

ABSTRACT

RANJAN, PRITESH. Robust Geographic Routing Protocol for Inter Drone Communication. (Under the direction of Dr. Rudra Dutta.)

With increasing onboard computing and communication capability, small Unmanned Aerial Vehicles (UAVs) or drones, individually or in platoons, are increasingly becoming an attractive prospective platform for many smart cities, smart agriculture, and environmental sensing applications. Autonomously operating multi-UAV fleets pose many research challenges like co-operative path planning, collision avoidance, and autonomous mission planning. Most approaches to controlling fleets presuppose reliable communication capability among the UAVs. The problem of routing in the airborne drone network is thus itself an important problem.

Traditional destination IP based routing protocols impose high overhead in aerial mobile ad-hoc networks due to node mobility which leads to frequent topology changes. Geographic routing or position-based routing protocols which take the aid of geographic locations of nodes have been demonstrated to scale better for mobile ad-hoc networks. However, most of these protocols have been designed with 2-dimensional networks in mind which are not directly applicable to 3-dimensional networks.

In this thesis, we study some geographical routing protocols and propose a geographic routing protocol for aerial networks. Our proposed protocol utilizes the inherently broadcast nature of the wireless medium to its advantage by making the intermediate nodes do a constrained flooding towards the destination. Our proposed routing scheme matches the reliability of a network-wide flooding algorithm while avoiding the overhead associated with flooding. We demonstrate the feasibility of our protocol among a group of quadrotor drones in Qrsim simulator working on plume wrapping problem and present statistical data on the performance evaluation on metrics like transmission overhead, average number of hops and delivery ratio.

© Copyright 2018 by Pritesh Ranjan

All Rights Reserved

Robust Geographic Routing Protocol for Inter Drone Communication

by
Pritesh Ranjan

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Computer Science

Raleigh, North Carolina

2018

APPROVED BY:

Dr. Muhammad Shahzad

Dr. Mihail Sichitiu

Dr. Rudra Dutta
Chair of Advisory Committee

DEDICATION

To my mother Dr. Pritam Kumari, my father Shri Rajiv Kumar and the memory of my uncle Shri Sanjeev Kumar Trivedi.

To the teachers and mentors — Dr. Rudra Dutta, Mr. Rahul Sharma, Mr. Sanjeev Sharma, Mrs. Meeta Kumar, Mr. Hasnat Hasan and Dr. Rakesh Sharma, Dr. Pritam Kumari — who helped and guided me at some critical times.

BIOGRAPHY

The author is from Muzaffarpur, Bihar, India. He received his Bachelors of Engineering in Computer Engineering from University of Pune (MIT College of Engineering) in 2014 and worked as a Software Engineer at Amdocs Inc. and Calsoft Inc. from 2014 to 2016. The author then enrolled in Masters of Science in Computer Science program in North Carolina State University in 2016.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Rudra Dutta for guiding me throughout my master's program. His insight and feedback were critical for the completion of my thesis.

I would like to thank Dr. Muhammad Shahzad and Dr. Mihail Sichitiu for serving on my advisory committee.

I would like to thank Angelyn and Harsh for helping at various stages of my work. Special thanks to Mrunal for proofreading my thesis draft and Nikhil for the discussions on probability, algebra, and life.

Many thanks to Shri Shambhu Sharan Prasad Roy, Shri Nandkishor Trivedi and Mr. Bal Mukund for their support and constant encouragement during my master's studies.

Last but not the least, my cousins, Prashant, Savyasachi, Aparajita, Rashmi and my friends for the fun, joy, and laughs.

TABLE OF CONTENTS

LIST OF TABLES	vii
LIST OF FIGURES	viii
Chapter 1 Introduction	1
1.1 Aerial networks and the challenges	1
1.2 Problem statement	3
1.3 Thesis Organization	4
Chapter 2 Background	5
2.1 Flying Ad-hoc networks (FANETs)	5
2.1.1 FANET Characteristics and Applications	6
2.2 Routing in 2D ad hoc networks	8
2.2.1 Topology-based routing protocols	8
2.2.2 Position-based routing protocols	9
2.3 Routing in 3D ad hoc networks	11
Chapter 3 FANET routing protocol	12
3.1 FANET Mission	12
3.2 Communication requirements	13
3.3 Routing protocol	18
3.3.1 Protocol description	18
3.3.2 Message primitives	21
Chapter 4 Implementation	27
4.1 Simulator Introduction: Qrsim	27
4.2 Qrsim Extensions	28
4.2.1 Radio propagation model	28
4.2.2 Processing delay	29
4.2.3 Packet Header	32
4.2.4 Message transmission	34
4.2.5 Location Information	34
4.2.6 Transmission Zone	36
4.2.7 Back-off time	39
4.2.8 Inter-UAV forces	40
Chapter 5 Simulation Results	43
5.1 Packet delivery ratio	44
5.2 Average number of hops	48
5.3 Average number of transmissions	50

5.4	Effect of network density	54
5.5	Effect of location inaccuracy	58
Chapter 6	Conclusions and Future Work	62
BIBLIOGRAPHY		64
APPENDIX		67
Appendix A	Sample Performance Metrics	68

LIST OF TABLES

Table 2.1	Comparison of different type of FANET nodes	6
Table 2.2	UAV Applications	7
Table 3.1	Communication requirements	17
Table 4.1	Free space path loss - equation parameters	31

LIST OF FIGURES

Figure 3.1	Mesh formation of UAVs at the start of the mission	14
Figure 3.2	UAVs wrapping around the plume. The trailing lines mark the trajectory followed by the UAVs	14
Figure 3.3	Desired final state of UAVs wrapped around the plume	15
Figure 3.4	Routing protocol motivation. Red and green nodes are closer to the SD lines and hence more relevant for a successful packet delivery . .	19
Figure 3.5	Transmission zones to restrict flooding between source and destination	19
Figure 4.1	Received signal strength vs distance	30
Figure 4.2	Packet loss vs distance. The packet loss increases from $\approx 5\%$ at 100 m to $\approx 90\%$ at 200 m	30
Figure 4.3	Setup to measure processing delay on BeagleBone Black	31
Figure 4.4	Round trip time between two BeagleBone Blacks	32
Figure 4.5	Single transmission zone with source at <i>sLoc</i> and destination at <i>dLoc</i> . Intermediate nodes use <i>pkt.sLoc</i> and <i>pkt.dLoc</i> to determine their position relative to the transmission zone	37
Figure 4.6	Multiple diverged transmission zones. Intermediate nodes use <i>pkt.tLoc</i> and <i>pkt.dLoc</i> to determine their position relative to the transmission zone	38
Figure 4.7	Back-off time calculation	39
Figure 4.8	Plot of force magnitude calculated using multipart function in Algorithm 12 as a function of inter-UAV distance	42
Figure 5.1	Random distribution of UAVs in the simulation space	44
Figure 5.2	Delivery ratio in mesh formation for flooding algorithm	45
Figure 5.3	Delivery ratio in mesh formation for petal routing	45
Figure 5.4	Delivery ratio in spherical node distribution for flooding algorithm .	46
Figure 5.5	Delivery ratio in spherical node distribution for petal routing	46
Figure 5.6	Delivery ratio in random node distribution for flooding algorithm . .	47
Figure 5.7	Delivery ratio in random node distribution for petal routing	47
Figure 5.8	Average number of hops in mesh formation for petal routing	49
Figure 5.9	Average number of hops in spherical node distribution for petal routing	49
Figure 5.10	Average number of hops in random node distribution for petal routing	50
Figure 5.11	Average number of transmissions per successful packet delivery in mesh formation for flooding algorithm	51
Figure 5.12	Average number of transmissions per successful packet delivery in mesh formation for petal routing	51
Figure 5.13	Average number of transmissions per successful packet delivery in spherical node distribution for flooding algorithm	52

Figure 5.14	Average number of transmissions per successful packet delivery in spherical node distribution for petal routing	52
Figure 5.15	Average number of transmissions per successful packet delivery in random node distribution for flooding algorithm	53
Figure 5.16	Average number of transmissions per successful packet delivery in random node distribution for petal routing	53
Figure 5.17	Compact node alignment for ideal network density	54
Figure 5.18	Delivery rate vs network density	56
Figure 5.19	Average number of hops vs network density	57
Figure 5.20	Average number of transmissions per successful packet delivery vs network density	57
Figure 5.21	Delivery Rate for flooding algorithm with inaccurate destination location in a random distribution of nodes	58
Figure 5.22	Delivery Rate for petal routing with inaccurate destination location in a random distribution of nodes	59
Figure 5.23	Average number of hops for petal routing with inaccurate destination location	59
Figure 5.24	Average number of transmissions per successful packet delivery for flooding algorithm with inaccurate destination location	60
Figure 5.25	Average number of transmissions per successful packet delivery for petal routing with inaccurate destination location	61
Figure A.1	Packet delivery ratio in random node distribution for flooding algorithm for median inter drone separation	69
Figure A.2	Packet delivery ratio in random node distribution for petal routing for median inter drone separation	70
Figure A.3	Average number of hops by successfully delivered packets	71
Figure A.4	Average number of transmissions per successful packet delivery in random node distribution for flooding algorithm	72
Figure A.5	Average number of transmissions per successful packet delivery in random node distribution for petal routing	73

CHAPTER

1

INTRODUCTION

1.1 Aerial networks and the challenges

The use of unmanned aerial vehicles (UAVs) originated in military operations and thanks to the advances in sensors, communication, and embedded systems, it has gradually expanded to civilian applications. Furthermore, the past decade has seen a surge in civil applications of UAVs which has outpaced military deployments, with about a million UAVs being registered with the Federal Aviation Authority [24]. The civilian applications of UAVs are in domains like scientific, recreational, agricultural, product delivery, surveillance etc. and has been classified into four broad categories, namely, search and rescue (SAR), coverage, delivery, and construction [9].

Single UAV systems have been in use for decades, but such systems have some challenging issues like limited communication range, bandwidth, dependency on a single system, operational cost etc. On the other hand, mini-UAVs have restricted capabilities in terms of power, sensing, communication, and computation; however, most of these issues can be mitigated by employing a team of multiple UAVs. Therefore, recently, applications which

employ a group of small UAVs have gained more interest because of the several associated benefits. Compared to a single large UAV working on a mission, a multi-UAV cluster of drones can be employed for efficient and reliable mission outcome because of the UAV sizes, capabilities, maneuverability, little or no threat to human life and buildings/properties. Moreover, a team of UAVs working together has the potential to perform a task that goes beyond the individual capabilities of a single UAV.

However, an autonomously operating cluster of UAVs has its own set of challenges like cooperative path planning, collision avoidance, reliable wireless connectivity among each other and to the base station (in some cases), mission specific quality of service (QoS) requirements etc. At the core of all these problems is the challenge to establish a robust inter-network — in an ad-hoc manner — that facilitates reliable and efficient communication and coordination among the UAVs in the team.

Mobile Ad-hoc NETWORKs (MANETs) and Wireless Sensor Networks (WSNs) are the traditional ways to set-up a network without any infrastructure. The nodes in the network act as the end-hosts as well as the routers to forward the messages. Setting up such networks is a challenging problem but there is an underlying assumption that the nodes are not highly mobile, and the connecting links have some degree of stability. This assumption is relieved in Vehicular Ad-hoc NETWORKs (VANETs), where the nodes are highly mobile, however, the nodes in a VANET follow some pattern (e.g. car following model [23]) and the movement occurs primarily in two dimensions. The solutions from WSNs, MANETs, and VANETs do provide some insights on how to proceed with the challenges of link diversity, QoS requirements, and high node mobility but it is still not clear as to whether networking protocols developed for ground networks can be readily deployed in UAV networks [9].

Ad-hoc networks for UAVs are generally termed as Flying Ad-hoc NETWORKs (FANETs) and a few of the characteristics that sets them apart from other ad-hoc networks are:

- High node mobility in three dimensions
- Heterogenous links: A UAV in a FANETs can be connected to other UAVs (air-to-air) or to a base station (air-to-ground) or even to a satellite (air-to-satellite)
- Primary motivation for their existence: ‘The aerial networks are not just communication networks and as such they have varying yet very specific mission requirements depending on the application’ [9]

In other words, the requirements from a FANET changes from application to application and even during the same application. Therefore, we need to study and revisit the communication requirements and the protocols at each layer of the inter-networking stack of the aerial networks.

There is a rich literature of work on routing protocols for MANETs. Some of the protocols have proposed to extend the traditional protocols of table-based routing in a proactive (keeping redundant paths or continuously checking on link quality) or reactive (find a route when needed) or hybrid (proactive inside a zone and reactive between zones) manner. Other efforts have proposed routing protocols that make use of geographical locations of the nodes. While geographical routing protocols have been experimentally shown to be more scalable and robust than topology-based protocols for MANETs, geographical routing protocols are still not directly applicable to aerial networks where the height of the network is more than the transmission radius of the nodes. This is because some of the key concepts in 2D geographic routing either do not apply in 3D networks or are computationally expensive [4].

1.2 Problem statement

In this thesis, we study the problem of routing packets in an aerial three-dimensional ad-hoc network with highly mobile nodes, where maintaining network topology either proactively or reactively is not feasible. Our approach is to use the inherently broadcast nature of the wireless medium as an advantage by making the intermediate nodes do an ‘opportunistic’ forwarding towards the destination node or region. We have studied the qualitative and quantitative requirements from a network for the plume wrapping and tracking problem [12], as a sample application of autonomous multi-UAV missions. We present an extended version of a previous work by this research group on geographic routing that was termed as ‘Petal Routing [3]’ and its applicability to the plume wrapping and tracking mission at hand. We show that our proposed protocol matches the reliability of a network-wide flooding algorithm while significantly reducing the overhead. Finally, we present a systematic study of the protocol’s performance based on metrics like transmission overhead, average number of hops, delivery ratio, the effect of network density.

1.3 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 introduces aerial networks its characteristics, applications and some of the popular routing techniques related to them. In Chapter 3 we have laid out the plume tracking mission and the requirements from a communication viewpoint, thereafter we have presented our routing scheme. In Chapter 4, we present our extensions in QrSim to implement multi-hop routing. Thereafter, in Chapter 5 we have studied the performance of our algorithm on various metrics. Finally, we present the conclusions and discuss some of the possible improvements in Chapter 6.

CHAPTER

2

BACKGROUND

2.1 Flying Ad-hoc networks (FANETs)

A wireless ad hoc network is a decentralized network of nodes where there is no specialized equipment like routers/switches/access points to facilitate communication, and the communication medium uses wireless protocols e.g. IEEE 802.11. Since there is no need to set-up any special infrastructure the network can be spun up quickly. Generally, nodes are mobile (hence termed as Mobile ad-hoc network - MANET) and depending on use-cases, node type or node mobility they are further classified as Vehicular ad hoc networks (VANETs) or Flying ad hoc networks (FANETs) or Smart-phone ad hoc networks (SPANs).

Flying ad hoc networks are a subset of MANETs and consist of nodes capable of flight. The nodes have high degree of mobility, and the distance between the FANET nodes is larger as compared to MANET nodes. These nodes can be manned airplanes, unmanned guided drones or unmanned unguided drones. The span of the network also varies from few hundred meters to several kilometers. One important distinction about FANETs is that the participating nodes have a common mission and their mobility model depends

Table 2.1 Comparison of different type of FANET nodes

Properties	Mini UAVs	Small UAVs	Large UAVs
Transmission Range	Small	Medium	Large
Altitude	about 300 m	about 2000 m	> 3000 m
Speed	Medium	High	Very High
Energy Autonomy	Restricted (Battery Dependent)	Not Restricted (On-board solar panels or fuel)	Not Restricted (On-board solar panels or fuel)

on the mission [21]. In this section, we discuss the different characteristics, applications, advantages and challenges of FANETs.

2.1.1 FANET Characteristics and Applications

2.1.1.1 FANET Characteristics

FANETs differ from MANETs on several characteristics. Some of them are as follows.

Node Type and mobility: Typical nodes in a FANET would be unmanned aerial vehicles (UAVs) which can be further differentiated based on their size, namely, large, small and mini UAVs [21]. Their mobility, transmission range, altitude, speed, and energy autonomy depends accordingly on their corresponding sizes. A comparison between these types of UAVs is presented in Table 2.1 [21].

Network Connectivity and QoS requirements: Depending on the UAV density (number of UAVs is a unit volume) and the node mobility, the connectivity varies. Although the node movement is predefined according to the mission, it can be updated during the mission due to sensor errors, environmental conditions or change of mission altogether. These affect the link quality and efficiency of routing protocols. QoS requirements also differ on a mission basis. Some missions are network delay tolerant while others are not. Depending on the applications, some FANETs can aim to optimize different subsets of QoS parameters like latency, bandwidth, packet loss, jitter etc.

Table 2.2 UAV Applications

Parameters	Multi-UAV cooperation	UAV-to-ground tasks	UAV-to-VANET collaborations
UAV density	High	Medium	Low
Ground Environment	Not aware	Aware	Aware
Infrastructure	No	Yes	Maybe
Obstacles effect	Low	High	Medium
Task Duration	Limited	Not Limited	Not limited
Human Operator	No	Yes	Maybe

2.1.1.2 FANETs applications

Application requiring FANET can be grouped into three categories based on the entities in the network and cooperation model among them [2].

