Article

UVCS: Unit Virtual Coordinate System for UAV Intra-Swarm Routing in GPS-Denied Environment

Yuliya Gaidamaka [1,2,*] and Konstantin Samouylov [1,2]

[1] Applied Probability and Informatics Department, Peoples' Friendship University of Russia (RUDN University), 6 Miklukho-Maklaya St., 117198 Moscow, Russia
[2] Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences (FRC CSC RAS), Institute of Informatics Problems, 44-2 Vavilov St., 119333 Moscow, Russia
* Correspondence: gaydamaka-yuv@rudn.ru

Abstract: Wireless ad hoc networks are the solution for providing network connectivity in challenging environments with a lack or absence of infrastructure. Data transmission in such networks typically adopts geographic routing protocols, which use geographic coordinates as addresses of network devices. However, geographic coordinates are not always obtainable, as traditional localization systems (GNSS, Wi-Fi, terrestrial infrastructure) might not be available due to signal loss. In this paper, we propose a method that assigns virtual coordinates to network nodes, which can be used as input for geographic routing protocols. The numerical results demonstrate the high topological similarity between the physical and the virtual network. Our method exhibits convergence advantages over conventional approaches and outperforms them in terms of the total number of discovered paths.

Keywords: UAV swarm, drone fleet, topology, machine-type communications, queuing network, graph theory, simulation

MSC: 68R10

check for updates

Citation: Gaidamaka, Y.; Samouylov, K. UVCS: Unit Virtual Coordinate System for UAV Intra-Swarm Routing in GPS-Denied Environment. *Mathematics* **2023**, *11*, 694. https://doi.org/10.3390/math11030694

Academic Editor: János Sztrik

Received: 22 December 2022
Revised: 16 January 2023
Accepted: 20 January 2023
Published: 30 January 2023

1. Introduction

The attention of equipment manufacturers and carriers is now turned to future networks with terabit data rates. Such networks are expected to mark a groundbreaking shift from the services of yesterday, such as the Internet of Things [1], to those of the future, such as the Internet of Everything (IoE) [2,3]. One IoE scenario which attracts increasing attention these days is an intelligent autonomous system that performs certain missions without human involvement. Such systems include autonomous UAVs fleets (or drones swarms, drones fleets) [4,5] capable of solving tasks and making decisions independently without interaction with the control center. This is especially relevant when the standard positioning technologies (Global Navigation Satellite System (GNSS), base stations, Wi-Fi) are not available [6].

Lack of positioning information from external sources can be caused by many reasons. There are areas where infrastructure deployment is challenging (mountainous terrain), and sometimes economically unprofitable (remote areas). In some areas, there is no network coverage due to natural reflectors (forests, caves). Nevertheless, some scenarios require communications to be maintained under any conditions. During search and rescue operations in remote and inaccessible places, it is crucial not only to keep a drone swarm connected but also to transmit information quickly and efficiently [7]. Another scenario could include the use of swarms indoors, for example, inside large shopping malls, to identify and eliminate the causes of local disasters. Indoors, the GPS signal is typically absent due to shielding, and additional sources of drone positioning information, such as the cellular network, may also be partly unavailable (e.g., due to an emergency situation).

There are many aspects to consider when designing an intelligent autonomous drone swarm system: control and management of the swarm, its efficiency, and performance of the distributed computations. Nevertheless, one of the main challenges is to maintain the connectivity of an autonomous drone swarm [8]. Connectivity is needed (i) to maintain swarm integrity (if any drone is lost, the swarm loses its structure, resulting in reduced functionality) (ii) to traverse various complex objects (e.g., tree crowns), and (iii) to perform distributed or a centralized computation within the swarm and exchange information between drones. Thus, it is necessary to ensure the efficient routing of information within the swarm.

As it was mentioned before, the support of UAVs (or drones) connectivity is challenging. One of the solutions is wireless ad hoc networks—decentralized wireless networks with no permanent structure [9]. Due to the simplicity and low cost of deployment, wireless ad hoc networks perfectly fit many applications in medical, intelligent transport, and other fields. Among all the possible use cases, UAV ad hoc networks remain the most promising one—for instance, the specifics of ad hoc networks perfectly suit the purposes of search and rescue operations [10].

Traditional routing protocols based on the awareness of each network node about the location of all other nodes require regular exchange of signaling information between all network nodes and are inefficient in these conditions due to excessive overheads. Alternatively, one might use data about the local position of devices obtained from external sources for routing in ad hoc networks. Geographical routing is an example of a protocol that uses the geographical coordinates of the device to deliver data. However, as mentioned above, external information may not always be available.

In this paper, we propose an iterative algorithm for positioning nodes in the developed Unit Virtual Coordinate System (UVCS), which allows, using the information available in the node only about the presence of neighbors in a given radius and the distance to each of them, to determine its coordinates in the UVCS. Based on the UVCS, it becomes possible to apply one of the geographic routing algorithms to build a route to any network node, without even knowing the direction to the destination node. The developed approach, which combines the UVCS system and a geographic routing algorithm, allows maintainenance self-organization and topology to ensure the connectivity of a drones swarm.

The iterative algorithm for positioning nodes in the UVCS system works on the principle of fast gradient descent and minimizes the difference between the known estimate of the distance to a neighboring node and the distance in the UVCS system, averaged over all neighboring nodes.

The main contributions of our study are as follows:

- we propose an approach for self-organization and topology support in drone swarms in the complete absence of external positioning information using the developed Unit Virtual Coordinate System;
- we propose metrics to analyze the characteristics of intra-swarm routing on top of the developed Unit Virtual Coordinate System;
- we carry out a comparative analysis of the characteristics of self-organization according to the topology formed by the developed algorithm, with the topology obtained with complete knowledge of the network. Methods of queuing theory are used to carry out the analysis.

The rest of the paper is organized as follows. Section 2 reviews the current state of the problem of connectivity support and routing in autonomous swarms. In Section 3, we introduce an algorithm for assigning virtual coordinates based on the locally available knowledge. In Section 4, we specify the mathematical model. Section 5 introduces the metrics of interest. The numerical results are presented in Section 6, and conclusions are drawn in the last section.

2. Related Work

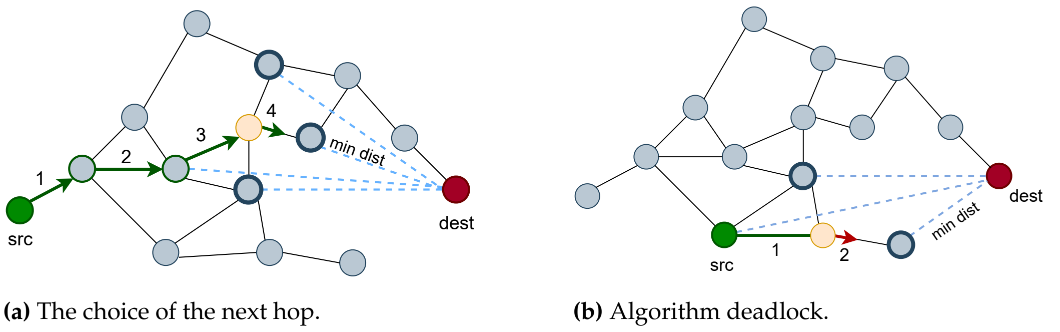
With the emergence of the Internet of Things (IoT), devices have been firmly embedded in daily lives. For simplicity of deployment and operation of such large decentralized systems, it is convenient to combine devices into a single distributed network, which can be controlled remotely. Wireless self-organizing dynamic networks utilized for this task are a particular case of well-known ad hoc networks.

