pp. 51–62. [CrossRef]

34. Chen, Y.; Cheng, W. Performance Analysis of RIS-equipped-UAV Based Emergency Wireless Communications. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Korea, 16–20 May 2022; pp. 255–260. [CrossRef]

35. Mohamed, E.M.; Hashima, S.; Hatano, K. Energy Aware Multiarmed Bandit for Millimeter Wave-Based UAV Mounted RIS Networks. *IEEE Wirel. Commun. Lett.* **2022**, *11*, 1293–1297. [CrossRef]

36. Hashima, S.; Hatano, K.; Mohamed, E.M. Multiagent Multi-Armed Bandit Schemes for Gateway Selection in UAV Networks. In Proceedings of the 2020 IEEE Globecom Workshops (GC Wkshps), IEEE, Taipei, Taiwan, 7–11 December 2020; pp. 1–6. [CrossRef]

37. Belkadi1, A.; Ciarletta, L.; Theilliol, D. Particle swarm optimization method for the control of a fleet of Unmanned Aerial Vehicles. *J. Phys. Conf. Ser.* **2015**, *659*, 012015. [CrossRef]

38. Mesquita, R.; Gaspar, P.D. A Path Planning Optimization Algorithm Based on Particle Swarm Optimization for UAVs for Bird Monitoring and Repelling–Simulation Results. In Proceedings of the 2020 International Conference on Decision Aid Sciences and Application (DASA), Sakheer, Bahrain, 8–9 November 2020; pp. 1144–1148. [CrossRef]

39. Li, M.; Du, W.; Nian, F. An Adaptive Particle Swarm Optimization Algorithm Based on Directed Weighted Complex Network. *Math. Probl. Eng.* **2014**, *9*, 2232. [CrossRef]

40. Sewak, M.; Sahay, S.K.; Rathore, H. Comparison of Deep Learning and the Classical Machine Learning Algorithm for the Malware Detection. In Proceedings of the 2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Busan, Korea, 27–29 June 2018; pp. 293–296. [CrossRef]

41. Latitude, Longitude and Coordinate System Grids. Available online: https://gisgeography.com/latitude-longitude-coordinates/ (accessed on 20 June 2021)

42. Janssen, V. Understanding coordinate reference systems, datums and transformations. *Int. J. Geoinformatics* **2009**, *5*, 41–53.

43. Ostrovsky, R.; Rabani, Y.; Schulman, L.J.; Swamy, C. The Effectiveness of Lloyd-Type Methods for the k-Means Problem. In Proceedings of the 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), IEEE, Berkeley, CA, USA, 21–24 October 2006; pp. 165–176. [CrossRef]

44. Varga, A.; Horning, A. An overview of the OMNeT++ simulation environment. In Proceedings of the Simutools'08–1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Brussels, Belgium, 3–7 March 2008; Volume 60, pp. 1–10.

45. Kurt, A.; Saputro, N.; Akkaya, K.; Uluagac, A.S. Distributed Connectivity Maintenance in Swarm of Drones During Post-Disaster Transportation Applications. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 6061–6073. [CrossRef]