Multi-UAV cooperation: Certain tasks require the UAVs to autonomously complete the mission in a time limit without human interaction. In such applications, there is no awareness of the ground environment and thereby doesn't need any ground infrastructure. The UAVs cooperate and coordinate according to the predetermined application and correct the flight path in response to the environmental factors and errors. Example applications would include plume wrapping, target detection, monitoring disasters etc.

UAV-to-ground collaboration: In certain applications, the UAVs need to periodically relay data to a ground location so that it can be acted upon in real time. Such applications require ground infrastructure which can be either mobile or fixed. Examples include search and rescue missions, military monitoring etc.

UAV-to-VANET collaboration: In the third kind, the aerial UAVs can communicate and coordinate with vehicles on the ground to accomplice certain tasks. Such applications may require ground infrastructure and human involvement. This type of hybrid ad hoc network is a relatively new kind of wireless communication. Examples include route guidance, traffic monitoring, data packet delivery etc.

The above information is summarized in Table 2.2

2.2 Routing in 2D ad hoc networks

In a large ad hoc network all the nodes might not be in the direct transmission range of each other, hence routing techniques are needed for successful communication between distant nodes. Node mobility and wireless medium leads to frequent topology changes and link failures which makes routing in MANETs a challenging task.

The common MANET routing protocols are generally grouped into two broad groups, namely, topology-based and position-based routing [4]. In this section, we will discuss some examples of these protocols and their properties.

2.2.1 Topology-based routing protocols

These approaches are like destination IP address-based routing. Topology-based routing protocols maintain a network graph and a path for each destination. The network graph creation can be further classified as proactive, reactive and hybrid approaches.

Proactive approach: In this routing approach, a node maintains the path to each node in the network through periodic updates to other nodes. Since the path is already known the packets can be immediately forwarded towards the destination. The downside is that due to high mobility the link information becomes stale quickly which leads to routing failure. The overhead of maintaining a dynamic network makes this approach difficult to scale.

Reactive approach In this routing approach, a node initiates path discovery when it needs to reach a destination. The node sends out route discovery request messages to other nodes (the node might keep the route information for future use) as needed.

Hybrid approach In hybrid approaches, the network is divided into zones. For intra-zone communications proactive approach is adopted while for inter-zone communications a reactive approach is adopted. While topology-based approaches guarantee packet delivery if the destination is reachable, they have a high maintenance cost in terms of memory and communication overhead. Moreover, topology-based approaches are more suitable for relatively static networks compared to MANETs.

2.2.2 Position-based routing protocols

These approaches route packets based on the geographic position of the destination. In highly dynamic networks, position-based routing approaches have been experimentally confirmed to be more scalable than those which don't use geographic information [25]. Geographic routing approaches don't need to share and store link information - rather just the immediate radio neighbors - and thereby make routing decisions on a hop-by-hop basis. Whereas, position-based routing requires the nodes to know their own location, which can be obtained by a location system such a GPS. The accuracy of a GPS might not be enough for multi-UAV systems which are highly mobile at very high speed. Moreover, GPS provides location updates every second which might not be sufficient for several multi-UAV missions. To overcome this issue, the nodes can be equipped with Inertial measurement units, which can be calibrated by GPS signals.

Position-based routing protocols are generally classified on two properties, 'location service' and 'forwarding strategy'.

2.2.2.1 Location Services

Besides their own location, the nodes may need to know the location of destination nodes. Most position-based routing protocols employ a location service to maintain and distribute the location of a node. The location services can be classified based on the number of nodes that maintain the location information and for the number of nodes for which the location servers maintain the information. Thereby, the four possible combinations can be [19]

Some-for-some: Some nodes in the network maintain and provide the location information for some of the nodes.

Some-for-all: Some nodes in the network maintain and provide the location information for all the nodes.

All-for-all: All nodes maintain the location information for every other node.

All-for-some: All nodes maintain the location information for a subset of the nodes. Moreover, there are different methods to distribute location information. A few of the proposed methods are flooding [1], grid location service [15] and quorum-based location service [8].

2.2.2.2 Forwarding Strategies

After a node has obtained the location of the destination node, there needs to be a forwarding/routing strategy to deliver the packet. Forwarding strategies in geographical routing can be grouped as follows:

Greedy forwarding Assuming a source node S needs to send a packet to a destination node D, where S and D are not in the direct transmission range of each other, node S will pick an intermediate node T from among the nodes in the transmission range of node S such that T is the closer to node D than S. Node T will follow the same procedure until a penultimate node which had D in its direct transmission range receives the packet and forwards the packet to D. Greedy routing strategies are guaranteed to be loop-free (since the packet never travels backward), however, greedy strategies don't guarantee packet delivery even though there is a path between source S to destination D. It happens when a node can't find an intermediate node which is closer to the destination than the node itself. Nodes in such situation are referred to as concave nodes. There are several variations of greedy forwarding and the recovery strategies presented in the literature [21] [4] [19] [25].

Face routing: In this technique, faces on a planar graph are traversed following a technique known as 'right hand rule', where the algorithm keeps track of all the times it crosses the line connecting the source to destination. After covering an entire face, the algorithm moves onto one of the intersections that are nearest to the destination. This continues until the destination is eventually reached [14] [4]. Face routing is also a common recovery strategy in greedy routing strategy. Although the complexity of face routing is higher than greedy routing, face routing guarantees delivery - where possible. Various modifications to the face routing strategy are presented in [4].

Hybrid Greedy-Face Routing: While Greedy routing has less algorithmic complexity, face routing guarantees delivery. Hybrid approaches combine the two approaches i.e. it uses greedy routing while it is possible and then resorts to face routing when it encounters a routing void (local maxima). The primary challenge in this approach is to determine the exact moment when the algorithm should switch back to greedy routing. If the switch is too early, then the algorithm might get stuck in another local maxima, but switching too late will lead to inefficient face routing for too long. The protocols that tackle this problem have been surveyed in [4].

Another approach that doesn't use either one of these approaches was presented in Location Aided Routing [13]. This approach determines a 'request zone' and an 'expected

zone' according to the packet parameters and then forwards the packet in that region.

2.3 Routing in 3D ad hoc networks

FANETs are characterized by high mobility which leads to a frequent change in the underlying terrains. Not only this may introduce frequent topology changes but also terrain-induced blind spots which affect the wireless channel. The wireless channel is also affected by the antenna orientation - which changes frequently due to high mobility - and thereby causes link fluctuations. Moreover, communication requirements in a FANET are mission specific and can widely vary from one type to another. For example, in a search and rescue mission, the information should be transmitted to the base station in real-time while some surveillance missions might choose to gather and store on board and delivering it to the base after the mission has completed or in batches.

These characteristics are peculiar to FANET and hence require an exclusive study of routing protocols which shall be particularly suited to aerial networking. However, most of the proposed geographic routing protocols and techniques have been developed with 2D networks in mind. Thus, these protocols are applicable for networks where the height of the network is less than the transmission radius of the nodes. However, they do not extend easily to 3D networks. For example, face routing would fail to guarantee delivery in 3D networks because planar sub-graph and face perimeters are only applicable to 2D planes [16].

One of the proposed approaches is the Unit Ball Graph [6], which extends the Unit Disk Graph approach. The authors in [6] [5] have concluded that a deterministic recovery strategy is impossible in 3D networks (as opposed in 2D networks) and hence they have proposed a randomized recovery strategy in their localized geographic routing algorithm. Some other 3D routing protocols have been summarized in [2] [10].

CHAPTER

3

FANET ROUTING PROTOCOL

Each multi-UAV mission has its own set of requirements in terms of size and number of vehicles, the size of the mission's region, payload, mission duration, environmental constraints, level of autonomy and mobility requirements. These all mission-oriented requirements - in turn - generate a peculiar set of demands from the communication infrastructure. Therefore, we will first present an overview of our multi-UAV mission at hand, the communication requirements and then a protocol that facilitates those primitives.

3.1 FANET Mission

The use of multi-cluster UAVs to track gas plumes like volcanic eruptions, forest fires, and environmental contamination has increased in the previous decade. In a plume tracking mission, the UAVs would be required to wrap around the plume, report the contamination level and possibly track the plume by moving along with it. Once the UAVs have been deployed they have complete device autonomy (i.e. each UAV decides its future waypoints and velocity itself using the inputs from onboard sensors and doesn't need these instruc-

tions from a human operator) and mission autonomy (i.e. the UAVs must cooperate and coordinate with each other for the successful completion of the mission). It should also be noted that although the UAVs coordinate in the decision-making process, a network-wide consensus might not be required for a specific UAVs operation. As the plume moves in time and space the UAVs should evenly spread out on the surface of the plume while maintaining a safe distance from each other and avoiding any static or dynamic obstacle in their path.

A detailed description of the problem and a multi-UAV distributed solution has been provided in [12]. In the proposed solution the authors have assumed that the mission starts with an initial formation of drones in a 2-dimensional grid (shown in Fig. 3.1). Thereafter, the UAVs move towards the plume autonomously without any ground support. Whenever a UAV detects a contamination it stops and maintains its position with respect to the plume while rest of the UAVs continue their search. If the inter-UAV distance crosses a threshold (too close or too far away) then the moving drone tries to move further or closer to the stationary UAV. These inter-drone forces maintain an approximate mesh structure while resulting in the UAVs wrapping around the plume. As an aid to visualizing this process, think of the mesh as a piece of cloth moving towards a spherical ball and eventually wrapping around the ball. Fig. 3.2 shows an intermediate state with drones wrapping around the plume and Fig. 3.3 shows a desired final state of UAVs which are wrapped around the plume.

3.2 Communication requirements

In this section, we shall outline the qualitative and quantitative requirements from a communications viewpoint for the successful communication of the mission. We shall use these requirements to design the routing protocol, the message primitives and eventually as a yardstick to measure the performance of the proposed protocol. It should also be noted that the requirements are geared towards a specific mission and since the UAV applications are diverse, these communication demands, and requirements shall widely vary for different applications. For a comprehensive analysis of various civil UAV applications, we refer the readers to [9], where the authors have classified the civil applications of UAVs into four broad categories, namely, Search and Rescue, Area or Network Coverage, Construction, Delivery/transportation and presented the requirements from a communications viewpoint.

Mission coverage: Applications like plume tracking, wildfire monitoring or plume wrap-

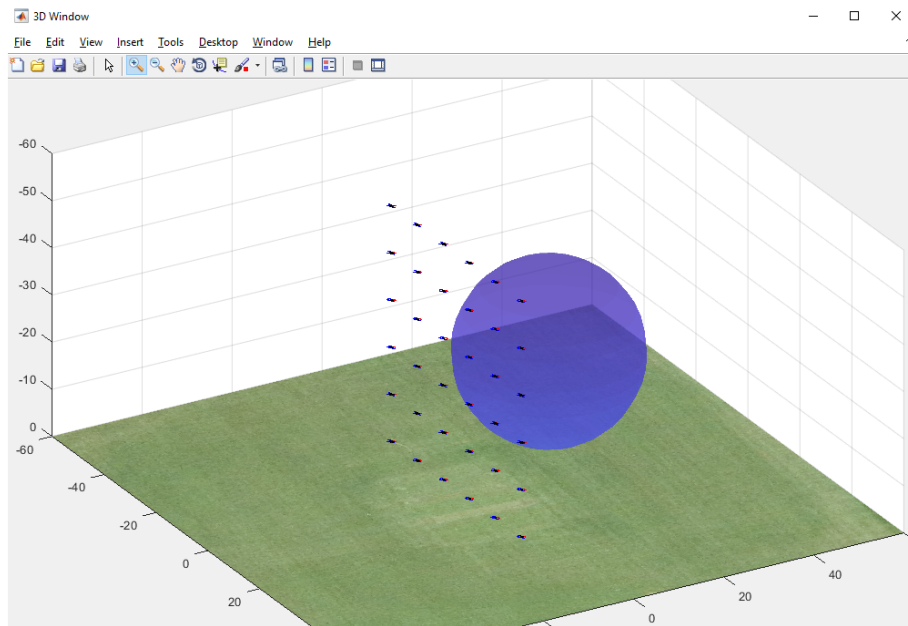


Figure 3.1 Mesh formation of UAVs at the start of the mission

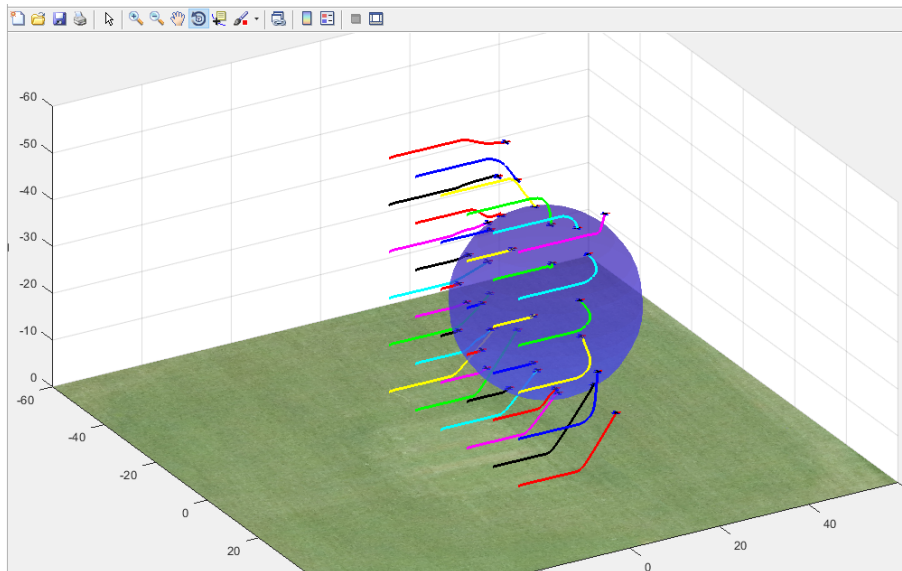


Figure 3.2 UAVs wrapping around the plume. The trailing lines mark the trajectory followed by the UAVs

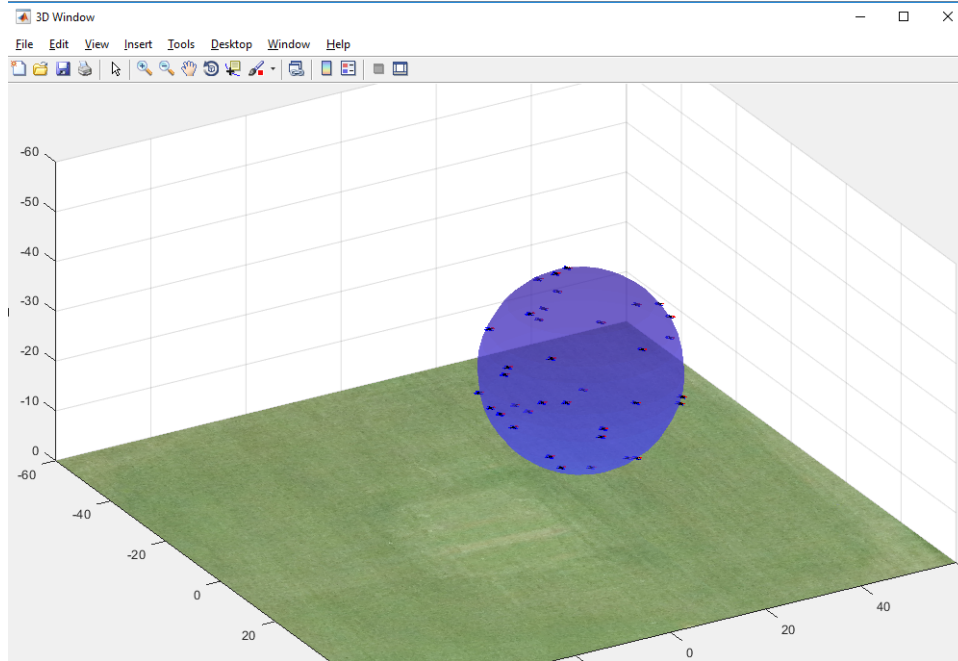


Figure 3.3 Desired final state of UAVs wrapped around the plume

ping typically span a medium to large sized area in the order of tens of kilometers [9]. Furthermore, the aerial network should consider the coverage volume expansion or shrinkage with time. The mission volume that we shall be considering would be $\approx 500m \times 500m \times 500m$.

Network requirement: Since the mission requires coordination among the participating UAVs, any two participating UAVs should be able to reliably communicate with each other (either directly or multi-hop connectivity). Moreover, the network should be able to reconfigure itself, in other words, the network should consider the nodes joining and leaving the network while the mission is in progress.

Number of nodes: The number of UAVs required in a mission depends on the mission area, transceiver characteristics and the type of nodes employed. Assuming off-the-shelf Wi-fi transceivers and the above mentioned mission coverage the required number of nodes would be around 30.

Connectivity to a base station: In a real-life deployment scenario, the UAVs must be connected to a base station for safety and security purposes, although the data-rate requirement of the link can be mission specific. In our case, the UAVs need to send their telemetry information (GPS and IMU data) to the base station, however since the mission is fully autonomous with distributed decision-making capacity the entities don't need to relay

the coordination data to the base station. A frequency of 4-5 Hz or less is the standard for telemetry data exchange [9].