An ad hoc network is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration [11]. A distinguishing feature of such networks is that each node acts as both an end system and a router. Ad hoc networks can be deployed in territories with no infrastructure, for example, in scenarios such as disaster relief. The topology of ad hoc networks changes much more frequently as compared to wired networks. The routing protocol must ensure network scalability, robustness [12], and energy efficiency [13].

Over the past decades, many algorithms have been proposed for routing in wireless communication networks [14]. Table-driven routing traditionally implemented in wired networks, despite the reliability of operation and simplicity of configuration, have a number of drawbacks when deployed on wireless ad hoc systems. Specifically, the dynamics of the system topology require frequent updates of the routing tables, sometimes even non-incremental ones. On the other hand, on-demand routing protocols, such as the Ad hoc On-Demand Distance Vector (AODV) routing protocol, introduce large delays required to route discovery prior to information transmission [15].

As an alternative solution to table-driven and on-demand routing, geographic routing protocols can be utilized [16,17]. According to them, packets are forwarded between nodes based on their geographic locations, which are utilized as addresses. At each intermediate node, the process of selecting the next node for forwarding is based on the geographic position of the destination node and the coordinates of the neighboring forwarding candidates. The simplest strategy is to use greedy forwarding and by sending the data packets to the neighbor that is geographically closest to the destination.

Figure 1a shows a scenario where data are transferred from the green node src to the red node dest. Specifically, the next node is chosen based on two characteristics. Firstly, it should be adjacent to the current node. Secondly, the distance between the potential node and the destination node should be minimal. The figure illustrates the choice of the node in the fourth step. The current node, highlighted in yellow, has four adjacent nodes (marked in bold). Comparing the distances between the bold and dest node (lengths of dashed lines) one ends up choosing the node at the minimum distance from the dest.



(a) The choice of the next hop. **(b)** Algorithm deadlock.

Figure 1. Explanation of the Greedy Forwarding Geographic Routing protocol.

However, this method can lead to deadlock situations. Figure 1b illustrates the case when packets are forwarded to a node without neighbors apart from the current node. In the second step the current yellow node choses the node that is at the shortest distance from the dest red node but has no neighbors. In such cases, alternative forwarding strategies such as an Adaptive Greedy-compass Energy-aware Multipath protocol (AGEM) [18] can be utilized.

Geographic routing algorithms offer a number of advantages, such as simplicity, efficiency, and scalability. Another important advantage of geographic routing is that there is no need to store large routing tables and/or flood the network with route discovery packets. However, nodes' location information may not always be available in specific environments such as deep woods, indoors, and mountains.

One of the ways how to utilize geographic routing without explicit nodes' location information is to rely upon the use of virtual coordinates instead of geographic coordinates. To be suitable for this task, virtual coordinates should reflect the position of nodes relative to each other in the virtual graphs similar to the physical world. To be applicable in practice, algorithms for deducing virtual coordinates should be distributed in nature, operating with minimal overhead by relying upon information about the closest proximity of nodes, for example, locations of neighbor nodes within the communications range [19]. Virtual coordinate systems (VCSs) can be classified into systems that rely upon the use of anchor nodes [20–22] and those that do not utilize this information [23,24]. Table 1 summarizes the related algorithms according to the anchor involvement. Anchor nodes (or landmarks) are designed to serve as a guide for other nodes in determining their virtual coordinates. A set of anchor nodes can consist of nodes that have information about their geographic location or are assigned according to some rule. In general, the performance of VCSs systems relying upon the anchor nodes is better as compared to those, where this information is not utilized.

Table 1. Examples of VCS algorithms Classification.

Anchor-Based	Anchorless
ENS [20]	VCP [23]
Iterative Anchor Selection Algorithm [21]	GPSR [24]
Distributed EATL Anchor Node Selection Algorithm [22]	GAR [19]
Selforganized Landmark Algorithm [25]	UDG [26]

In VCS algorithms that utilize anchor nodes, the location of the anchor nodes in the network is of special importance. In [25], the authors proposed two utilize the voting functions, whose values are assigned to network nodes and define the priority to be declared as anchor nodes. The authors propose two anchor node distribution patterns: (i) uniformly over the network at the maximum distance relative to each other and (ii) along the network boundary. The simulation results demonstrate that the proposed algorithm improves the efficiency of greedy forwarding, preserving the simplicity and high scalability of this routing technique. The authors in [27] utilized perimeter nodes as anchor ones and considered two cases: (i) when none of the nodes has knowledge about its position and (ii) when some of the nodes have location information. They proceed by proposing geographic routing protocols by replacing geographic coordinates with virtual coordinates. Their study was further continued in [28], where an approximate solution for the values of virtual coordinates in terms of fixed coordinates of peripheral anchor nodes is proposed.

To find virtual coordinates, it is convenient to represent the network as a graph. Unit disk graph (UDG) models have proven useful in simulating a variety of real-world physics problems. In [29], UDGs are defined as follows: given n circles with the same radius in the plane, construct a graph with n vertices corresponding to n circles and edges between two vertices if one of the corresponding circles contains the center of the other. The UDG model has been repeatedly used in routing problems [26,30].

Summarizing the related work, we note that the anchor-based algorithms proposed to date so far are a good fit for static networks. Although the complexity and execution time of the anchor node assignment algorithm is greatly reduced, this process requires both time and computational resources. In the cited works, anchor nodes are usually distributed along the edge of the network, but due to the nodes' mobility the edge nodes change, which will require the dynamical reassignment of anchor nodes. The choice of the number of anchor nodes is also not unambiguous. In our paper, we design an algorithm that efficiently utilizes only the information available on the neighboring nodes which solves the described above issue. Specifically, instead of choosing anchor nodes, we employ locally available information and received signal strength (RSS) to calculate the virtual coordinates of the nodes.

3. Requirements for Implementation of the Virtual Coordinate System Algorithm

The algorithm developed in this paper works in a system of virtual coordinates called the Unit Virtual Coordinate System. The algorithm assumes that each node has and regularly updates information about all neighboring nodes. In the case of applying the UVCS to a UAVs swarms, due to radio signal attenuation the distance at which UAVs are able to communicate is limited, so we consider neighbors to be adjacent nodes directly available for communication via a radio interface without any intermediate relay nodes. The name of the system came about by analogy with the unit disk graph [29], if the maximum distance between directly connected UAVs is normalized to one.

The main objective of the proposed UVCS is to enable a network node to calculate its virtual address based on locally available network knowledge in order to minimize the overhead of distributing information throughout the network. The system works in such a way that network nodes do not need to have access to information obtained from external sources, such as absolute geographic GPS coordinates or cellular infrastructure data.

To maintain proper operation of the system, each node in the network must perform a set of actions independent of the other nodes in the network, i.e., a node makes all decisions locally and only notifies the other nodes of the decisions made. All nodes in the network must perform the same set of actions based on the chosen algorithm. This ensures that the system can operate under dynamic conditions, even when individual nodes in the network are completely isolated. These actions include (shown schematically in Figure 2):

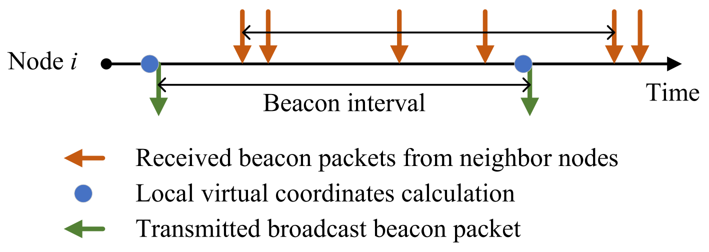

Node *i*
Beacon interval
Time
Received beacon packets from neighbor nodes
Local virtual coordinates calculation
Transmitted broadcast beacon packet

Figure 2. Functioning of the virtual coordinate system from the point of view of a single node.

1. Constantly wait for signal data packets from other nodes in the network and updating information about them in its database.
2. Timely removal of information about other nodes in the network from its database if the signal data packet was not received from them for a certain period of time.
3. Within a given period, recalculate its virtual coordinates based on the collected information about the other nodes of the network since the last recalculation of coordinates (or inclusion in the network).
4. Within a given period, notify all neighboring nodes of the network about their calculated coordinates, and, if set by the selected algorithm, other necessary data, such as the coordinates of their one-hop neighbors. This action is performed by sending a special broadcast data signal packet containing all the necessary data.
5. If specified by the selected algorithm, to perform instantaneous processing of signal messages about the subnet merging.

The proposed system allocates and dynamically assigns logical addresses to nodes based on information received from their immediate neighbors, which periodically exchange special broadcast data signal packets, and estimates their distances based on the RSS of the signal packet. The system dynamically adapts to changing network conditions, including network mergers and disconnections, node disconnections, and node mobility. Thus, the network can dynamically create and maintain a topology, based on which the network nodes have the ability to instantly determine the next node (hop) when forwarding a data packet to the destination. It is worth noting that routes are built according to geographic routing based on the information about the location of nodes in the UVCS. In a similar formulation, the problem is solved in [19], however, unlike our model, the GAR algorithm requires information not only about immediate neighbors, the so-called "first handshake" neighbors, but also about neighbors of neighbors, i.e., "second handshake" neighbors. Thus, maintaining a network topology by the GAR algorithm requires a slightly larger amount of signaling traffic and higher overheads. Algorithm 1 shows a single iteration of the UVCS algorithm for a single node. The same algorithm is assumed to run on every node in the considered network indefinitely to support the timely updates of a node's virtual coordinates, which are required in the case of nodes' mobility. The number of operations on a node at each iteration of the algorithm is of the order of the number of node's neighbors and depends on the density of nodes in the network which leads to polynomial computational complexity for the whole network. The low complexity of the algorithm makes it possible to run the algorithm in real time even on embedded devices.

Algorithm 1: Single iteration of UVCS algorithm.

Input:
1. $s_{n,i}$: virtual coordinates of node i at previous iteration n
2. Data packets from neighboring nodes with data:
    - j: ID of the one-hop neighbor of node i
    - $\mathbf{s}_{n,j}$: virtual coordinates of node j at iteration n
    - N_j: a list of one-hop neighbors' IDs of node j
    - a set of virtual coordinates of N_j
3. $\rho(\mathbf{s}_i, \mathbf{s}_j)$: an estimate of the physical distance between nodes i and j

Result: Virtual coordinates $\mathbf{s}_{n+1,i}$ of node i at iteration $n+1$ of the algorithm

```
Collect beacon packets from neighboring nodes for a specified beacon period;
if No packets collected then
    if $\mathbf{s}_{n,i} \neq NULL$ then
        $\mathbf{s}_{n+1,i} = \mathbf{s}_{k,i}$;
    else
        $\mathbf{s}_{n+1,i} = rand()$;
    end if
else
    Calculate $\mathbf{s}_{n+1,i}$ according to (5);
end if
Broadcast new beacon $\{i, \mathbf{s}_{n,i}, N_i, \mathbf{s}_{n,j}, j \in N_i\}$;
```

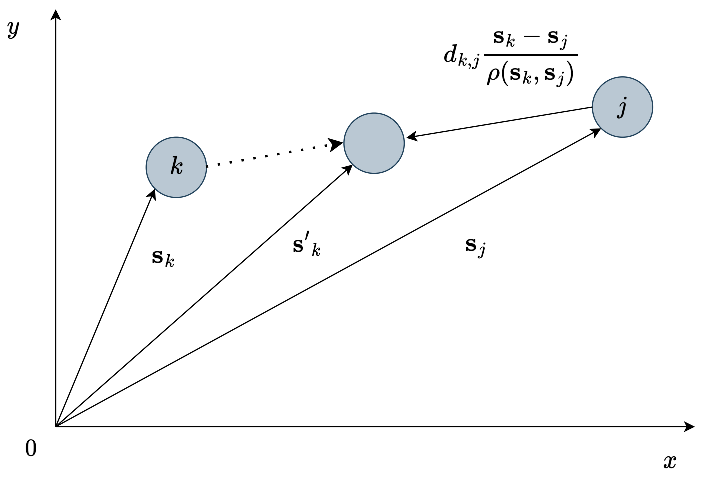
In the next section, we show the mathematical model underlying our algorithm for assigning or refinement virtual coordinates in the UVCS.

4. Mathematical Model at the Heart of the Algorithm

Consider a UAV swarm represented as a mobile network whose nodes can appear, disappear, and move, and in which two nodes are connected if circles of unit radius around these nodes intersect. Recall that the transmission range, i.e., maximum distance between directly connected UAVs, is limited due to radio signal attenuation and the model construction is normalized to one. The corresponding finite oriented graph is defined as $G = (\mathcal{V}, \mathcal{A})$ with set of vertices $\mathcal{V} = \{1, 2, \ldots, N\}$, set of edges $\mathcal{A} \subset \{(i,j) | i,j \in \mathcal{V}, i \neq j\}$, and edge lengths $0 < d_{ij} < 1, (i,j) \in \mathcal{A}$. The purpose of the algorithm is to find a set of virtual coordinates $\boldsymbol{\sigma} = (\mathbf{s}_1, \ldots, \mathbf{s}_N)$, where $\mathbf{s}_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, 2, \ldots, N$, such that for all edges $(i,j) \in \mathcal{A}$ the distance between point $\mathbf{s}_i$ and point $\mathbf{s}_j$ is close to d_{ij} (Fig. 3). A more precise formulation of the minimization problem will be given later.