Connectivity between the other UAVs: Since the UAVs are required to maintain an approximate mesh structure they need to approximate the distance with other drones. Therefore, the UAVs need to update each other of their GPS locations in real-time. A suitable frequency of this data exchange could be 4-5 Hz. Also, in real deployment scenarios, the UAV cluster can be composed of drones with varying capabilities (UAVs have different sensors) which demands a need for one-to-one communication. Therefore, the UAVs must have a reliable connectivity to each other in either a single or multi-hop fashion.

Collision avoidance: Once a UAV detects another UAV (e.g. via RADAR) in its flight path, they both need to navigate around each other to avoid a collision. To check if a future waypoint is void of other UAVs, the source UAV can send an anycast to that specific destination region. If some other UAV happens to be in the region then they should negotiate their way around. It should be noted that the originator UAV doesn't know or care about the identity of the drone, rather the message is addressed to the geographical region. This requires a capability to geocast packets where the destination location is the address. These messages are not delay-tolerant and need to be delivered in real time.

Group communication: Sometime a group of UAVs which are a subset of all the UAVs in the network might need to act in accordance of each other. For example, while tracking the plume the UAVs would organize into smaller groups and then move as a cluster around the plume. This would require a mechanism to reach a consensus regarding the group formation, joining or leaving a group and a leader election by the members of the group.

Data type and data rate demands: The traffic in the aerial UAV network consists of control data (remote controlled data exchange), telemetry data (GPS and IMU data), coordination data (waypoint, mission plan exchange) and sensed data (sensor data). The actual traffic demands depend on the sensor-on-board the UAVs, the type of traffic exchanged in the network and the frequency of such exchange. As per [27], imagery and chemical data would need to be collected which would require chemical analyzer, an infrared sensor, image or possibly a video camera. The sensed traffic is sensor dependent and can be, low-rate, bursty or high-rate. In our case, the UAVs need to exchange coordination data among each other, sensed data with the base station and telemetry data with each other as well as the base station. The data exchange with the base station can be periodic and delay-tolerant but the coordination data among the drones must be real-time. As far as

Table 3.1 Communication requirements

Requirements	Values
Mission Coverage	Medium $\approx 500m \times 500m \times 500m$ or Large
Network mode	ad hoc
Number of nodes	≈ 30
Coordination data exchange frequency	4-5 Hz [9]
Sensed data exchange frequency	30 Hz for video, 1 Hz for IR and chemical analyzer [9]
Telemetry data exchange frequency	4-5 Hz [9]
Coordination data link rate (up/down)	4.8 /64 kbaud [9]
Sensed data link rate (up/down)	4.8-9.6/64 kbaud for chemical data [27] 1-2 Mbps for visual data.
Delay	50-100 ms [9]
Traffic type	periodic and real time (coordination data, telemetry data), delay tolerant (sensed data)

data rate is concerned 1 Mbps for images and 2 Mbps for video streaming is sufficient with a typical delay of not more than 50-100 ms [9].

Scalability: The number of nodes can be increased due to two reasons. (1) Increase the coverage volume (2) Increase the density of nodes to get a high-resolution sensory data. A scalable network should handle the increase in a number of nodes/devices in the network and the performance should not degrade unreasonably.

Localization and time synchronization: The UAVs need to know and report their own location with an accuracy of $\approx 5m$. This level of accuracy is achievable with off-the-shelf GPS devices and can be improved to a few centimeters by using dual-frequency receivers and/or augmentation systems [7]. The UAVs can also synchronize their clocks through the information provided by GPS.

The above information is summarized in Table 3.1.

3.3 Routing protocol

3.3.1 Protocol description

As we discussed in Chapter 2, proactively or reactively maintaining routes in tables is not a suitable solution for routing between highly mobile FANET nodes. Therefore a routing scheme which does not maintain end-to-end paths and doesn't employ next-hop forwarding would be more robust. On the other hand, a routing scheme that always floods a packet (nodes discard duplicate receptions to avoid routing loops) in the network doesn't need to maintain any information and is highly robust (i.e. flooding guarantees delivery if the destination node is reachable). However, the associated overhead makes it unsuitable for large and frequent data transfers.

A key observation that reduces the number of re-transmissions in flooding is that every node in the network doesn't need to forward a packet for successful delivery; provided some nodes which lie in the direction of the destination node have received and transmitted the packet. For example let's consider a 2 dimensional representation in Fig. 3.4, with source node 'S' and destination node 'D' the nodes with *black* dots don't need to transmit if the nodes with *red or green* dots have received and transmitted the packet. It should be noted that the red and green dots are closer to the straight line connecting the source and destination node as compared to the black nodes. In this regard we signify the volume enclosing these '*more relevant*' nodes as the '**transmission zone**'. Therefore, if a source node 'S' knows the location of a destination node 'D' then 'S' can define a transmission zone and restrict re-transmissions to the nodes which lie in the zone, thereby reducing the overhead. This idea of constrained flooding is derived from the concept of the petal in [3] and that of request zone in [13].

These 'transmission zones' can be defined as different shapes, for example a capsule, cuboid or a spheroid Fig. 3.5 and each have their own peculiar properties. In other words our routing scheme is a restricted flooding algorithm that reduces the overhead by restricting the re-transmissions to a volume between the source and destination nodes.

We also observe that there is an opportunity to further reduce the number of retransmissions. For example in Fig. 3.4 if all the nodes with red dots receive and transmit the packets then even if the nodes with green dots don't re-transmit, the packets shall be successfully delivered to the destination. It should be noted that the red nodes are closer to the source-destination line and also closer to the destination node as compared to the green nodes.

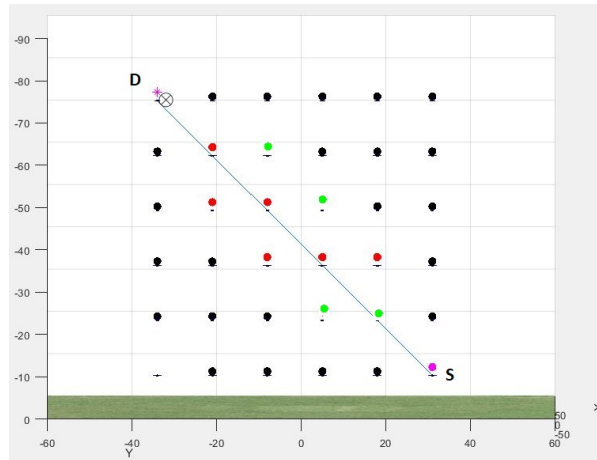


Figure 3.4 Routing protocol motivation. Red and green nodes are closer to the SD lines and hence more relevant for a successful packet delivery

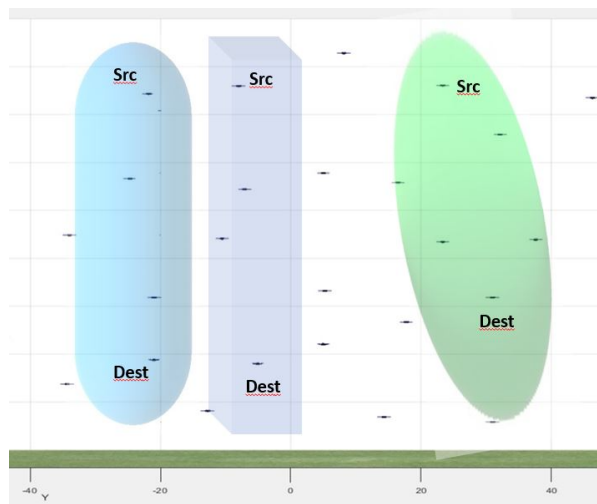


Figure 3.5 Transmission zones to restrict flooding between source and destination

This ‘opportunistic’ selection of the nodes can be achieved by introducing a ‘**back-off and re-transmit**’ mechanism. Specifically, when an intermediate node ‘N’ receives a packet ‘p’, then ‘N’ calculates a ‘ B_{off} ’ time, appends the packet in a waiting queue and registers a callback for the expiry of ‘ B_{off} ’ time. In the meanwhile ‘N’ registers the locations from where ‘N’ heard duplicate transmissions of ‘p’. After ‘ B_{off} ’ has expired, the node ‘N’ determines the centroid of duplicate transmission locations and if the centroid is closer to the destination compared to the node itself then the node ‘N’ drops the packet otherwise ‘N’ re-transmits the packet. The algorithm to calculate the ‘back-off’ time is presented in Section 4.2.7.

Algorithm 1: Schedule received packet: Petal Routing

```

Schedule (pkt)
  if pkt.pId ∈ transmittedPktIdSet then
    discard(pkt)
    EXIT
  if insideTransmissionZone(pkt, myLoc) == TRUE then
    if pkt.pId is in bufWait then
      Increase duplicate count for pkt.pId
      save pkt.tLoc
    else
      boffTime = calculateBoffTime(pkt)
      add pkt to bufWait
      registerCallback(boffTime, pkt)

```

To this extent, following are the assumptions underlying our routing scheme:

Assumptions:

1. The wireless antennas on the nodes are omnidirectional.
2. The nodes are reachable from each other i.e. a flooding algorithm can guarantee packet delivery between any pair of nodes and we use the delivery rate of flooding algorithm as an upper bound for our algorithm.
3. The nodes always know their location via GPS.

Algorithm 2: Transmit or Drop packet: Petal Routing call-back handler

```
TransmitOrDrop (pkt)
  dupCoords = duplicateTransmissionCoordinates(pkt)
  if length(dupCoords) > nodeDensity * boundary(dupCoords) then
    centroid = centroid(dupCoords)
    if distance(myLoc, pkt.dLoc) < distance(centroid, pkt.dLoc) then
      | discard(pkt)
    | pkt.tLoc = myLoc
    | transmit(pkt)
  else
    | pkt.tLoc = myLoc
    | transmit(pkt)
  | Add pkt.pId to transmittedPktIdSet
```

4. The nodes synchronize their clock via GPS.
5. A source node knows an approximate location of the destination node which is maintained in a location table and provided by a location service. Different implementations of a location service have been mentioned in Section 2.2.2.1 and our implementation has been explained in Section 4.2.5.
6. The contention resolution and scheduling issues at the hop time scale are handled by the MAC layer

In a nut-shell, our routing scheme benefits from the inherently broadcast wireless medium and restricts the re-transmissions to a specific volume around the source - destination line and cancelling out redundant transmissions in the process. This routing scheme is presented in Algorithm 1 and 2. A 2D equivalent of our routing scheme is presented in [3] and the routing scheme is termed as ‘petal routing’. Although, all our references are in 3 dimensions, we shall continue the same nomenclature and refer our 3D routing scheme as ‘Petal routing’.

3.3.2 Message primitives

As discussed in Section 3.2, the common scenarios when a UAV would need to use the communication service would be the following:

- Distribute telemetry data (e.g. GPS, IMU data) with all the UAVs in the network periodically.
- Notify all the UAVs in the network of a critical event (e.g. high plume concentration at a location).
- Relay telemetry and sensed data to the base station periodically.
- Exchange coordination data (e.g. Future waypoint) to UAVs in a specific direction (e.g. to negotiate conflicting paths).
- Communication among a group of UAVs for cluster formation, leader election, request to join or exit the cluster.
- Coordinating among the cluster for a consensus on the cluster's behavior.

Now we shall present the message primitives that shall facilitate the above communications.

1. *NetworkWideFlood(message, sourceID)*: Deliver the message to all the reachable nodes in the network.
2. *HopLimitedFlood(message, sourceID, HTL)*: Deliver the messages to the neighbouring nodes which are HTL hops away.
3. *Unicast(message, sourceID, destinationID)*: Deliver the message to a specific node.
4. *Geocast(message, sourceID, destinationCoordinates, radius)*: Deliver the message to all the nodes in the specified region.
5. *Multicast(message, sourceID, nodeIDs)*: Deliver the message to all the nodes in the *nodeIDs* list.

Here *sourceID* and *destinationID* are unique identifiers for a node - possibly IP addresses, while *destinationCoordinates* together with *radius* represent a spherical region with specified coordinate as the center and specified radius. *nodeIDs* is a list containing the ID of nodes in a group.

We will now present a high-level algorithmic description of these message primitives.

3.3.2.1 Network Wide Flood primitive

Network wide flood primitive ensures that the packet shall be received by all the nodes that are reachable. When a source node 'S' needs to send a message to every node in the network, it creates a header with the following parameters and encapsulates the data in this header.

$$header = Header(destId = FLOOD)$$

Node 'S' then broadcasts this message to all its neighbors in the wireless medium. An intermediate node 'N', on receiving the message for the first time reads the contents and rebroadcasts it. Thereafter, 'N' discards any duplicate receptions of the message. This guarantees that the flooding is loop-free. The benefit of Flooding is the protocol's robustness however there is a high overhead associated with flooding and hence it is suitable for small packets only.

3.3.2.2 Hop Limited Flood

The primitive allows to restrict the span of flooding by the 'Hops to Live(HTL)' parameter. For example, if a source node 'S' wants to send a message to all its immediate neighbors (i.e. in direct radio range) then 'S' shall set the HTL value to 1 in the header. Conceptually, HTL is equivalent to 'Time To Live (TTL)' in an IP network.

$$header = Header(destId = FLOOD, HTL)$$

It should be noted that hop limited flooding can be used for a network wide flooding by setting HTL equal to infinity (e.g. 15 is assumed to be infinity in RIP). We represent infinity by the macro NETWORK_DIAMETER.

The part of receive algorithm that deals with FLOOD packets is depicted in Algorithm 3

3.3.2.3 Geocast Primitive

Geocast is used when a UAV wants to communicate with any node present in a geographic region. This is useful in scenarios where two nodes detect a conflicting path and need to

Algorithm 3: Receive(msg): Flood

```
Receive (msg)
  if msg.pId ∈ transmittedMsgIdSet then
    discard(msg)
    EXIT
  if msg.destId == FLOOD then
    if msg.pId ∉ seenIdSet then
      msg.HTL = msg.HTL - 1
      Add (seenIdSet, msg.pId)
      if msg.HTL > 0 then
        transmit(msg)
      else
        discard(msg)
    else
      ...
```

negotiate their way around each other.

When a source node 'S' needs to send a message to *any* node in a spherical region 'R' with center at '*dLoc*' and radius '*r*' then 'S' creates a header with the following parameters.

$$header = Header(destId = ANY, dLoc = [x, y, z], radius = r)$$

Source node 'S' then encapsulates the data in this header and transmits the packet. An intermediate node 'N' processes the message according to the Algorithm presented in 4

3.3.2.4 Unicast primitive

Unicast is used when a source node 'S' needs to send a message to a destination node 'D'. Unicast uses the same underlying mechanism as geocast to route a message except that 'S' first consults its location table for '*dLoc*' - the location of destination 'D' - and creates a header with the following parameters.

$$header = Header(destId = destID, dLoc = dLoc)$$

Algorithm 4: Receive(msg): Geocast

```
Receive (msg)
  if msg.pId ∈ transmittedMsgIdSet then
    discard(msg)
    EXIT
  if msg.destId == ANY then
    if insideDestinationRegion(msg, myLoc) then
      Transmit(msg)
      Add msg.pId to transmittedPktIdSet
      EXIT
    else
      Algorithm 1
```

Source node 'S' then encapsulates the data in this header and transmits the packet. Moreover, in Unicast the destination is a particular node instead of a regions. An intermediate receiving node 'N' follows the Algorithm 5.

Algorithm 5: Receive(msg): Unicast

```
Receive (msg)
  if msg.pId ∈ transmittedMsgIdSet then
    discard(msg)
    EXIT
  if msg.destId = myId then
    if msg.ackReq == TRUE then
      ackRep = prepareAckReply()
      send(ackRep);
    else
      algorithm 1
```

3.3.2.5 Multicast primitive

In our routing scheme, *multicast* employs *unicast* at application layer i.e. converting a multicast into multiple unicasts — each one directed to a specific node in the group. Thereby, the receiving end of the algorithm is the same as Algorithm 5 whereas the sending part is as depicted in Algorithm 6

Algorithm 6: Send(msg, nodeIDs): Multicast

```
Send (msg, nodeIDs)  
  foreach nodeId ∈ nodeIDs do  
    unicast(msg, nodeId)
```

CHAPTER

4

IMPLEMENTATION

4.1 Simulator Introduction: Qrsim

QRSim [20] is a multi-vehicle software simulator developed at Department of Computer Science, University College London. QRSim allows a set of UAVs to communicate and cooperate to achieve common goals. It simulates the dynamics of the UAVs as well as the sensors (GPS, IMU, camera) and the inaccuracies in the sensors and environment (e.g. wind, GPS errors). The simulator also includes the implementation of some specific task scenarios. For example, a pursue evasion game, a search and rescue mission etc. The simulator is specifically helpful in simulating general higher-level tasks that involve multiple platforms which sense and react in their environment. The QRSim has been implemented in MATLAB and the source code is available on GitHub [17] under modified BSD license.

We now present a brief overview of the main concepts of the simulator and the relevant classes in its implementation [18].

Platforms represent quadrotor dynamics and sensor models and other platform specific phenomena like aerodynamic turbulence. Platforms subclass the abstract class Platform

(defined in */qrsim/platforms/Platform.m*)

Environment objects represent the phenomena that have a direct or indirect impact of the platforms but are not specific to the platforms like wind or the satellite vehicles of the GPS system or the obstacles in the flight path. Environment objects subclass *EnvironmentObject* (defined in */qrsim/environment*).