Let us introduce the following notation: $\mathcal{S}_i = \{j \in \mathcal{V} | (i, j) \in \mathcal{A}\}$: the set of all vertices to which there are edges from node i; $\mathcal{S}_j^{-1} = \{i \in \mathcal{V} | j \in \mathcal{S}_i\}$: the set of vertices from which the edges enter node j. We also define the distance between points $\mathbf{s}_k$ and $\mathbf{s}_j$ as

$$\rho(\mathbf{s}_i, \mathbf{s}_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}. \tag{1}$$



y
$d_{k,j}\frac{\mathbf{s}_k - \mathbf{s}_j}{\rho(\mathbf{s}_k, \mathbf{s}_j)}$
j
k
$\mathbf{s}_k$
$\mathbf{s}'_k$
$\mathbf{s}_j$
0
x

Figure 3. Explanation of an intuitive solution to the localization problem.

When solving the localization problem, we should specify point $\mathbf{s}_k$ in relation to point $\mathbf{s}_j$ in such a way that node k becomes $d_{k,j}$ far from node j. Intuitively, it would be good to move the point $\mathbf{s}_k$ to a new position $\mathbf{s}'_k$:

$$\mathbf{s}'_k = \mathbf{s}_j + d_{k,j}\frac{\mathbf{s}_k - \mathbf{s}_j}{\rho(\mathbf{s}_k, \mathbf{s}_j)} = \mathbf{s}_k + \left(1 - \frac{d_{k,j}}{\rho(\mathbf{s}_k, \mathbf{s}_j)}\right)(\mathbf{s}_j - \mathbf{s}_k). \tag{2}$$

If we average the desired positions of the point $\mathbf{s}_k$ with respect to all the surrounding points $\mathbf{s}_j$, $j \in \mathcal{S}_k$, then we obtain

$$\mathbf{s}'_k = \mathbf{s}_k + \frac{1}{|\mathcal{S}_k|}\sum_{j \in \mathcal{S}_k}\left(1 - \frac{d_{k,j}}{\rho(\mathbf{s}_k, \mathbf{s}_j)}\right)(\mathbf{s}_{n,j} - \mathbf{s}_{n,k}). \tag{3}$$

This brings us to the iterative procedure

$$\mathbf{s}_{n+1,k} = \mathbf{s}_{n,k} + \tfrac{\varepsilon_n}{|\mathcal{S}_k|} \sum_{j\in\mathcal{S}_k} \left(1 - \tfrac{d_{k,j}}{\rho(\mathbf{s}_{n,k},\mathbf{s}_{n,j})}\right)(\mathbf{s}_{n,j} - \mathbf{s}_{n,k}) =$$

$$= \mathbf{s}_{n,k} - \frac{\varepsilon_n}{|\mathcal{S}_k|} \sum_{j\in\mathcal{S}_k} \left(1 - \frac{d_{k,j}}{\rho(\mathbf{s}_{n,k},\mathbf{s}_{n,j})}\right)(\mathbf{s}_{n,k} - \mathbf{s}_{n,j}), k = 1,2,\ldots,N, n = 1,2,\ldots \quad (4)$$

where $0 < \varepsilon_n < 1$: the parameter determining how drastically the new value $\mathbf{s}_{n+1,k}$ of the vector $\mathbf{s}_k$ will differ from its previous value $\mathbf{s}_{n,k}$ [31]. Iterations (4) are very similar to the coordinate-wise minimization of some function $F(\sigma)$ with iterations

$$\sigma_{n+1} = \sigma_n - \varepsilon_n \pi_n, \quad n = 1,2,\ldots \quad (5)$$

There are various methods for choosing parameters ε_n and vectors π_n in (5) [32]. The classic approach is the fast descent method, in which the gradient $\nabla_k F(\sigma_n)$ is taken as a vector π_n.

In Appendix A, it is shown that, for the minimization problem of a function,

$$F(\sigma) = F(\mathbf{s}_1,\ldots,\mathbf{s}_N) = \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{S}_i} F_{i,j}(\rho(\mathbf{s}_i,\mathbf{s}_j)) \quad (6)$$

and with differentiated terms $F_{i,j}(x)$ the gradient $\nabla F(\sigma)$ is given by

$$\nabla F(\sigma) = (\nabla_1 F(\sigma), \nabla_2 F(\sigma), \ldots, \nabla_N F(\sigma)), \quad (7)$$

$$\nabla_k F(\mathbf{s}_1,\ldots,\mathbf{s}_N) = \sum_{j\in\mathcal{S}_k} \frac{f_{k,j}(\rho(\mathbf{s}_k,\mathbf{s}_j))}{\rho(\mathbf{s}_k,\mathbf{s}_j)}(\mathbf{s}_k - \mathbf{s}_j) + \sum_{i\in{\mathcal{S}_k}^{-1}} \frac{f_{i,k}(\rho(\mathbf{s}_i,\mathbf{s}_k))}{\rho(\mathbf{s}_i,\mathbf{s}_k)}(\mathbf{s}_k - \mathbf{s}_i), \quad (8)$$

where $f_{i,j}(x) = \frac{d}{dx}F_{i,j}(x)$.

An important special case of graph $G = (\mathcal{V},\mathcal{A})$ is the one with symmetric adjacency matrix. Regarding application to a UAVs swarm analysis, this property reflects the features of a connection in the real swarm — if data transfer from node i to node j is possible, then it is possible in the opposite direction. In this case, the conditions

$$\mathcal{S}_i^{-1} = \mathcal{S}_i, i \in \mathcal{V},\ F_{i,j}(x) = F_{j,i}(x), j \in \mathcal{S}_i, i \in \mathcal{V}, \quad (9)$$

are satisfied. For such graphs, (8) takes the form

$$\nabla_k F(\mathbf{s}_1,\ldots,\mathbf{s}_N) = 2\sum_{j\in\mathcal{S}_k} \frac{f_{k,j}(\rho(\mathbf{s}_k,\mathbf{s}_j))}{\rho(\mathbf{s}_k,\mathbf{s}_j)}(\mathbf{s}_k - \mathbf{s}_j). \quad (10)$$

For illustration, if $F_{k,j}(x)$ and $f_{k,j}(x)$ in (6), (8) have the form of

$$F_{k,j}(x) = \frac{A_k}{8}(x^2 - {d_{k,j}}^2)^2, \qquad f_{k,j}(x) = \frac{A_k}{2}x(x^2 - {d_{k,j}}^2), \quad (11)$$

then Equation (10) takes the form of

$$\nabla_k F(\mathbf{s}_1,\ldots,\mathbf{s}_N) = A_k \sum_{j\in\mathcal{S}_k} (\rho^2(\mathbf{s}_k,\mathbf{s}_j) - {d_{k,j}}^2)(\mathbf{s}_k - \mathbf{s}_j), \quad (12)$$

where A_k are some coefficient, $k \in \mathcal{V}$. One may compare to (3) with $A_k = |\mathcal{S}_k|^{-1}$.