State class maintains the state of all the objects taking part in the simulation. For example, the state structure stores variables like ‘the simulator time (t)’, ‘pseudo-random number generator streams ($rStreams$)’, ‘the simulator time step (DT)’, ‘the handle to the 3D graphics visualization ($display3D$)’.

Steppable is an abstract class that represents every object that evolves with time. For example, the location of UAV updates after each time quantum (DT). Hence the method *step()* exposed by the Steppable class is called which called *update()* on the UAV.

Task class allows defining a variety of UAV scenarios and the various objectives for the platforms. The Task class provides a way to derive task objects that specify a scenario and an activity. It exposes methods like *init()*, *updateReward()*, *reward()*, *reset()* and *step()*.

Other abstract classes: QRSim API defines several other abstract classes like *AerodynamicTurbulence*, *Sensors*, *AHARS*, *OrientationEstimator*, *Gyroscope*, *Altimeter*, *Accelerometer*.

The main QRSim class defines methods to initialize, set up and control the simulator, namely (a). *init('taskName')* (b) *qrsim.reset()* (c) *qrsim.resetSeed()* (d) *step(U)*.

4.2 Qrsim Extensions

To simulate the message routing protocol and the plume wrapping mission, we had to add/extend some features. In this section, we describe our additions/extensions and our rationale behind our decisions.

4.2.1 Radio propagation model

Our propagation model is based on free space model derived from the Friis transmission formula. [26]

The received signal strength R can be calculated as

$$R = T - L + N \tag{4.1}$$

Where,

- R : Signal strength at the receiver.
- T : Signal strength at the transmitter. (Can be different for each UAV)
- L : Path Loss
- N : Normal random variable with mean 0 and standard deviation σ

And Path Loss (L) is calculated by the free space path loss equation.

$$L(dB) = 20 * \log_{10}d + 20 * \log_{10}(f) + 32.44 - Gt_x - Gr_x \quad (4.2)$$

Where

- d : the distance between the transmitter and receiver (Km)
- f : the frequency of transmission (MHz)
- Gt_x : the transmitter antenna gain
- Gr_x : the receiver antenna gain.

Thereafter, Message Reception Probability (r) = $P(R > R_{th})$. Where r_{th} is the threshold for successful reception of a message. Appropriate values of T, R, R_{th} can be plugged in from the data sheet of WL18xxMOD WiLink™Wi-Fi®[11] (Section 5.6 and 5.7 in the data sheet) used in BeagleBone black.

The values used in our simulation are mentioned in 4.1 and the corresponding received signal strength plot and packet loss plot are shown in Fig. 4.1 and Fig. 4.2:

The function to get the signal strength at the destination is Algorithm 7

4.2.2 Processing delay

In simulating message transmission we have ignored propagation delay and transmission delay in our calculation. This is because our estimate of packet processing delay at a UAV is much smaller in comparison to the time quantum (DT) in which the QRSim simulator updates the simulation state. DT is a configurable parameter with the default value of 0.2 seconds. Below we present an estimate of the propagation delay, experimental setup and

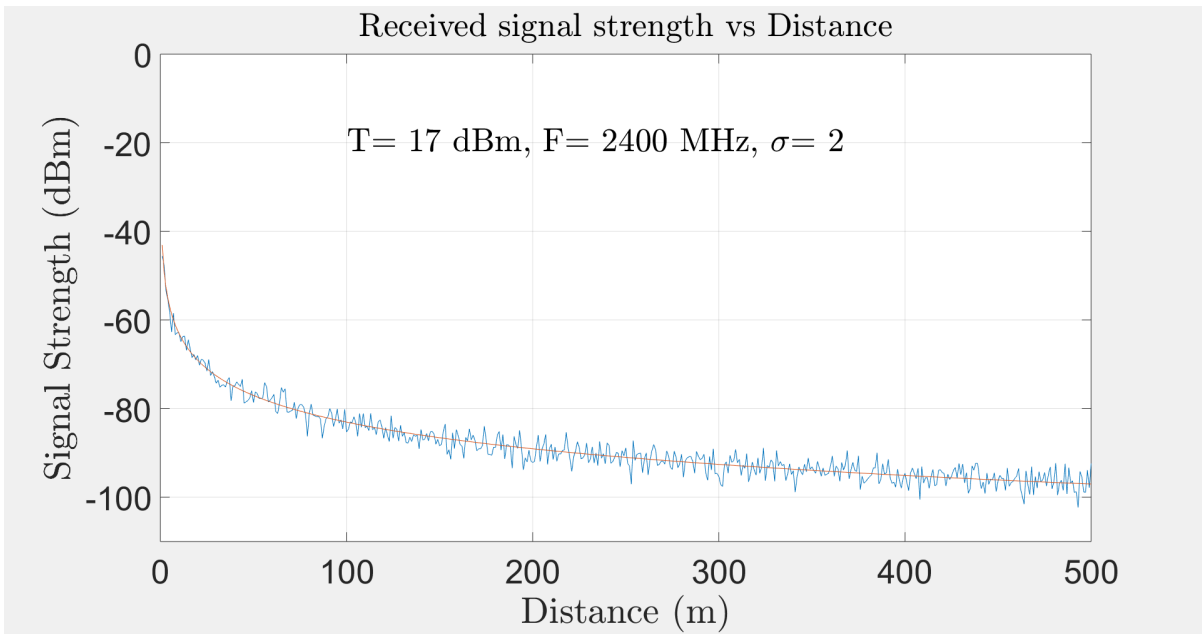


Figure 4.1 Received signal strength vs distance

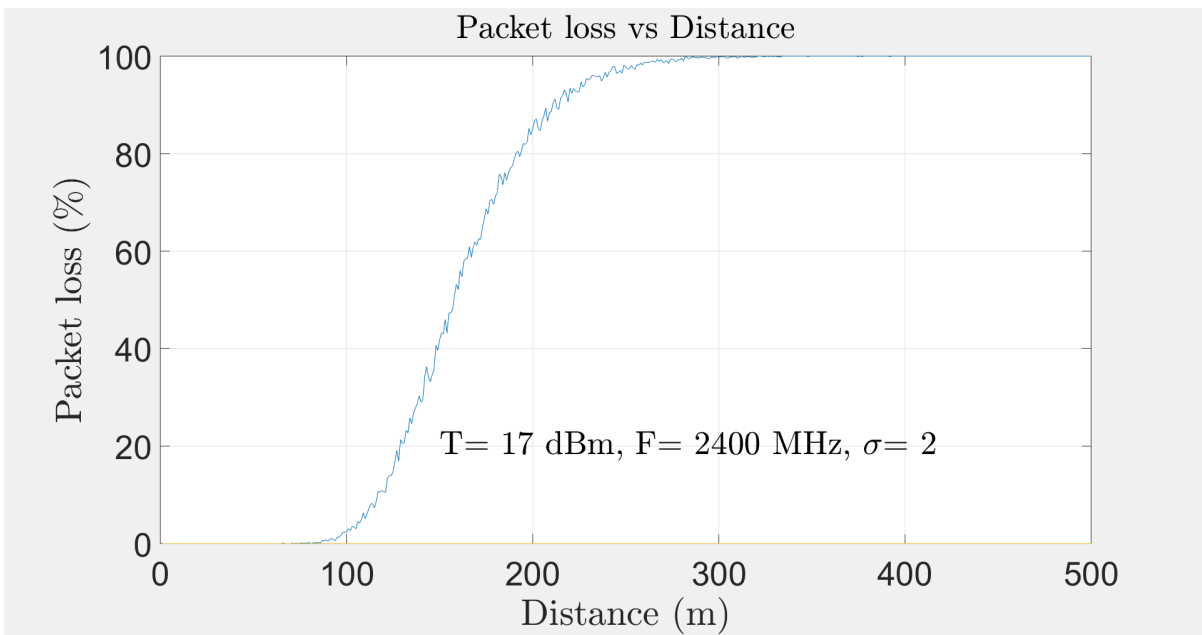


Figure 4.2 Packet loss vs distance. The packet loss increases from $\approx 5\%$ at 100 m to $\approx 90\%$ at 200 m

Table 4.1 Free space path loss - equation parameters

Parameters	value [11]
Transmitter Strength (T)	17.1 dBm
Receiver Threshold (R_{th})	-87.2 dBm
Transmission Frequency (f)	2400 MHz
Standard Deviation (σ)	2
Receiver Antenna Gain (G_{T_x})	0 dBm
Transmitter Antenna Gain (G_{R_x})	0 dBm

Algorithm 7: Signal strength at destination

```
signalStrength ( $T, f, d, G_{T_x}, G_{R_x}$ )  
  output: residualStrength: dBm  
  fspl =  $20 \times \log_{10}(d) + 20 \times \log_{10}(f) + 32.44 - G_{T_x} - G_{R_x}$   
  mean = 0  
  sigma = 2  
  N = normrnd(mean, sigma)  
  residualStrength = T - fspl + N
```



Figure 4.3 Setup to measure processing delay on BeagleBone Black

data on the message processing delay (transmission + reception) on a BeagleBone Black device. The setup in Fig. 4.3 is similar to the configuration proposed in [22] to measure network processing delay.

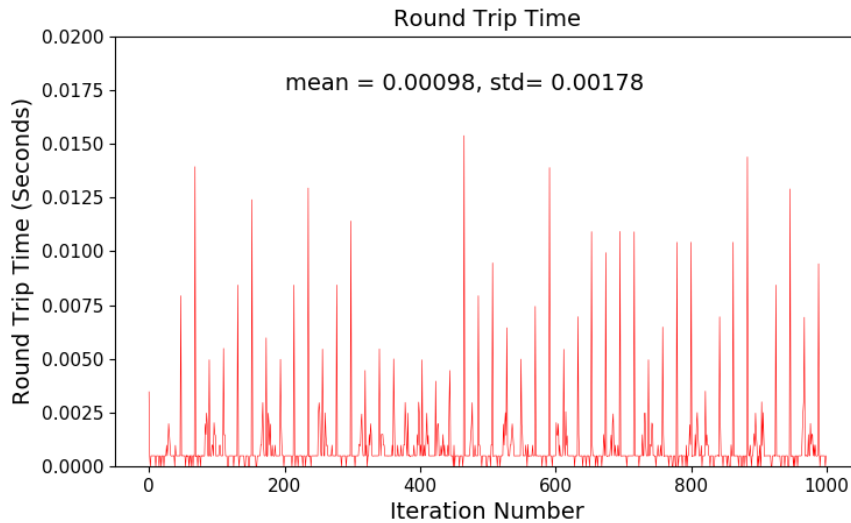


Figure 4.4 Round trip time between two BeagleBone Blacks

A sample plot of round trip time over 1000 iterations is shown in Fig. 4.4. It becomes evident that the total message processing delay is much smaller to the time step. Hence, in our simulation, a message transmitted in the time quantum ‘x’ is also received in the same time quantum ‘x’.

4.2.3 Packet Header

The packet header contains the arguments which define the source, destination, transmission zone and other parameters which govern how a packet should be routed. Following are the fields that are required in a packet header for our routing scheme.

- Packet ID (*pId*): A number that uniquely identifies this packet among those transmitted by this node. The combination of source node identification and packet ID should uniquely identify a packet in the FANET.

- Source Node ID (*srcId*): A unique identifier for the source node. It could be IP address of the node.
- Source Location (*sLoc*): The GPS coordinate of the source node.
- Destination Node ID (*destId*): A unique identifier for the destination node. If a packet is destined to a geographical region, then this shall be set to the macro ANY. If a packet is to be flooded in the network, then this value should be set to the macro FLOOD.
- Destination Location (*dLoc*): GPS coordinate of the destination as known to the source node. In case of flood packet, this would be set to NULL or be ignored.
- Destination Location Timestamp (*The time when the destination node's location was updated*).
- Destination Region Radius (*r*): In case of a geocast packet, the radius along with *dLoc* defines the destination spherical region for the packet.
- Transmitter ID (*trId*): A unique identifier for the latest transmitter of the packet.
- Transmitter Location (*trLoc*): GPS coordinate of the node that had last transmitted the packet.
- Timestamp (*timestamp*): The time when this packet was transmitted by the source.
- Zone Width (*w*): The width of the transmission zone for the packet. Explained in Section 4.2.6.
- Minimum Zone Width (*minWid*): The minimum width of the transmission zone for the packet. Explained in Section 4.2.6.2.
- Hops To Live (*HTL*): A node forwards a message only if $HTL > 0$ and reduces HTL by 1 before forwarding the packet.
- Acknowledgement Required (*ackReq*): This flag denotes whether the source node is expecting an *ack* reply.
- Diverged Zone (*divZone*): This flag denotes whether an intermediate node can update the message headers. For example, if the intermediate node has a location information of the destination that is later than the message's *timestamp*. This is explained in Section 4.2.6.1

4.2.4 Message transmission

The Qrsim simulator has a *Pelican* class which represents a quadrotor drone. An object of pelican class stores drone specific data like location coordinates, velocity, rotor thrust etc. To simulate message transmission we added the following data members to the class.

- *transmitter_strength*
- *receiver_threshold*
- *transmitter_gain*
- *receiver_gain*
- *transmission_frequency*
- *in_msg_queue*
- *seen_msg_set*
- *transmitted_msg_set*

When a transmitting node ‘T’ needs to send a message ‘msg’ to a destination ‘D’, the simulation first checks if the signal strength at the destination will be higher than the receiver’s threshold (8). If yes, then the simulation appends the message to the *in_msg_queue* of the destination node. (9)

A receiver iterates over each message in its message queue and discards any duplicate messages. For new messages, the node does the required processing according to the message header and adds the *msg.pId* to the *seen_msg_set*. This is explained in Algorithm 10

4.2.5 Location Information

For the geographical routing protocols to work successfully, the source node needs to know the geo-location (or an approximation) of the destination location. To provide this information we need a service that maintains and distributes the location table. A classification of location services has been mentioned in Section 2.2.2.1. In our simulation we have implemented an all-for-all location service i.e. every node maintains the location information of every other node in the network.

Algorithm 8: transmissionSuccess(T, D)

Result: transmissionSuccess: TRUE/FALSE
transmissionSuccess = FALSE
d = distance(T.location, D.location)
f = T.transmission_frequency
ts = T.transmitter_strength
gTx = T.transmitter_gain
gRx = D.receiver_gain
receiverThreshold = D.receiver_threshold
strength = signalStrength(ts, f, d, gTx, gRx)
if *strength* > *receiverThreshold* **then**
 | transmissionSuccess = TRUE
end

Algorithm 9: sendMessage(msg, T, D)

Result: status: TRUE/FALSE
status = FALSE
if *msg.HTL* <= 0 **then**
 | EXIT
msg.HTL = *msg.HTL* - 1
if *transmissionSuccess(T, D)* == *TRUE* **then**
 | append(D.in_msg_queue, msg)
 | status = TRUE

Algorithm 10: Read and process the messages in a drone's buffer

readMessages (*drone*)
 | **inputs:** drone object of class Pelican
 | **foreach** *msg* ∈ *drone.in_msg_queue* **do**
 | **if** *msg* ∈ *drone.seen_msg_set* **then**
 | discard(msg)
 | **else**
 | processMessage(msg)
 | add(drone.seen_msg_set, msg)
 | Delete(drone.in_msg_queue, msg)

We have employed a hop limited broadcast algorithm to distribute the location table, i.e. in each time step, every node broadcasts its **location table** to the nodes in its transmission range. The receiving node ‘R’ checks if the location of a node ‘N’ in the message and the timestamp of when the information was generated. If the location in the message is new, then ‘R’ updates its location table. It should be noted that the nodes don’t transmit the message again. This is achieved by setting the ‘HTL’ parameter in the message header to 1. It should be noted that a fast moving node can choose to increase the HTL to cover a larger volume in the same time-step.

The effect is that two nodes ‘p’ and ‘q’ which are ‘x’ transmission ranges apart will have a location information that is $x \cdot DT$ time steps old and therefore not an exact but rather an approximate location of each other. The motivation for this implementation is derived from the concept termed as ‘distance effect’ in [1], which states

The greater the distance separating two nodes, the slower they appear to be moving with respect to each other. Thus, nodes that are far apart, need to update each other’s location less frequently than nodes closer together.

However, in [1] the authors have associated an ‘age’ with the messages which determines how far a message should be transmitted from the source location and the nodes periodically send messages with ‘large’ and ‘small’ age. However, in our case, the information flows gradually like waves in each time-step.

4.2.6 Transmission Zone

As detailed in Section 3.3 the idea behind our routing algorithm is to restrict retransmissions to only those nodes that are inside a region. Among the choices presented in Section 3.3.1 we have chosen a spheroid volume as our transmission zone because such a volume is a generalization of the other three, provides a tapered region and gives equal significance to nodes at the same distance from the ‘source - destination’ line.

As depicted in fig Fig. 4.5 the spheroid has $sLoc(p, q, r)$ and $dLoc(a, b, c)$ as their foci points and ‘pkt.w’ as its minor axis length. An intermediate node ‘N(x, y, z)’ shall forward a message if and only if N is inside the spheroid. We check for this condition using the definition of the ellipsoid ‘The sum of the distances to the two foci points is constant for every point on the curve.’

$$distance(N, sLoc) + distance(N, dLoc) == K \tag{4.3}$$

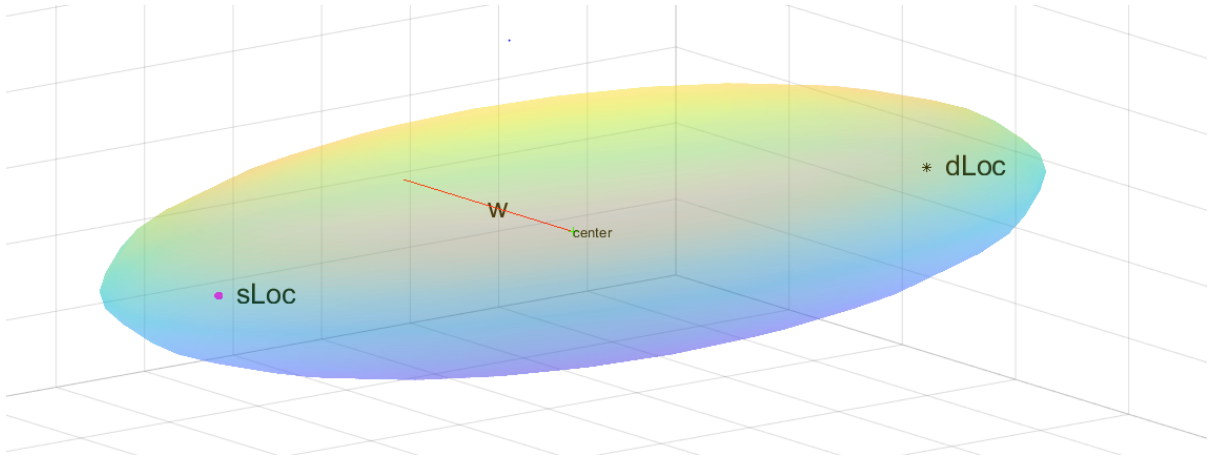


Figure 4.5 Single transmission zone with source at sLoc and destination at dLoc. Intermediate nodes use *pkt.sLoc* and *pkt.dLoc* to determine their position relative to the transmission zone

where K is the length of the major-axis. Hence, if the left hand side of equation 4.3 is $\leq K$ then, the point N is inside the spheroid.

4.2.6.1 Diverged transmission zone:

We observe that there are at least two scenarios where the above described transmission zone might not contain the intermediate nodes for successful transmission.

1. Outdated destination location: As discussed in Section 4.2.5, if two UAVs are multi-hops away then they might not have an exact location of each other rather an approximate location which is a few time-steps old. However, the intermediate nodes might have a more precise location.
2. Routing void: is encountered when there are insufficient nodes close to the 'source - destination' line for a successful packet delivery however there are nodes outside the periphery of the transmission zone which can deliver the packets.

In these two cases the source node can increase the zone width which has a disadvantage of increasing the number of transmissions. Another approach is to let the intermediate nodes update the packet headers with newer information. For example, in Fig. 4.6, the source node 'S' needs to send a packet to destination node 'D'. 'S' creates a packet with headers that specify the zone 'P-1' around 'sLoc - dLoc'. The intermediate nodes 'T0' and 'T1' know the current location of D and update the packet with headers that specify zones

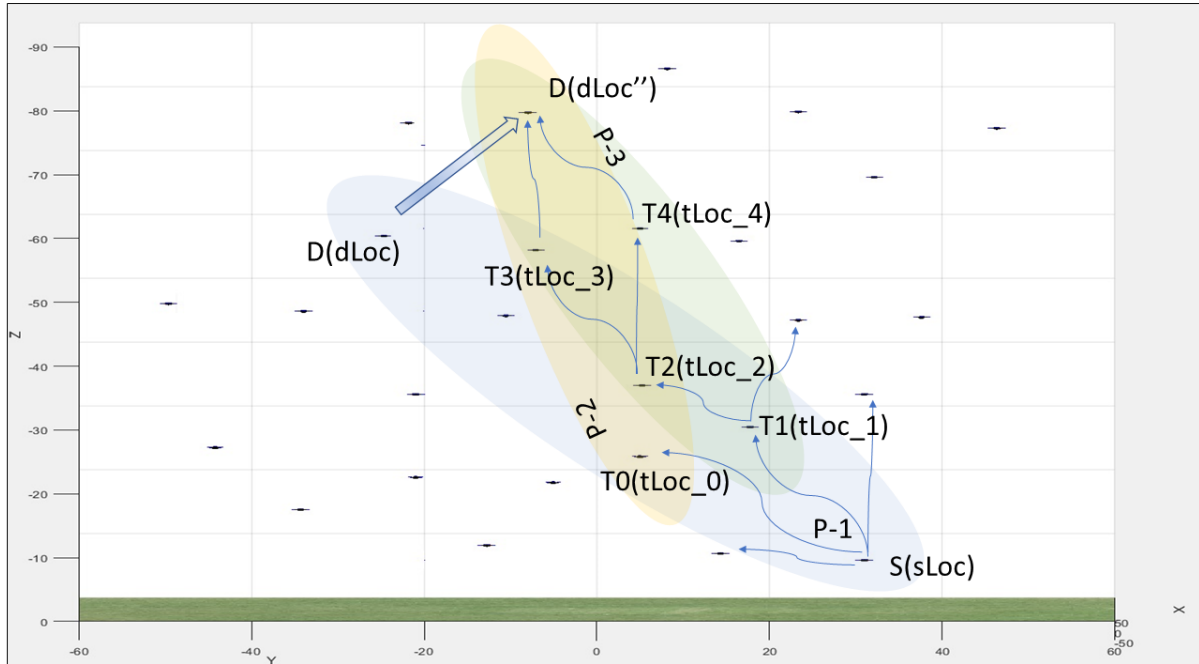


Figure 4.6 Multiple diverged transmission zones. Intermediate nodes use $pkt.tLoc$ and $pkt.dLoc$ to determine their position relative to the transmission zone

'P-2' and 'P-3'. Afterward 'T3' and 'T4' receive the packet and being inside the updated zones transmit the packet which is received by the destination D at 'dLoc''. These approach makes the zones to diverge from the original one and hence we will refer to this scheme as **diverge zone routing**. This approach also helps navigate the routing void with a thinner zone width as compared to the single zone scheme.

This approach is in contrast with the scheme where every intermediate node considers the same transmission zone with $pkt.sLoc$ and $pkt.dLoc$ as the focii points of the spheroid and we shall compare the performance of multiple diverged transmission zones with a single transmission zone scheme in 5.

4.2.6.2 Zone width:

A critical parameter of our algorithm is the zone width, which determines how much should the transmissions spread out. A narrow width will lead to less transmissions but the reliability of packet delivery shall reduce. On the other hand higher zone width shall lead to a higher delivery rate but also increase the number of retransmissions. Therefore it

is important to pick a width that balances out the two factors. In our implementation the zone width is a function of source — destination distance. i.e.

$$\text{zone width} = \frac{\text{distance}(\text{source}, \text{destination}) * \text{widthPercent}}{100} \quad (4.4)$$

Where *widthPercent* is a parameter which controls the width of the zone. In Chapter 5 we study the performance metrics as a function of *widthPercent*. It should be noted that this approach leads to thinning of zone as the packet approaches the destination which would lead to increase in packet drops near the destination. Therefore in diverged transmission zones scheme there is a minimum width which is a '*minWidth*' percent of the 'source — destination' distance.

4.2.7 Back-off time

We discussed in Section 3.3.1, that the number of transmissions can be further decreased by making the node *back-off* for a certain time and then decide whether to transmit or not. In this section we shall describe the method of calculating the back-off duration.

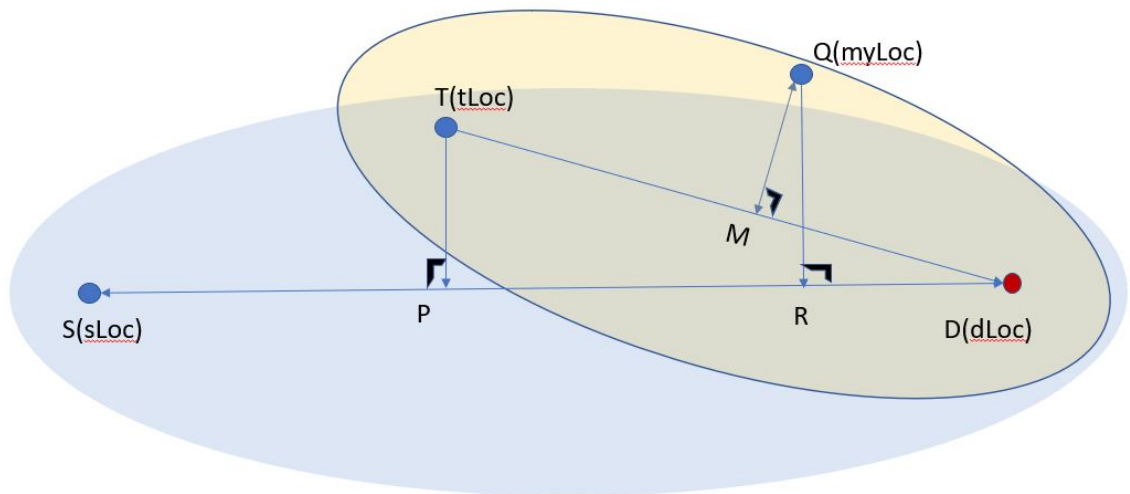


Figure 4.7 Back-off time calculation

The main goal of back-off calculation is that the node closer to the destination and the source - destination line should transmit first. Therefore, the *backOff* calculation is divided

into two parts.

- tB1 → back-off time proportional to the distance from destination. This is bounded above by $tUb1$.
- tB2 → back-off time proportional to the distance from the source - destination line. This is bounded above by $tUb2$.

For example, as depicted in Fig. 4.7, intermediate node T needs to determine the back-off time. Node T uses the displacement across the SD line (i.e. PD) to calculate ‘tB1’ and then uses the distance from SD line (i.e. TP) to calculate ‘tB2’. In case of diverge transmission zone the intermediate node uses the distance from ‘transmitter - destination’ line to calculate ‘tB2’. This is depicted in Algorithm 11. In our implementation we have $tUb1 = 0.002s$ and $tUb2 = 0.0005s$.

Algorithm 11: Back-off time calculation at intermediate nodes

```

BackOffTime ( $sLoc, dLoc, tLoc, myLoc, tUb1, tUb2, zoneType$ )
  output: backOff seconds
  TD' = projection of  $\overrightarrow{TD}$  on  $\overrightarrow{SD}$ 
  tB1 =  $\frac{tUb1 * TD'}{SD}$ 
  if  $zoneType == SINGLE$  then
    | X = sLoc
  else
    | X = tLoc
  XP = distance of X from  $\overrightarrow{XD}$ 
  tB2 =  $\frac{tUb2 * XP}{XD}$ 
  backOff = tB1 + tB2

```

4.2.8 Inter-UAV forces

The UAVs are initially arranged in a grid and then this grid is maintained during the mission. To maintain the grid formation whenever two UAVs, go out of a certain range, they exert a virtual pull towards each other, and when they come too close they exert a virtual push

towards each other. Moreover, once two UAVs are beyond a certain limit, their influence should gradually wear off. We have used Equation 4.5 and 4.6 to calculate the piecewise differentiable function presented in Algorithm 12. A plot of the force magnitude as a function of distance is shown in Fig. 4.8.

$$f1(X) = \text{hyperbolicSine}(X) \quad (4.5)$$

$$f2(X) = \frac{X}{1 + |X|} \quad (4.6)$$

Algorithm 12: Inter UAV pull and push force magnitude

```

force (dst, fMax)
  inputs: dst: Distance between two drones;
           fMax: maximum pull or push force
  y1 = sinh(b - dmax)
  if dst < a then
    | force = f2(dst-a) * fmax - y1
  else if dst < dmin then
    | force = f1(dst - dmin)
  else if dst < dmax then
    | force = 0
  else if dst < b then
    | force = f1(dst - dmax)
  else if dst < c then
    | force = f2(dst - b) * fMax + y1
  else if dst < d then
    | force = f2(d - dst) * fMax + y1
  else if dst < e then
    | force = f1(e - dst)
  else
    | force = 0

```

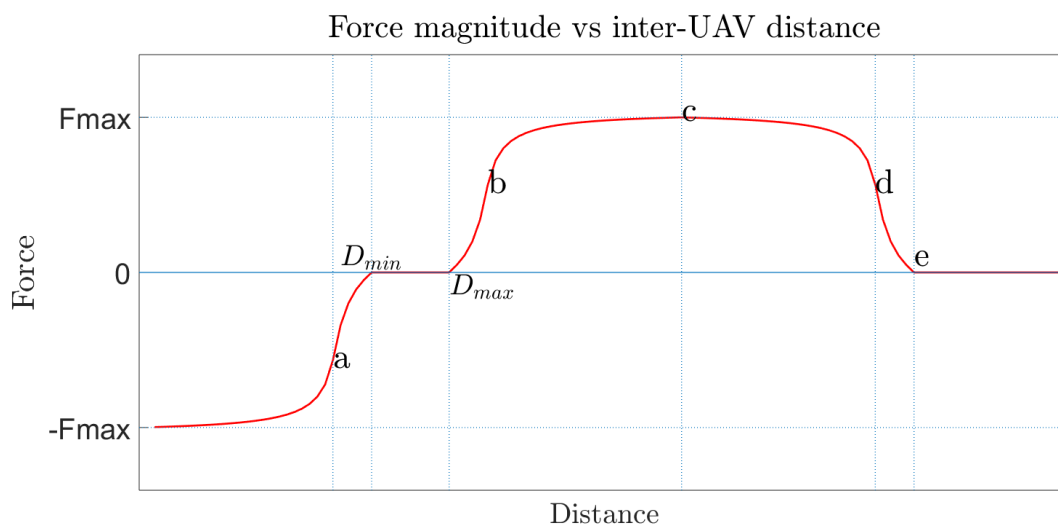


Figure 4.8 Plot of force magnitude calculated using multipart function in Algorithm 12 as a function of inter-UAV distance

CHAPTER

5

SIMULATION RESULTS

We have used Qrsim quadrotor simulator [20] - briefly explained in Section 4.1 - for evaluating our protocol. The volume of the simulated flying zone is 120 units \times 120 units \times 60 units, out of which we have used 80 units \times 80 units \times 40 units where drones can be placed. We also have a scaling variable which scales the simulation units i.e. a scaling factor of 5 will make 1 unit in simulation to 5 meters. The drones have a reliable transmission range of $\approx 90m$ (Fig. 4.2) and the wi-fi antenna is omnidirectional. The simulation state changes at a time quantum of 0.2 seconds.

Initially, the drones start in a mesh formation arranged in a 2D grid as shown in Fig. 3.1 and gradually they wrap around the plume - which, in our case, for simplicity, has been assumed to be spherical Fig. 3.3. Besides these two formations, we have also evaluated our results for a random distribution of drones in the simulation space Fig. 5.1.

The messages sent among the drones are assumed to be small and we have not considered transmission errors and congestion separately, rather they are assumed to be represented by the free space path loss model explained in Section 4.2.1 and the parameters which are in Table 4.1.

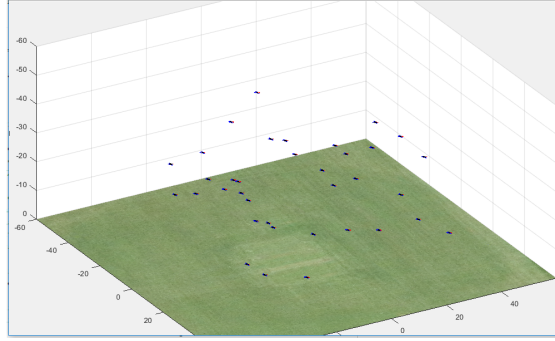


Figure 5.1 Random distribution of UAVs in the simulation space

In the following sections we compare the performance of our routing scheme to flooding. We shall be studying the following performance metrics ‘Packet delivery ratio’, ‘Average number of hops’, ‘Average number of transmissions’ [21]:

We compare these metrics for three arrangements of the UAVs namely, *mesh* - Fig. 3.1, *random* - Fig. 5.1, *spherical* - Fig. 3.3. The three formations have peculiar properties, for example the spherical formation has the plume acting as a *routing void*, whereas the mesh formation represents a 2D formation and thus we can study the performance of our 3D routing scheme in a 2D setting. Finally, random placement of the drones lets us study the performance in a generic situation. We have presented the aggregate results for 15 iterations where in each iteration, 250 packets are transferred between the 5 furthest node pairs.