Therefore, the localization problem of obtaining σ takes the form of minimization (7) subject to (9), (10).

In the UVCS approach, descending in an iterative procedure (4) is carried out by cycles in which the number of steps is equal to the number of variables of the minimized function, in our case $3N$. It is also possible to carry out optimization at each step not for individual variables, but for groups of some variables. In our case, those groups are vectors $\mathbf{s}_i = (x_i, y_i, z_i), i = 1, 2, \ldots, N$ [33].

To calculate the next approximation, the gradient descent method suggests

$$\mathbf{s}_{n+1,k} = \mathbf{s}_{n,k} - \lambda_{n,k} \nabla_k F(\mathbf{s}_{n,1}, \mathbf{s}_{n,2}, \ldots, \mathbf{s}_{n,N}), \quad k = 1, 2, \ldots, N, \quad n = 1, 2, \ldots, \tag{13}$$

where $\lambda_{n,k} > 0$ is some constant, $\nabla_k F(\mathbf{s}_{n,1}, \mathbf{s}_{n,2}, \ldots, \mathbf{s}_{n,N})$ is a gradient of the function $F(\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_N)$ by the variable $\mathbf{s}_k$, i.e,

$$\nabla_k F(\mathbf{s}_{n,1}, \mathbf{s}_{n,2}, \ldots, \mathbf{s}_{n,N}) = \begin{pmatrix} \frac{d}{dx_k} F(\mathbf{s}_1, \ldots, \mathbf{s}_N) \\ \frac{d}{dy_k} F(\mathbf{s}_1, \ldots, \mathbf{s}_N) \\ \frac{d}{dz_k} F(\mathbf{s}_1, \ldots, \mathbf{s}_N) \end{pmatrix}. \tag{14}$$

The convergence of iterative procedure (13) is defined by the Lipschitz condition [34]. The parameter $\lambda_{n,k}$ should be chosen in such a way that after the replacement $\mathbf{s}_{n,k}$ of $\mathbf{s}_{n+1,k}$ the new value of the function $F(\mathbf{s}_{n,1}, \mathbf{s}_{n,2}, \ldots, \mathbf{s}_{n,N})$ increases drastically.

Finally, in the UVCS approach the gradient descent algorithm itself is as follows.

By carrying out n cycles, we have the following.

- Vectors $\mathbf{s}_{n,2}, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N}$ remain unchanged while we minimize the function $F_{n+1,1}(\mathbf{s}_1) = F(\mathbf{s}_1, \mathbf{s}_{n,2}, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N})$ by the vector $\mathbf{s}_1$.
  We obtain the approximation $(\mathbf{s}_{n+1,1}, \mathbf{s}_{n,2}, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N})$, where $\mathbf{s}_{n+1,1}$—-the optimal value of the vector $\mathbf{s}_1$.
- Vectors $\mathbf{s}_{n+1,1}, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N}$ remain unchanged while we minimize the function $F_{n+1,2}(\mathbf{s}_2) = F(\mathbf{s}_{n+1,1}, \mathbf{s}_2, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N})$ by the vector $\mathbf{s}_2$.
  We obtain the approximation $(\mathbf{s}_{n+1,1}, \mathbf{s}_{n+1,2}, \mathbf{s}_{n,3}, \ldots, \mathbf{s}_{n,N})$, where $\mathbf{s}_{n+1,2}$— is the optimal value of the vector $\mathbf{s}_2$.
  ...
- Vectors $\mathbf{s}_{n+1,1}, \mathbf{s}_{n+1,2}, \ldots, \mathbf{s}_{n+1,N-1}$ remain unchanged while we minimize the function $F_{n+1,N}(\mathbf{s}_N) = F(\mathbf{s}_{n+1,1}, \mathbf{s}_{n+1,2}, \ldots, \mathbf{s}_{n+1,N-1}, \mathbf{s}_N)$ by the vector $\mathbf{s}_N$.
  We obtain the approximation $(\mathbf{s}_{n+1,1}, \mathbf{s}_{n+1,2}, \ldots, \mathbf{s}_{n+1,N-1}, \mathbf{s}_{n+1,N})$, where $\mathbf{s}_{n+1,N}$— is the optimal value of the vector $\mathbf{s}_N$.

After N steps, where each step corresponds to vector $\mathbf{s}_k$, $k = 1, 2, \ldots, N$, the $(n+1)$-th cycle is finished. The process is repeated for $n = 1, 2, \ldots,$ until the stopping condition is satisfied either by the execution time or by an achieved accuracy.

As a result of solving the localization problem for each vertex $i \in \mathcal{V}$, we obtain the values $\mathbf{s}_i$ of virtual coordinates, that form the desired set of virtual coordinates $\sigma = (\mathbf{s}_1, \ldots, \mathbf{s}_N)$ for UAVs in the swarm. Note that the representation σ allows us to work with it by standard routing methods. In Section 5, we define the performance metrics and in Section 6 evaluate the performance of the proposed UVCS approach in comparison with Greedy Forwarding Geographic Routing (Geo) algorithm, Gradient Assisted Routing (GAR) algorithm [19], and Dijkstra algorithm as a benchmark.

5. Performance Metrics

The development of metrics for evaluation of the effectiveness of the developed UVCS approach for self-organization and topology support in a drone swarm is a separate task, that has two aspects. On the one hand, positioning accuracy can be evaluated, while, on the other hand, the ability to maintain swarm connectivity should be estimated. The peculiarity is that graphs built on the known physical coordinates and on the calculated virtual coordinates are isomorphic [35,36]. It should also be taken into account that different routing algorithms in the same coordinate system can give different results in terms of the route characteristics.

A routing algorithm can be formally described by the rule for choosing the first vertex $n_k(i)$ on the path from vertex i to vertex k. Here $n_k(i) \in \overline{\mathcal{S}_i}$, $i,k \in \mathcal{V}$, where $\overline{\mathcal{S}_i} = \mathcal{S}_i \cup \{i\}$ is the set of vertices containing edges originating from vertex i including a vertex i itself.

For the Geo algorithm the choice of node $n_k(i)$ follows the principle

$$n_k(i) = \underset{j \in \mathcal{S}_i}{\operatorname{argmin}} \rho(\mathbf{s}_j, \mathbf{s}_k), i \neq j. \tag{15}$$

This section describes several performance metrics: topological similarity index, routing similarity index, number of discovered paths, and average discovered path length. All metrics depend on the routing algorithm that works on the top of the coordinate system—either physical or virtual.

The Geo routing algorithm (15) is used by default for all metrics, and we also use Dijkstra's algorithm for the shortest path finding as a benchmark to evaluate the last two metrics—number and mean length of discovered paths.

The *topological similarity index* compares the degree to which the virtual graph replicates the physical graph in terms of distances between vertices. We calculate by (1) the distance $\rho(\mathbf{s}_i, \mathbf{s}_j)$ in all pairs (i,j) of vertices, regardless of their connectivity, and, using Pearson's correlation coefficient [37], compare the pairwise distances between them.

The next metrics of interest are a group of routing metrics—routing similarity index, number of discovered paths, and average discovered path length.

The *routing similarity index* is based on application of the apparatus of closed queuing networks and is carried out in two steps. The first step is to develop the stochastic routing matrix $\mathbf{P} = [P(i,j)]$ of transition probabilities, $i,j \in \mathcal{V}$. Assuming that, from each node, a customer equiprobably chooses one of the adjacent nodes from $\overline{\mathcal{S}_i}$, we obtain

$$\sum_{j \in \overline{\mathcal{S}_i}} \delta_{n_k(i),j} = 1, \qquad \sum_{k=1}^{N} \sum_{j \in \overline{\mathcal{S}_i}} \delta_{n_k(i),j} = N. \tag{16}$$

Then, the transition probability from node i to node j, $i,j \in \overline{\mathcal{S}_i}$, is as the following:

$$P(i,j) = \frac{1}{N} \sum_{k=1}^{N} \delta_{n_k(i),j}. \tag{17}$$

In the second step, we solve the balance equations

$$\mathbf{pP} = \mathbf{p}, \qquad \mathbf{pu} = 1, \tag{18}$$

and obtain a vector $\mathbf{p}$ with the components $p(i)$ representing the proportion of time spent by the customer in node $i \in \mathcal{V}$ or the frequency of usage of node i while routing. As a measure of the difference between the two routing schemes the following metric can be considered:

$$\|\mathbf{p}_1 - \mathbf{p}_2\| = \sum_{i=1}^{N} |p_1(i) - p_2(i)|. \tag{19}$$

In case $\mathbf{p}_1$ is a vector related to the physical graph, and $\mathbf{p}_2$ is related to the virtual one, metric (19) shows the degree of similarity of routing in the physical and virtual graphs. The approach (19) can also be used to evaluate the degree of similarity of two routing algorithms.

Finally, we define two more routing metrics—the *number of paths* and the *average path length* metrics. They show how many paths are found and how many edges they contain on average at a given time stamp according to the given routing scheme.

6. Numerical Results

In this section, we evaluate the performance of the proposed UVCS virtual coordinate system on the top of the Geo routing algorithm based on (i) topological metrics, describing the similarity between virtual and physical topology, and (ii) routing metrics, reflecting the routing performance. We also compare the proposed approach to other routing schemes such as the Gradient Assisted Routing (GAR) algorithm [19] with virtual coordinates, as well as the Geo routing algorithm with known physical coordinates and the Dijkstra algorithm as a benchmark.

6.1. Metrics of Interest and Scenario

As the primary indicator of the overall similarity between the network topology and the virtual one, we utilize the so-called topology similarity index. This index is defined as the Pearson correlation coefficient [37] between pairwise distances in the virtual and physical topology. For the routing performance, we provide a set of different metrics that, when considered together, propose deep insights into the developed approach's performance by assessing how close it is to the perfect case with full knowledge based on the Dijkstra algorithm. These metrics include (a) path similarity index that shows how close are chosen routes to those detected by the Dijkstra algorithm, (b) total number of discovered routes, and (c) average length of discovered routes, reflecting the ability of the routing protocol to discover paths in a network. Finally, we also compare the proposed approach to the Gradient Assisted Routing (GAR) algorithm specified in [19].

To perform the comparison, we consider a wireless stationary network with a maximum of 50 randomly located nodes. Nodes appear one by one with a small delay (approximately five time steps) until all 50 nodes are online. At every given time instant, all currently online nodes organize a fully connected graph. Once all nodes are online, some additional time is given for the UVCS algorithm to converge.

6.2. Topological Metric

We start with the topological metric in Figure 4 illustrating the topology similarity index as a function of the successive algorithm time steps. It can be observed that the GAR algorithm provides a closer overall topology resemblance to the original physical graph, while both GAR and the proposed UVCS approaches improve the virtual topology in time. We specifically emphasize the ability of the proposed UVCS algorithm to converge much faster—in less than 100 steps. The Geo routing algorithm based on physical coordinates is not included in that figure, as it relies on the real nodes coordinates, resulting in the one-to-one similarity.

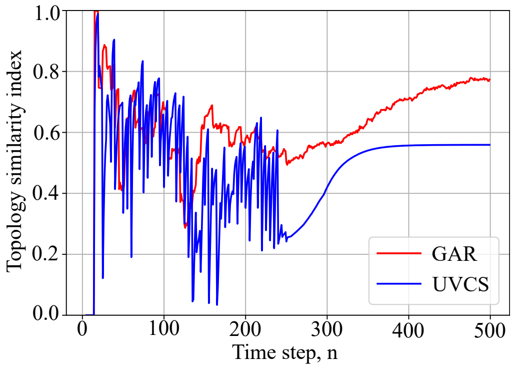


Figure 4. Topology similarity index.

6.3. Routing Metrics

Recall that the main aim of the proposed UVCS algorithm is to perform efficient routing between possibly mobile nodes. At the same time, observe that even perfect topology matching does not guarantee that the routing performance will be similar for different routing algorithms. For this reason, we now proceed to investigate the routing performance of the proposed UVCS approach using different routing metrics including practical geographical routing protocols.

Here, we start with the path similarity index provided in Figure 5 and show how fast the routes may converge to the shortest ones based on the Dijkstra algorithm. Recall that the closer the value is to 0 the better the match in terms of selected paths is. Here, we can see that the proposed algorithm again converges to its final solution much faster, even though the similarity index is not as good as for the GAR or Geo algorithms—0.6 against 0.5 and 0.4, respectively.

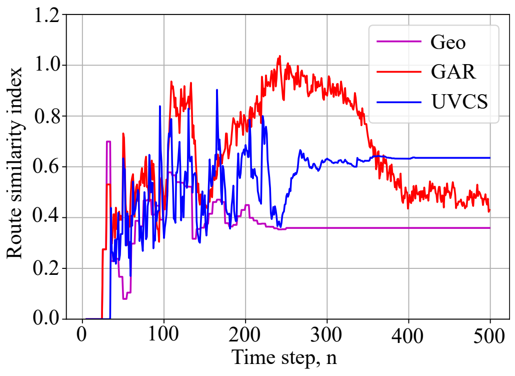


Figure 5. Routing similarity index.

One of the weakest points of the GAR, originally proposed in [19], is a low number of total discovered paths in the network. Even though the proposed UVCS algorithm does not show the best performance in terms of topology similarity metric, Figure 6 shows that the total number of discovered routes is higher compared to the GAR algorithm. This number is even comparable and is getting very close to the Geo algorithm that relies on external information for routing. To achieve these performance gains in terms of the number of discovered paths, the proposed algorithm in some cases finds longer, less optimal routes to the destination, shown in Figure 7, which may be considered a minor drawback for delay critical applications. However, recalling that the UVCS does not rely on any external positioning information, unlike the original Geo algorithm, this slight increase in average path length enables a much higher number of discovered paths together with a faster convergence time.