5.1 Packet delivery ratio

Packet delivery ratio (PDR): is defined as the ratio of ‘count of all packets successfully delivered (P_r)’ to ‘count of all the packets transmitted by the sender (P)’. PDR characterizes the robustness and reliability of a routing scheme. The bigger is PDR, the better is the performance of the protocol. The packet delivery ratio of flooding algorithm is considered as the upper bound for any routing mechanism since if the destination node is reachable then the probability of packet delivery is high due to the redundant transmissions. This can be observed in Fig. 5.2 which shows a plot of PDR vs Hops to Live (HTL) for our implementation of flooding algorithm. It should be noted that for small HTL values (e.g. $HTL = 1$) the PDR is low because the destination is more than 2 hops apart from the source and the intermediate nodes don’t forward the packets. However as HTL approaches the diameter of the network

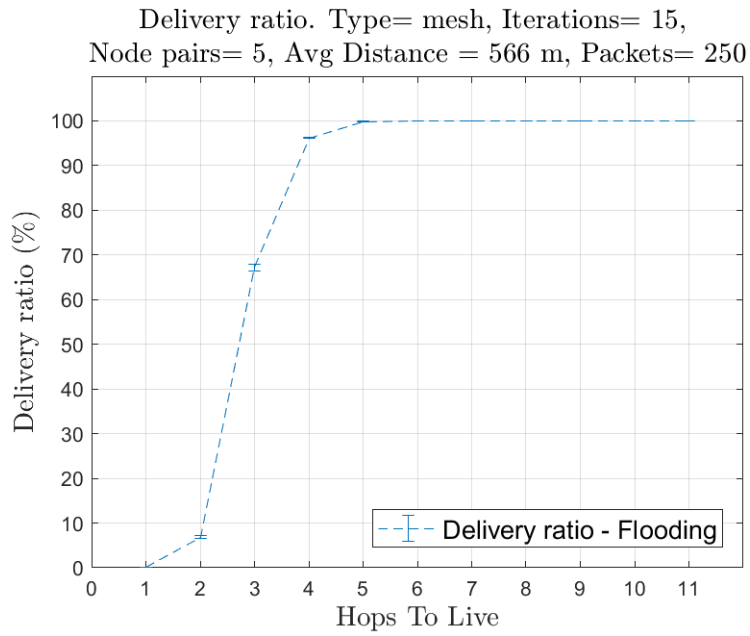


Figure 5.2 Delivery ratio in mesh formation for flooding algorithm

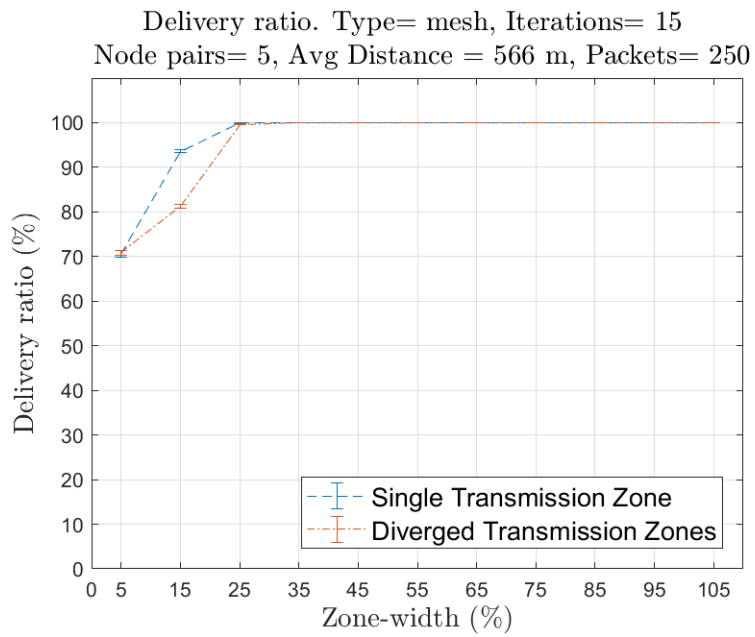


Figure 5.3 Delivery ratio in mesh formation for petal routing

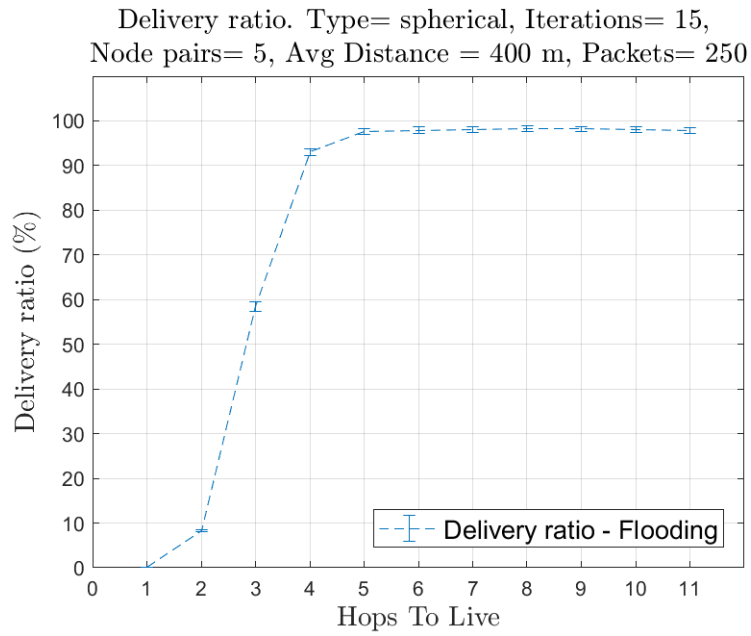


Figure 5.4 Delivery ratio in spherical node distribution for flooding algorithm

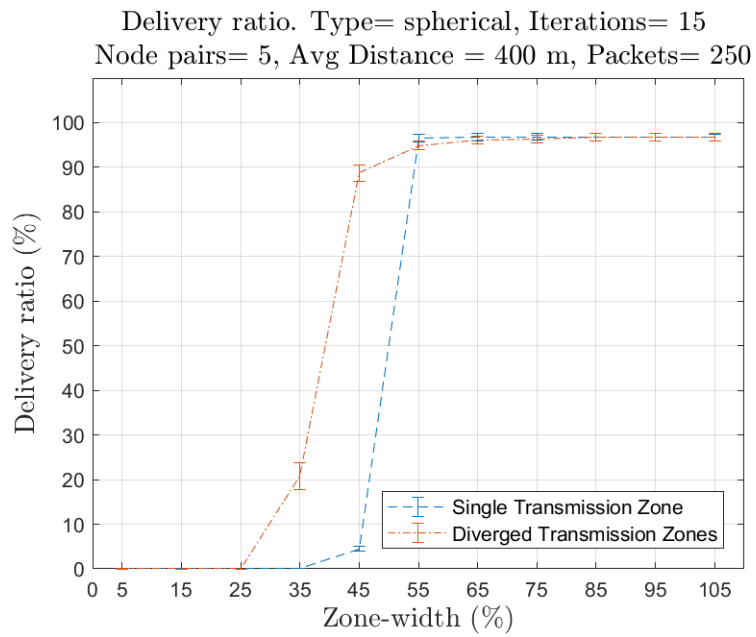


Figure 5.5 Delivery ratio in spherical node distribution for petal routing

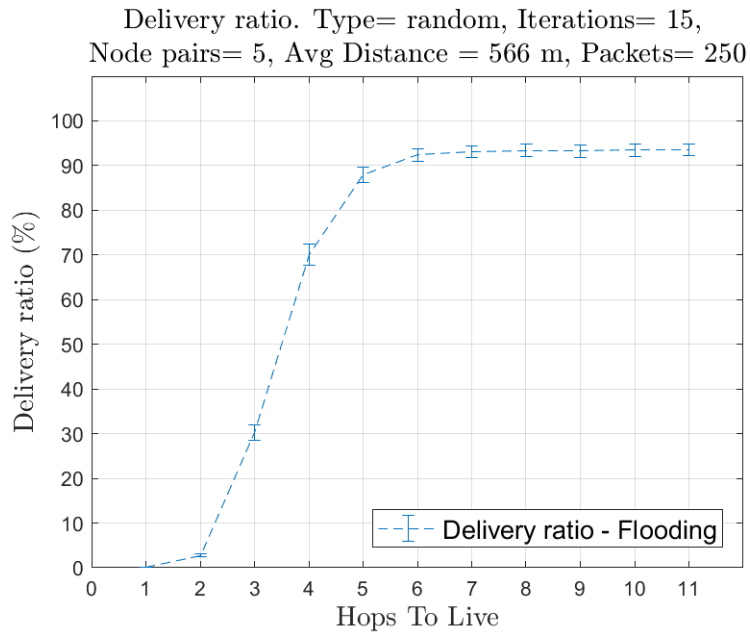


Figure 5.6 Delivery ratio in random node distribution for flooding algorithm

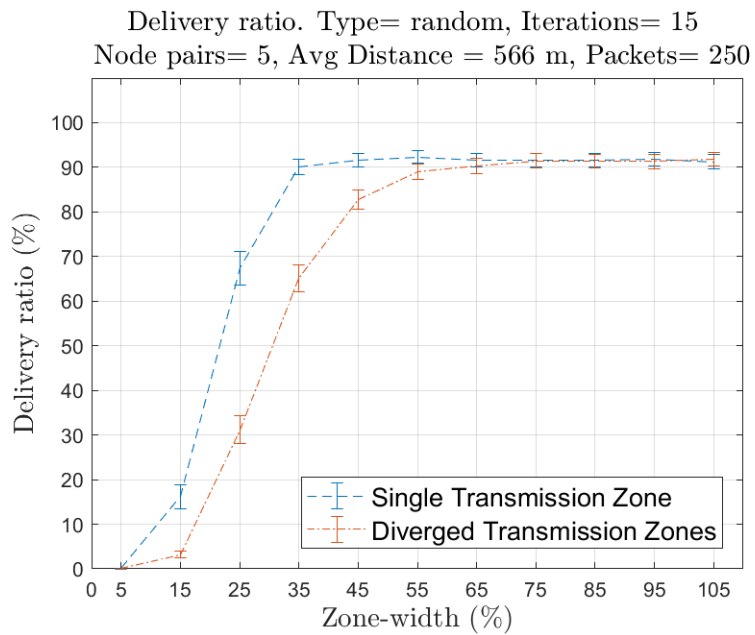


Figure 5.7 Delivery ratio in random node distribution for petal routing

(HTL > 4) PDR approaches 100% delivery ratio.

Fig. 5.3 shows the PDR of petal routing protocol in a mesh formation. We observe that even for petal width of 5% and 15%, the delivery ratio is between 70% to 90% which is due to the alignment of intermediate nodes in a straight line from source to destination. As expected the PDR approaches 100% for higher petal width.

Fig. 5.4 shows the PDR of flooding in a spherical formation. The distance between the two furthest nodes is ≈ 400 m (diameter of the plume) which again causes the PDR to be low for HTL < 4 and gradually increase to 100% for HTL > 4. Fig. 5.5 depicts the PDR for the single and diverged transmission zones in case of petal routing. Unlike mesh formation here we notice that PDR is low for petal width of 15% and 25% which is expected because of the routing void created by the plume. It should be noted that the PDR for diverged transmission zone is higher as compared to single transmission zone for petal width of 35% and 45%. This observation supports our hypothesis as presented in Section 4.2.6.1.

Finally, Fig. 5.6 shows the PDR of flooding algorithm for a random distribution of UAVs in the flying space. Unlike mesh and spherical formation the PDR doesn't reach 100% in random distribution rather stays near 90% for both flooding and petal routing. We can also observe the effect of a thinning petal near the destination for diverged transmission zone in Fig. 5.7 where a single transmission zone achieves a higher PDR as compared to diverged transmission zone for petal width around 35%.

5.2 Average number of hops

Average number of hops (H): is the average number of hops made by a successfully delivered packet. In other words 'H' is the number of nodes that transmitted a packet that was successfully delivered to the destination. The number of hops a packet makes before reaching the destination is a representation of the resources consumed in delivering the packet. With antennas having fixed transmitting strength, reducing the number of hops reduces the energy expenditure. Therefore since we have assumed antennas have fixed transmitting strength, a smaller 'H' is desired.

Fig. 5.8 shows the number of hops made by packets before reaching destination in a mesh formation. Since the nodes are arranged in a straight line even with an inter-drone distance of ≈ 500 meters the packets reach the destination in about 4 hops. Fig. 5.9 plots a similar graph for the spherical distribution. We can see that the number of hops made by

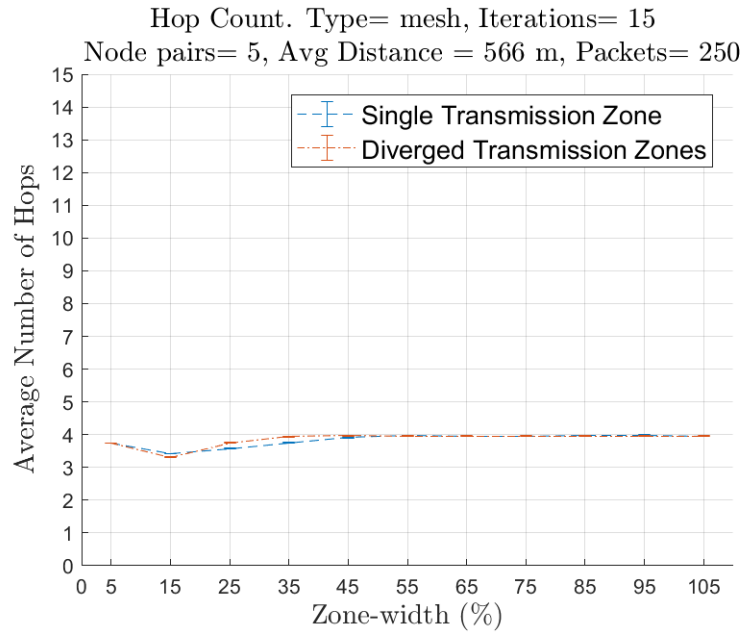


Figure 5.8 Average number of hops in mesh formation for petal routing

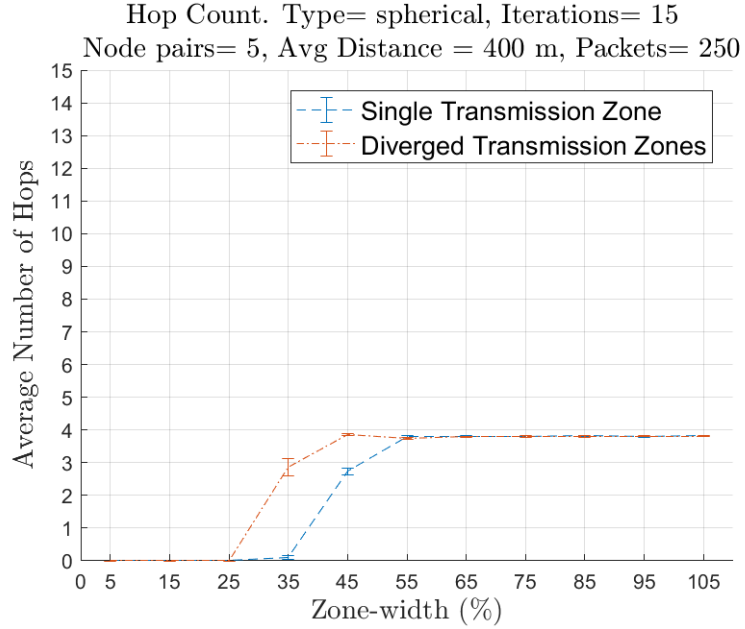


Figure 5.9 Average number of hops in spherical node distribution for petal routing

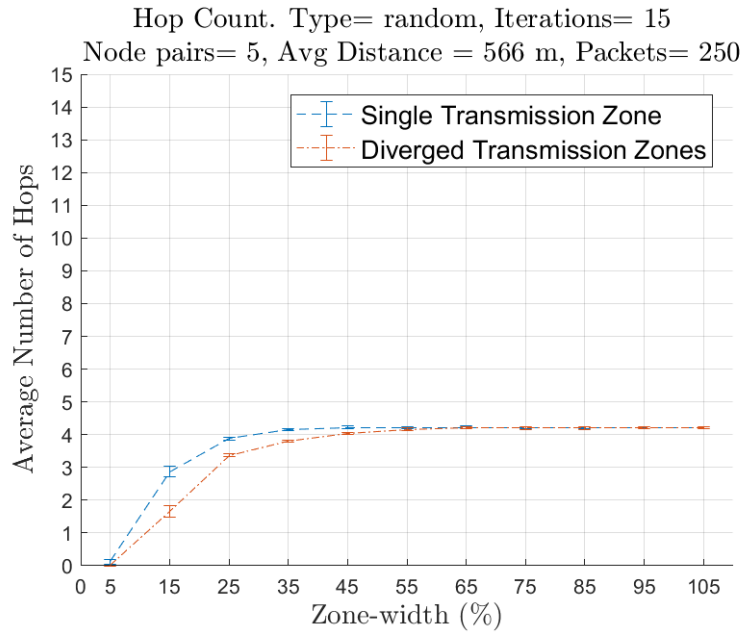


Figure 5.10 Average number of hops in random node distribution for petal routing

packets in diverged transmission zone is higher for petal width of 35% and 45% because the intermediate nodes can route the packets over the spherical void even with thin petal whereas there are not sufficient number of nodes in the other case.

Finally Fig. 5.10 shows the average number of hops made before successful delivery in a random distribution of nodes. Again, we can see that due to thinning of the petal as the packets reach close to the destination the average number of hops is less in random distribution of nodes.