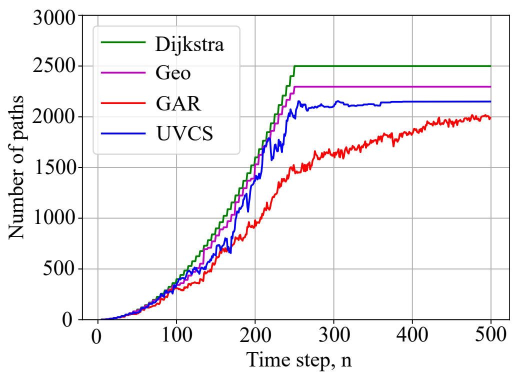


Figure 6. Total number of discovered paths.

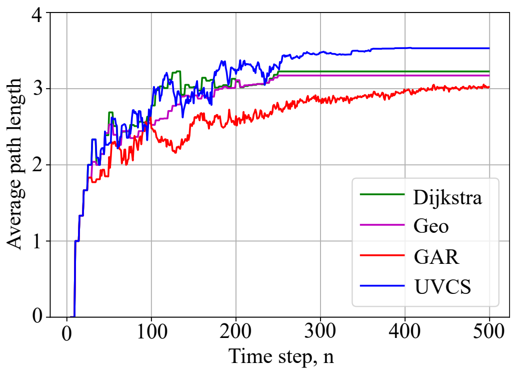


Figure 7. Average discovered path length.

7. Conclusions

This paper proposes an approach to organize and maintain the topology of a distributed network. The proposed UVCS algorithm is designed to find the coordinates of a network node, operating only with local information available at the node itself. One of the advantages of the UVCS is its minimal input information requirements. Further, the virtual coordinates can be used to implement geographic routing in the network. In the performance evaluation, we assessed the UVCS by topological and routing metrics. Numerical analysis showed that the virtual coordinates found with the UVCS algorithm are close to the physical coordinates. Secondly, the UVCS algorithm outperforms all known algorithms in terms of the total number of discovered paths, which can be critical to maintaining the connectivity of a UAV swarm. However, the main advantage of the UVCS is its fast convergence. In future, the problem can be developed in the direction of building virtual coordinate systems with anchor nodes, while the number and rule for assigning nodes as an anchor can become the subject of research to maintain a balance is between positioning accuracy and overheads due to the amount of service traffic when implementing the algorithm. Analysis of data transmission delay depending on the route scheme is also a part of this research.

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

Funding: The reported study was funded by RSF, project number 22-29-00694, https://rscf.ru/en/project/22-29-00694/ (accessed on 21 December 2022).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank Valeriy Naumov for the ideas that led to the development of the iterative algorithm.

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

Appendix A

Proof. Let us proof formulas (7) and (8).

First note that the derivatives of distance $\rho(\mathbf{s}_i, \mathbf{s}_j)$ on x_k have the following properties:

$$\frac{\partial}{\partial x_k}\rho(\mathbf{s}_k, \mathbf{s}_j) = \frac{\partial}{\partial x_k}\rho(\mathbf{s}_j, \mathbf{s}_k) \quad \text{and} \quad \frac{\partial}{\partial x_k}\rho(\mathbf{s}_i, \mathbf{s}_j) = 0, \quad \text{if} \quad k \neq i \quad \text{and} \quad k \neq j. \tag{A1}$$

$$\frac{\partial}{\partial x_k}\rho(\mathbf{s}_k,\mathbf{s}_j) = \frac{1}{2}\frac{2(x_k - x_j)}{\sqrt{(x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2}} = \frac{(x_k - x_j)}{\rho(\mathbf{s}_k,\mathbf{s}_j)}. \quad \text{(A2)}$$

Similar formulas are valid for the derivatives $\rho(\mathbf{s}_i,\mathbf{s}_j)$ on y_k and z_k.
Using (A2), we obtain

$$\frac{\partial}{\partial x_k}F(\sigma) = \frac{\partial}{\partial x_k}\sum_{i=1}^{N}\sum_{j\in S_i}F_{i,j}(\rho(\mathbf{s}_i,\mathbf{s}_j)) = \quad \text{(A3)}$$

$$= \frac{\partial}{\partial x_k}\Big(\sum_{j\in S_k}F_{k,j}(\rho(\mathbf{s}_k,\mathbf{s}_j)) + \sum_{i\in S_k^{-1}}F_{i,k}(\rho(\mathbf{s}_i,\mathbf{s}_k))\Big) = \quad \text{(A4)}$$

$$= \sum_{j\in S_k} f_{k,j}(\rho(\mathbf{s}_k,\mathbf{s}_j))\frac{(x_k - x_j)}{\rho(\mathbf{s}_k,\mathbf{s}_j)} + \sum_{i\in S_k^{-1}} f_{i,k}(\rho(\mathbf{s}_i,\mathbf{s}_k))\frac{(x_k - x_i)}{\rho(\mathbf{s}_i,\mathbf{s}_k)}. \quad \text{(A5)}$$

Similarly, using (A5), formulas for derivatives are derived for $\frac{\partial F(\sigma)}{\partial y_k}$ and $\frac{\partial F(\sigma)}{\partial z_k}$. $\square$

References

1. *P2413/D0.4.6, March 2019*; IEEE Draft Standard for an Architectural Framework for the Internet of Things (IoT). IEEE: Piscataway Township, NJ, USA, 2019; pp. 1–265.
2. Asheralieva, A.; Niyato, D. Optimizing Age of Information and Security of the Next-Generation Internet of Everything Systems. *IEEE Internet Things J.* **2022**, *9*, 20331–20351. [CrossRef]
3. Liu, Y.; Dai, H.N.; Wang, Q.; Shukla, M.K.; Imran, M. Unmanned aerial vehicle for internet of everything: Opportunities and challenges. *Comput. Commun.* **2020**, *155*, 66–83. [CrossRef]
4. Jamil, S.; Rahman, M.; Fawad, E. A Comprehensive Survey of Digital Twins and Federated Learning for Industrial Internet of Things (IIoT), Internet of Vehicles (IoV) and Internet of Drones (IoD). *Appl. Syst. Innov.* **2022**, *5*, 56. [CrossRef]
5. Chen, W.; Liu, J.; Guo, H.; Kato, N. Toward robust and intelligent drone swarm: Challenges and future directions. *IEEE Netw.* **2020**, *34*, 278–283. [CrossRef]
6. Power, W.; Pavlovski, M.; Saranovic, D.; Stojkovic, I.; Obradovic, Z. Autonomous navigation for drone swarms in GPS-denied environments using structured learning. In Proceedings of the IFIP International Conference on AI Applications and Innovations, Neos Marmaras, Greece, 5–7 June 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 219–231.
7. Yavuz, D.; Akbıyık, H.; Bostancı, E. Intelligent drone navigation for search and rescue operations. In Proceedings of the 2016 24th Signal Processing and Communication Application Conference (SIU), Zonguldak, Turkey, 16–19 May 2016; pp. 565–568. [CrossRef]
8. Mishra, D.; Natalizio, E. A survey on cellular-connected UAVs: Design challenges, enabling 5G/B5G innovations, and experimental advancements. *Comput. Netw.* **2020**, *182*, 107451. [CrossRef]
9. Haas, Z.J.; Deng, J.; Liang, B.; Papadimitratos, P.; Sajama, S. *Wireless Ad Hoc Networks*; Wiley Encyclopedia of Telecommunications; Wiley: Hoboken, NJ, USA, 2003.
10. Ganesh, S.; Gopalasamy, V.; Sai Shibu, N.B. Architecture for Drone Assisted Emergency Ad-hoc Network for Disaster Rescue Operations. In Proceedings of the 2021 International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 5–9 January 2021; pp. 44–49. [CrossRef]
11. Broch, J.; Maltz, D.A.; Johnson, D.B.; Hu, Y.C.; Jetcheva, J. A performance comparison of multi-hop wireless ad hoc network routing protocols. In Proceedings of the 4th annual ACM/IEEE International Conference on Mobile Computing and Networking, Dallas , TX, USA, 25–30 October 1998; pp. 85–97.
12. Melodia, T.; Pompili, D.; Akyildiz, I. Optimal local topology knowledge for energy efficient geographical routing in sensor networks. In Proceedings of the IEEE INFOCOM 2004, Hong Kong, China, 7–11 March 2004; Volume 3, pp. 1705–1716. [CrossRef]
13. Chiang, S.; Huang, C.; Chang, K. A Minimum Hop Routing Protocol for Home Security Systems Using Wireless Sensor Networks. *IEEE Trans. Consum. Electron.* **2007**, *53*, 1483–1489. [CrossRef]
14. Chahal, S.; Singh, M. An Extensive Literature Review of Various Routing Protocols in Delay Tolerant Networks. *Int. Res. J. Eng. Technol. (IRJET)* **2017**, *4*, 1309–1312.
15. Perkins, C.E.; Belding-Royer, E.M.; Das, S.R. *Ad hoc On-Demand Distance Vector (AODV) Routing*; RFC 3561; RFC Editor, 2003 Available online: https://www.rfc-editor.org/rfc/rfc3561 (accessed on 21 December 2022).
16. Kaur, H.; Singh, H.; Sharma, A. Geographic Routing Protocol: A Review. *Int. J. Grid Distrib. Comput.* **2016**, *9*, 245–254. [CrossRef]
17. Lyu, C.; Gu, D.; Zhang, X.; Sun, S.; Zhang, Y.; Pande, A. SGOR Secure and scalable geographic opportunistic routing. *Comput. Commun.* **2015**, *59*, 37–57. [CrossRef]

18. Medjiah, S.; Ahmed, T.; Krief, F. AGEM: Adaptive Greedy-Compass Energy-Aware Multipath Routing Protocol for WMSNs. In Proceedings of the 2010 7th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 9–12 January 2010; pp. 1–6. [CrossRef]
19. Samuylov, A.; Moltchanov, D.; Kovalchukov, R.; Gaydamaka, A.; Pyattaev, A.; Koucheryavy, Y. GAR: Gradient assisted routing for topology self-organization in dynamic mesh networks. *Comput. Commun.* **2022**, *190*, 10–23. [CrossRef]
20. Dhanapala, D.C.; Jayasumana, A.P. Anchor selection and Topology Preserving Maps in WSNs—A Directional Virtual Coordinate based approach. In Proceedings of the 2011 IEEE 36th Conference on Local Computer Networks, Bonn, Germany, 4–7 October 2011; pp. 571–579. [CrossRef]
21. Bouchoucha, T.; Ding, Z. Anchor selection for topology inference and routing in wireless sensor networks. *J. Commun. Inf. Netw.* **2020**, *5*, 318–323. [CrossRef]
22. Fan, Y.; Qi, X.; Yu, B.; Liu, L. A Distributed Anchor Node Selection Algorithm Based on Error Analysis for Trilateration Localization. *Math. Probl. Eng.* **2018**, *2018*, 7295702. [CrossRef]
23. Awad, A.; German, R.; Dressler, F. Exploiting Virtual Coordinates for Improved Routing Performance in Sensor Networks. *IEEE Trans. Mob. Comput.* **2011**, *10*, 1214–1226. [CrossRef]
24. Filardi, N.; Caruso, A.; Chessa, S. Virtual Naming and Geographic Routing on Wireless Sensor Networks. In Proceedings of the 2007 12th IEEE Symposium on Computers and Communications, Santiago, Portugal, 1–4 July 2007; pp. 609–614. [CrossRef]
25. Baskakov, S. Landmarks Selection Algorithm for Wireless Sensor Networks. In Proceedings of the 2008 Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems, Venezia, Italy, 20–24 October 2008; pp. 361–369. [CrossRef]
26. Kuhn, F.; Wattenhofer, R.; Zollinger, A. Ad-Hoc Networks beyond Unit Disk Graphs. In Proceedings of the DIALM-POMC '03, 2003 Joint Workshop on Foundations of Mobile Computing, San Diego CA USA, 19 September 2003; Association for Computing Machinery: New York, NY, USA, 2003; pp. 69–78. [CrossRef]
27. Rao, A.; Ratnasamy, S.; Papadimitriou, C.; Shenker, S.; Stoica, I. Geographic Routing without Location Information. In Proceedings of the MobiCom03: Ninth Annual International Conference on Mobile Computing and Networking, San Diego CA USA, 14–19 September 2003. . [CrossRef]
28. Jadbabaie, A. On geographic routing without location information. In Proceedings of the 2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601), Nassau, Bahamas, 14–17 December 2004; pp. 4764–4769.
29. Clark, B.N.; Colbourn, C.J.; Johnson, D.S. Unit disk graphs. *Discret. Math.* **1990**, *86*, 165–177. [CrossRef]
30. Kaplan, H.; Mulzer, W.; Roditty, L.; Seiferth, P. Routing in Unit Disk Graphs. *Algorithmica* **2018**, *80*, 830–848 [CrossRef]
31. Schechter, S. Iteration Methods for Nonlinear Problems. *Trans. Am. Math. Soc.* **1962**, *104*, 179–189. [CrossRef]
32. Ortega, J. M.; Rheinboldt, W.C. *Iterative Solution of Nonlinear Equations in Several Variables*; Academic Press: Cambridge, MA, USA, 1970.
33. Golstein, E.; Jusin, D. Methods of calculation and synthesis of pulse automatic systems. *Autom. Remote Control* **1963**, *24*.
34. Poliak, B. *Introduction to Optimization*; Optimization Software: New York, NY, USA, 1987.
35. Bunke, H.; Shearer, K. A graph distance metric based on the maximal common subgraph. *Pattern Recognit. Lett.* **1998**, *19*, 255–259. [CrossRef]
36. Wallis, W.; Shoubridge, P.; Kraetz, M.; Ray, D. Graph distances using graph union. *Pattern Recognit. Lett.* **2001**, *22*, 701–704. [CrossRef]
37. Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 1–4.