5.3 Average number of transmissions

Average number of transmissions (NT): is defined as the total number of transmissions divided by the total number of packets successfully delivered. In an ideal routing protocol 'NT' would be equal to 'number of hops' \times 'number of packets delivered'. In other words number of transmissions is the combined transmissions from all nodes for each successful packet delivery. 'NT' influences the energy spent in routing a packet also lower number of transmissions leads to lesser collisions in a wireless medium. Hence, a lower value of 'NT' is desirable.

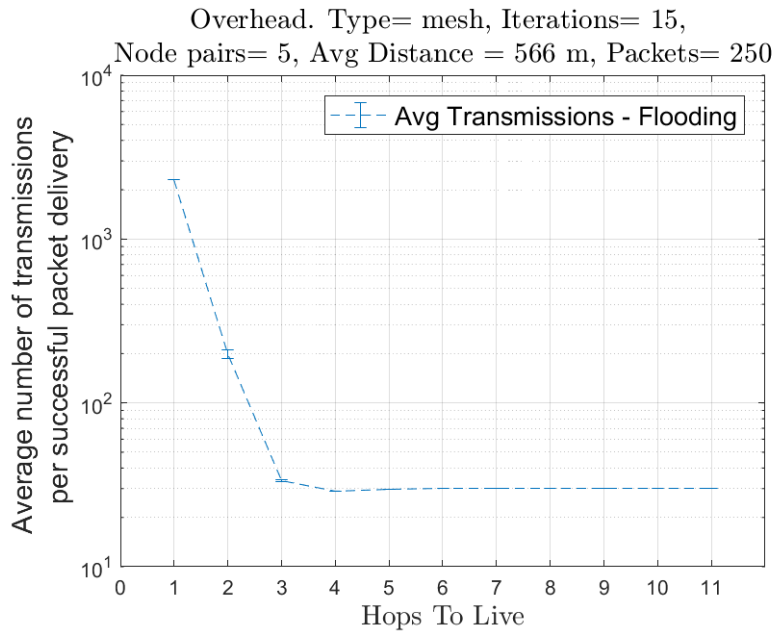


Figure 5.11 Average number of transmissions per successful packet delivery in mesh formation for flooding algorithm

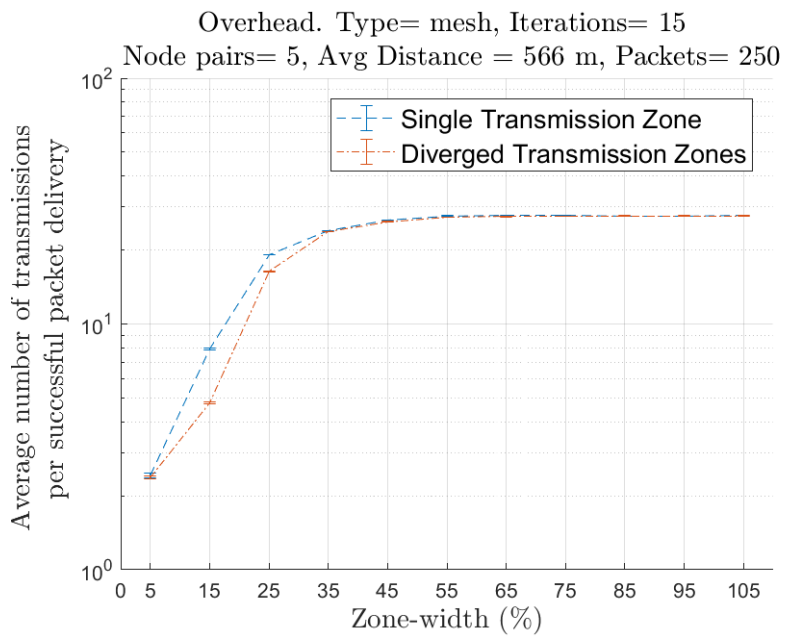


Figure 5.12 Average number of transmissions per successful packet delivery in mesh formation for petal routing

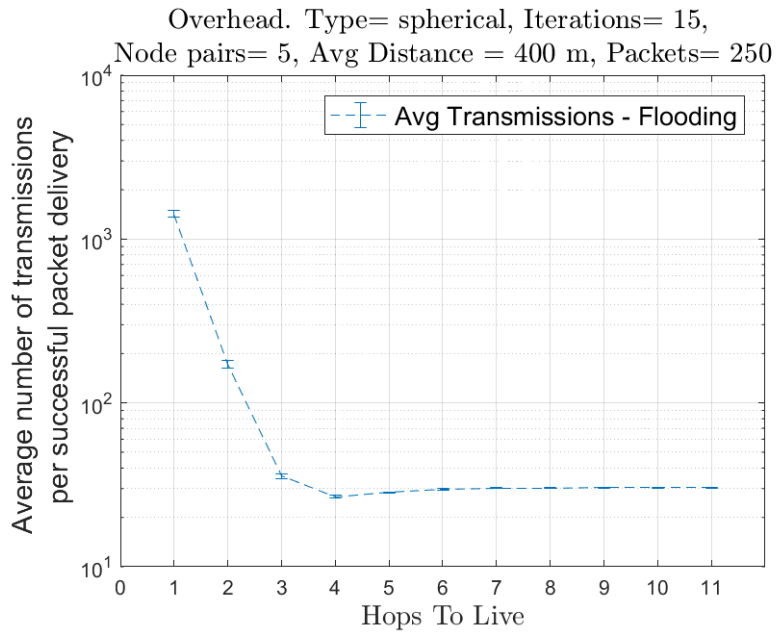


Figure 5.13 Average number of transmissions per successful packet delivery in spherical node distribution for flooding algorithm

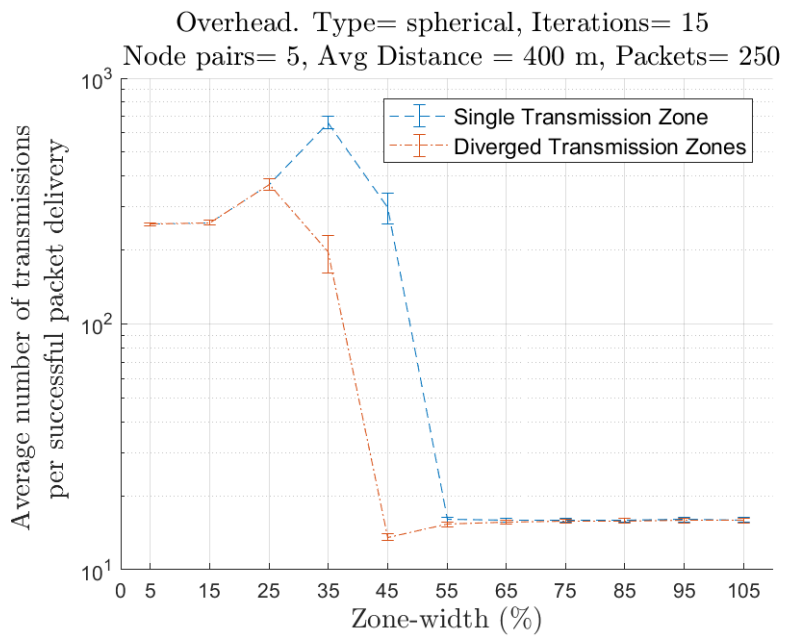


Figure 5.14 Average number of transmissions per successful packet delivery in spherical node distribution for petal routing

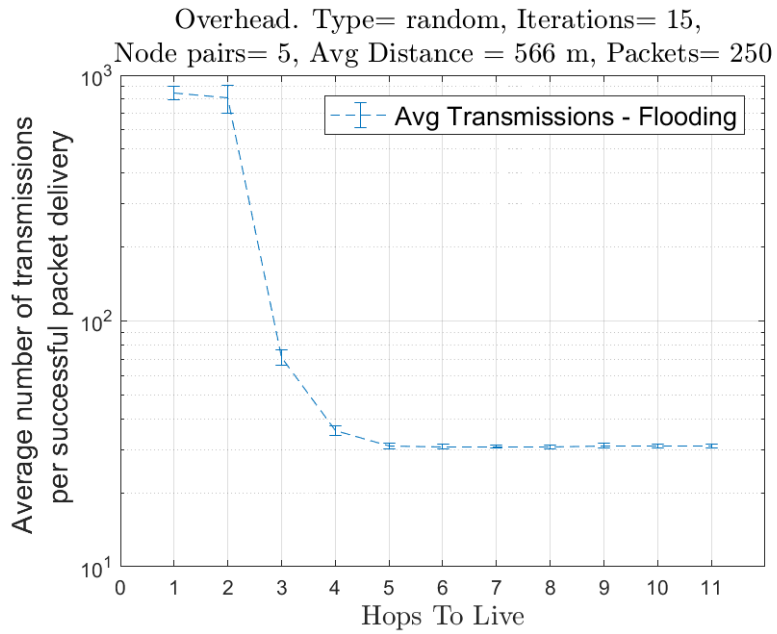


Figure 5.15 Average number of transmissions per successful packet delivery in random node distribution for flooding algorithm

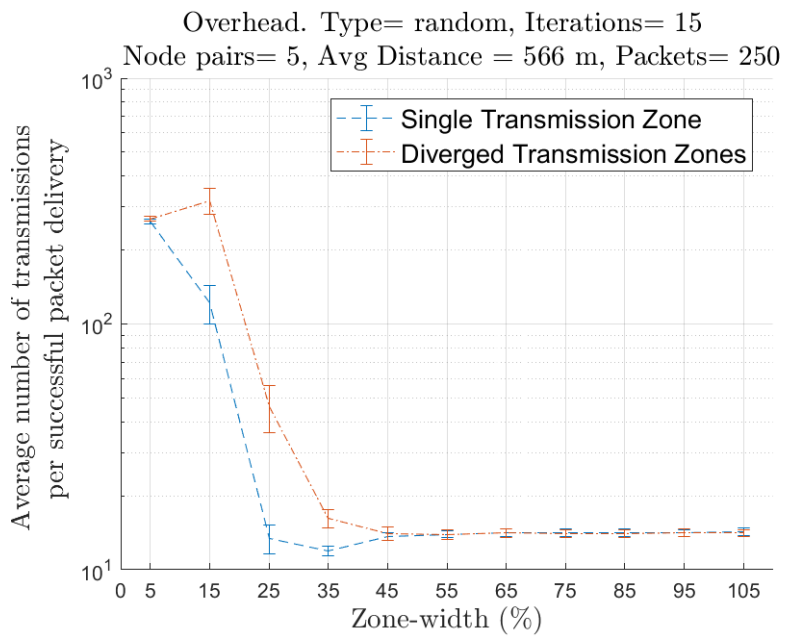


Figure 5.16 Average number of transmissions per successful packet delivery in random node distribution for petal routing

For any packet the upper bound of ‘NT’ is the total number of nodes in the network (because each node transmits the packet at most once). This can be noticed for flooding algorithm in Fig. 5.11 where NT is approximately equal to (≈ 30) the total number of nodes (= 36). whereas in our routing algorithm (Fig. 5.12) the number of transmissions drops significantly (about half of flooding) due to the restricted flooding and ‘back-off and transmit’ strategy. It should be noted that if the delivery rate is low the average number of transmissions gets high hence we see the large values for small HTL or thin petal widths.

5.4 Effect of network density

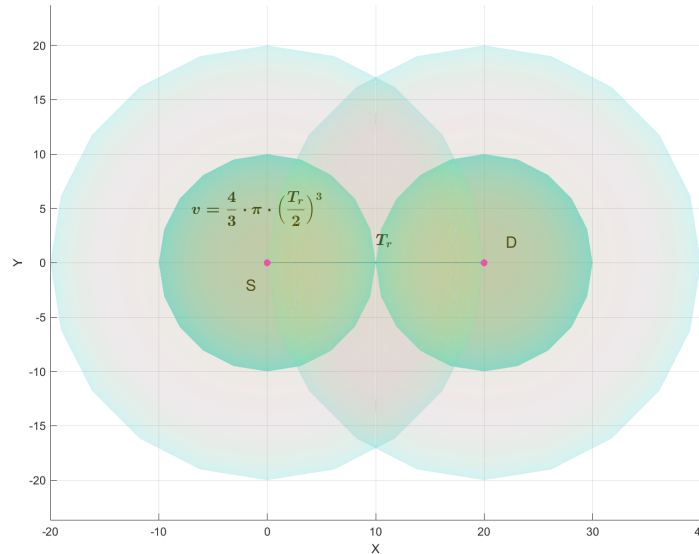


Figure 5.17 Compact node alignment for ideal network density

Network density: To achieve a reliable direct radio communication between a pair of nodes, the maximum separation between the two nodes should be less than the transmission radius of the transmitting antenna. Therefore — assuming omnidirectional antennas with transmission radius T_r — an optimal number of nodes will result in a crystal like packing of space — the sphere being a node’s zone of influence with the node at its center

and $\frac{T_r}{2}$ as its radius. This is depicted in Fig. 5.17. A lower number of nodes will result in less reliable communication whereas higher number will result in redundant transmissions and collisions. Hence, we define network density as the ratio of the volume of flying space to the combined volume of the node's sphere of influence. i.e.

$$\text{Network Density} = \frac{L \times W \times H}{\text{nodeCount} \times \frac{4}{3} \times \pi \times \left(\frac{T_r}{2}\right)^3} \quad (5.1)$$

and we classify the network as sparse, ideal and dense as per the following criteria.

$$\begin{aligned} \text{Sparse:} & < 0.8 \\ \text{Ideal:} & [0.8, 1.2] \\ \text{Dense:} & > 1.2 \end{aligned} \quad (5.2)$$

For example, with scaling factor = 4, Number of Drones = 43, and flying space volume = $80 \times 80 \times 40 \text{unit}^3$ and $T_r = 90m$,

$$\begin{aligned} \text{we have,} & \quad V = 80 \times 80 \times 40 \times 64 = 16384000 \text{unit}^3 \\ \text{and} & \quad v = 43 \times \frac{4 \times 3.14 \times \left(\frac{90}{2}\right)^3}{3} \approx 16404930 \text{unit}^3 \\ \Rightarrow \text{Network Density} & = \frac{V}{v} \approx 1 \end{aligned}$$

With 36 drones, the network density is about 1.2, hence the metrics we presented in the previous sections were for ideal network density. In this section we shall study the effect of network density on the performance our routing scheme and flooding.

As we observed in Section 5.1, the PDR for flooding algorithm depends on the HTL and is $> 90\%$ for $HTL > 4$ whereas PDR depends on petal width in case of petal routing and $> 90\%$ for $width > 35\%$. Therefore, to study the effects of network density we have picked $HTL = 10$ for flooding and petal width = 35% for petal routing. As in previous sections we present the aggregate results for 30 iterations where in each iteration, 300 packets are transferred between the furthest node pair.

Fig. 5.18 shows the variation in delivery ratio vs network density. We can observe that the PDR for flooding is close to that of single transmission zone scheme of petal routing even for sparse network density. This demonstrates the robustness of petal routing i.e. even

in sparse networks the reliability of petal routing is as good for flooding for appropriate petal width.

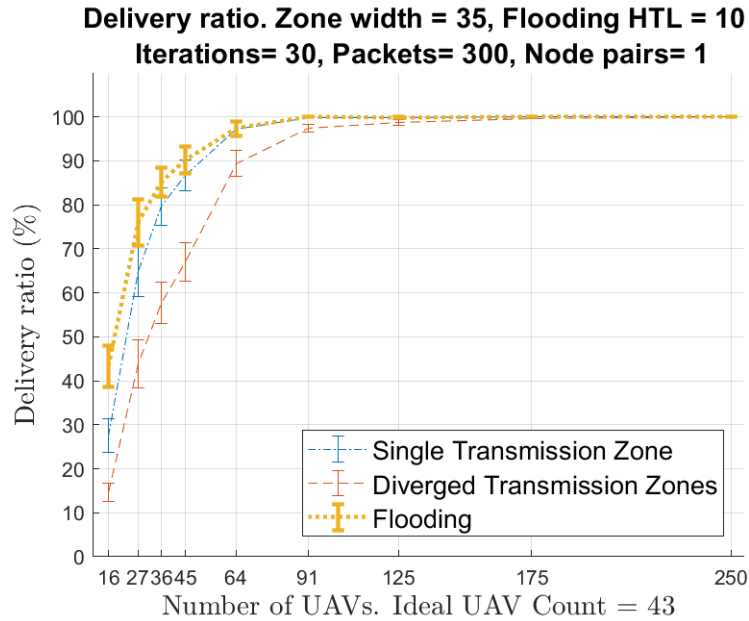


Figure 5.18 Delivery rate vs network density

Fig. 5.19 depicts the change in average number of hops as compared to the network density. We can observe that the number of hops doesn't vary much with increase in network density. This is also expected since in both the algorithms the packets propagate in a breadth first manner and hence find the shortest path to the destination.

In Fig. 5.20 we observe that the average number of transmissions increases with increase in network density. However, the increase is more steep for flooding algorithm as compared to petal routing. Specifically, for flooding algorithm, the average number of transmissions is roughly equal to the number of nodes in the network. This is expected since all the nodes which receive a packet transmit at least once in case of flooding; however, in our routing scheme we restrict the retransmissions to a zone and by back-off mechanism as well.

**Average Number of hops. Zone width = 35, Flooding HTL = 10
Iterations= 30, Packets= 300, Node pairs= 1**

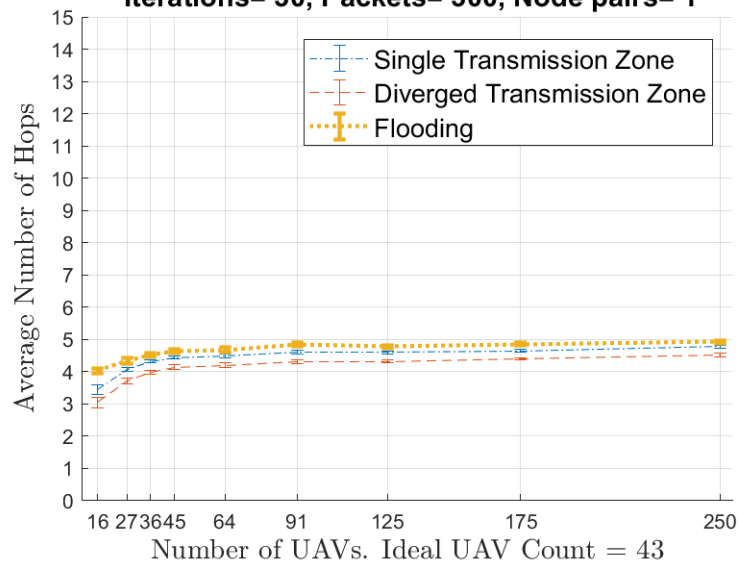


Figure 5.19 Average number of hops vs network density

**Transmissions Count. Zone width = 35, Flooding HTL = 10
Iterations= 30, Packets= 300, Node pairs= 1**

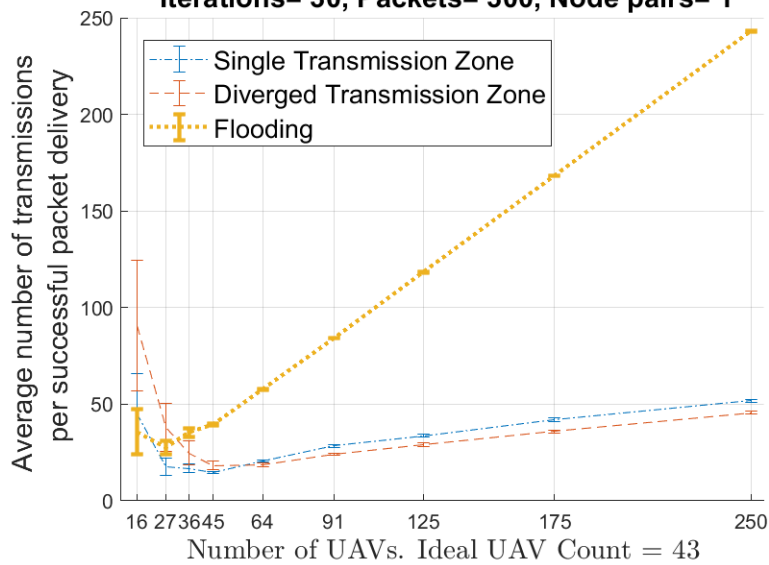


Figure 5.20 Average number of transmissions per successful packet delivery vs network density

5.5 Effect of location inaccuracy

In this section we shall study the effect of node mobility on the previously studied performance metrics. We have studied a random distribution of nodes in the flying volume where each node has a random velocity. As we discussed in Section 4.2.5 the difference in the actual location of the destination node and the location in the location table of the source node is a function of speed of the destination node as well as the distance between the source and the destination node. The following results are with 36 drones randomly distributed in a $(80 \times 80 \times 40)$ volume with a scaling factor of 5.

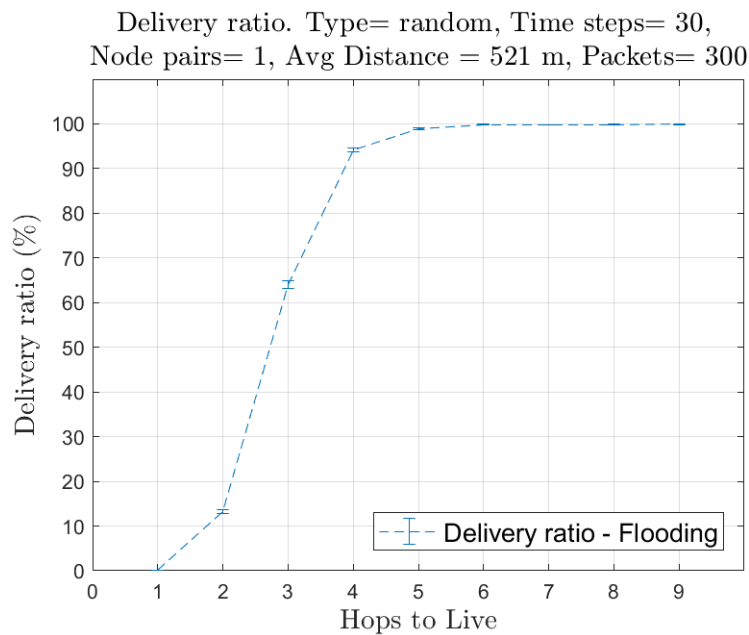


Figure 5.21 Delivery Rate for flooding algorithm with inaccurate destination location in a random distribution of nodes

Fig. 5.21 depicts a plot of delivery rate for flooding algorithm, whereas Fig. 5.22 depicts a plot for delivery rate for petal routing. We can notice that for large HTL (> 4) and petal width ($> 35\%$) the delivery rate for both the algorithms are comparable. However an interesting point to note is that the delivery rate with diverged transmission zones scheme is better for smaller petal widths. This validates our motivation for diverged transmission zone as presented in Section 4.2.6.1.

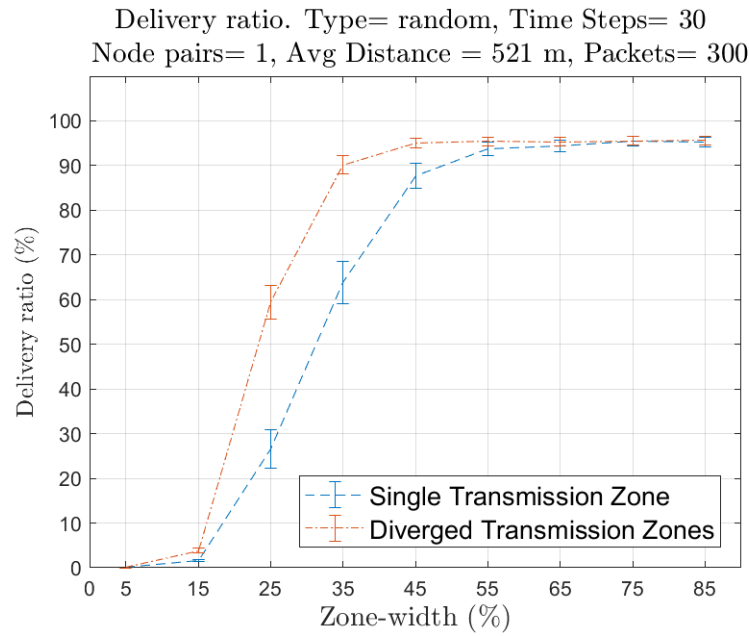


Figure 5.22 Delivery Rate for petal routing with inaccurate destination location in a random distribution of nodes

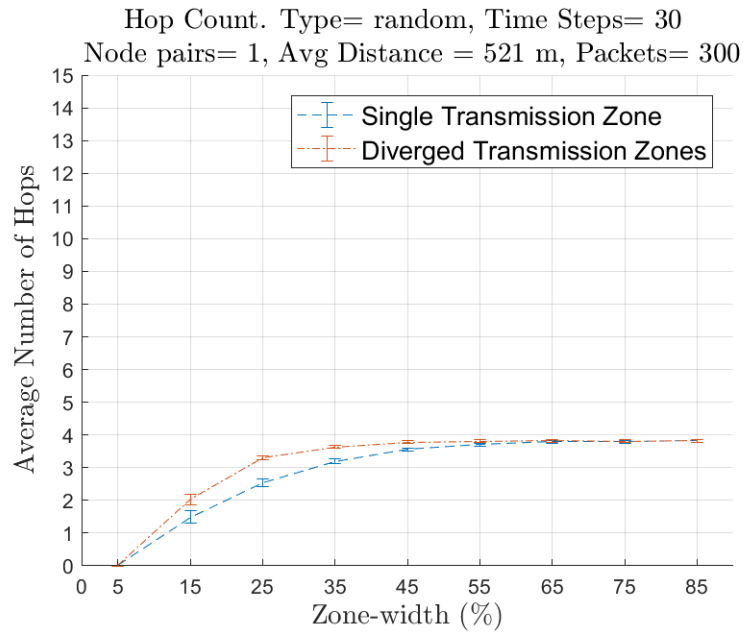


Figure 5.23 Average number of hops for petal routing with inaccurate destination location

Fig. 5.23 depicts the average number of hops between the two furthest nodes. As we can notice from Fig. 5.23 the average number of hops to a reliable delivery is about 4 hops which is substantiated by Fig. 5.21.

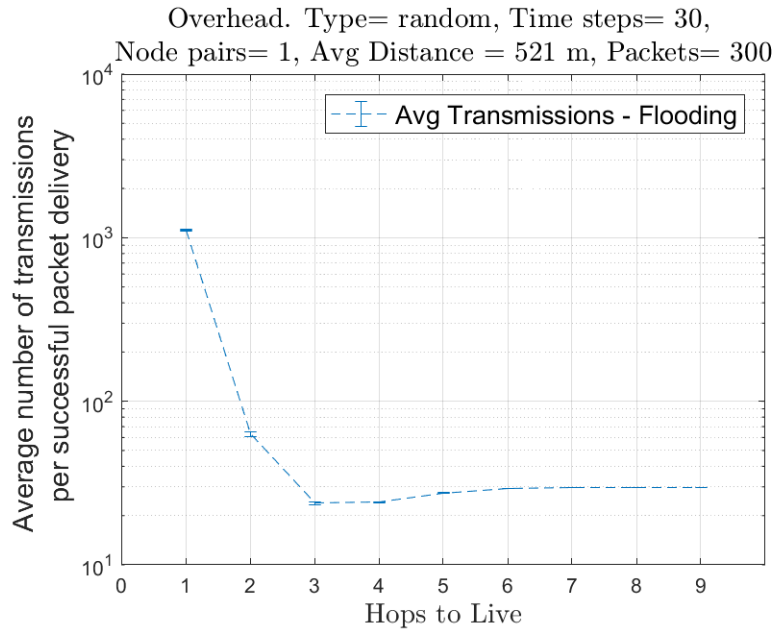


Figure 5.24 Average number of transmissions per successful packet delivery for flooding algorithm with inaccurate destination location

In Fig. 5.24 we present the average number of tries for each successful delivery of a packet for flooding algorithm and in Fig. 5.25, we depict the same metric for petal routing algorithm. As with previous observations the average is equal to the number of nodes for flooding (≈ 30) whereas it is significantly less (≈ 10) for petal routing due to limited flooding and back-off mechanism.

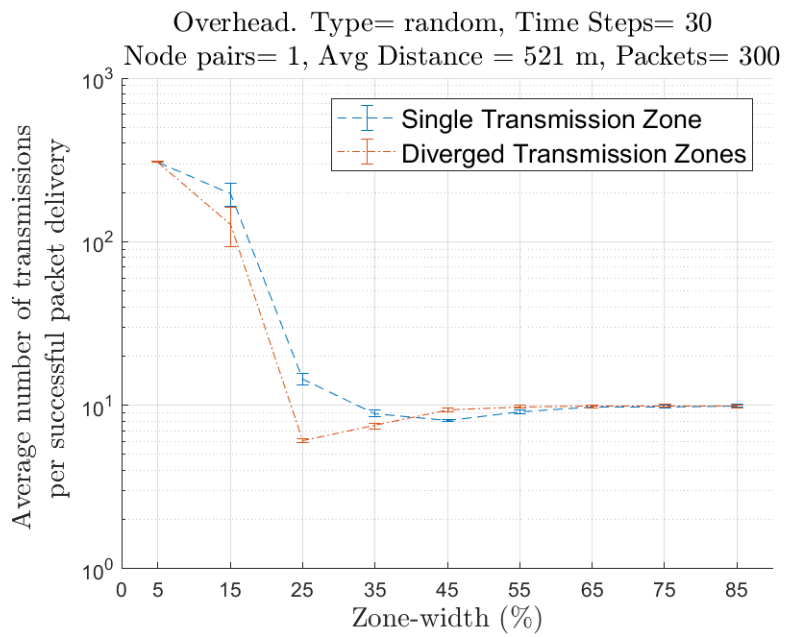


Figure 5.25 Average number of transmissions per successful packet delivery for petal routing with inaccurate destination location

CHAPTER

6

CONCLUSIONS AND FUTURE WORK

In this thesis we presented a geographic routing algorithm that doesn't maintain either end-to-end paths or neighbor information hence it avoids the overhead associated with exchanging control information and computation heavy path calculations. Our algorithm doesn't do a route discovery, which eliminates the typical delay associated with reactive routing algorithms and makes it oblivious of node movement.

In case of 2 dimensional node arrangement a small petal width ($\approx 15\%$) is sufficient for reliable delivery and less transmission overhead. For a random distribution of nodes $\approx 35\%$ petal width matches the delivery rate of flooding while reducing the overhead by half. Generally, single transmission zone scheme has a better performance but in presence of routing voids diverse transmission zones achieve a higher delivery rate with smaller petal widths (45% as compared to 55%). In all the cases the number of hops is nearly equal to flooding however for dense networks the transmission overhead for petal routing remains fairly constant whereas it increases linearly with flooding.

It would be interesting to study the comparison of performance metrics with other popular MANET routing protocols like Adhoc on Demand Distance Vector Routing (AODV)

or Optimized Link State Routing protocol or with geographic routing protocols like Reactive-Greedy-Reactive routing protocol. A direct comparison with the metrics from these protocols shall give a better understanding of the scenarios where petal routing would be more suitable. Another interesting direction would be employ petal routing protocol in actual FANET network and study its performance and suitability in a real life scenario.

REFERENCES

- [1] Basagni, S. et al. "A Distance Routing Effect Algorithm for Mobility (DREAM)". *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom '98. Dallas, Texas, USA: ACM, 1998, pp. 76–84.
- [2] Bekmezci, Äřlker et al. "Flying Ad-Hoc Networks (FANETs): A survey". *Ad Hoc Networks* **11.3** (2013), pp. 1254–1270.
- [3] Biswas, T. & Dutta, R. "Spatially Diffuse Pathsets for Robust Routing in Ad Hoc Networks". *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*. 2011, pp. 1–6.
- [4] Cadger, F. et al. "A Survey of Geographical Routing in Wireless Ad-Hoc Networks". *IEEE Communications Surveys Tutorials* **15.2** (2013), pp. 621–653.
- [5] Durocher, S. et al. "On routing with guaranteed delivery in three-dimensional ad hoc wireless networks". *Wireless Networks* **16.1** (2010), pp. 227–235.
- [6] Flury, R. & Wattenhofer, R. "Randomized 3D Geographic Routing". *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*. 2008.
- [7] GPS.gov. *GPS accuracy*. 2017. URL: <https://www.gps.gov/systems/gps/performance/accuracy/> (visited on 06/15/2018).
- [8] Haas, Z. J. & Liang, B. "Ad hoc mobility management with uniform quorum systems". *IEEE/ACM Transactions on Networking* **7.2** (1999), pp. 228–240.
- [9] Hayat, S. et al. "Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint". *IEEE Communications Surveys Tutorials* **18.4** (2016), pp. 2624–2661.
- [10] Huang, H. et al. "Three-dimensional geographic routing in wireless mobile ad hoc and sensor networks". *IEEE Network* **30.2** (2016), pp. 82–90.
- [11] Instruments, T. *WL18xxMOD WiLink™8 Single-Band Combo Module âĀŞ Wi-Fi @, Bluetooth @, and Bluetooth @Low Energy (LE)*. URL: <http://www.ti.com/lit/ds/symlink/wl1805mod.pdf> (visited on 06/02/2018).
- [12] John, A. A. B. & Dutta, R. "Cooperative trajectory planning in an intercommunicating group of UAVs for convex plume wrapping". *2017 IEEE 38th Sarnoff Symposium*. 2017, pp. 1–6.

- [13] Ko, Y.-B. & Vaidya, N. H. “Location-aided Routing (LAR) in Mobile Ad Hoc Networks”. *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom '98. Dallas, Texas, USA: ACM, 1998, pp. 66–75.
- [14] Kuhn, F. et al. “An Algorithmic Approach to Geographic Routing in Ad Hoc and Sensor Networks”. *IEEE/ACM Transactions on Networking* **16.1** (2008), pp. 51–62.
- [15] Li, J. et al. “A Scalable Location Service for Geographic Ad Hoc Routing”. *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. MobiCom '00. Boston, Massachusetts, USA: ACM, 2000, pp. 120–130.
- [16] Liu, S. et al. “Hybrid Position-Based Routing Algorithms for 3D Mobile Ad Hoc Networks”. *2008 The 4th International Conference on Mobile Ad-hoc and Sensor Networks*. 2008, pp. 177–186.
- [17] London, U. C. *QrSim: Multiplatform quadrotor simulator*. 2014. URL: <https://github.com/UCL-CompLACS/qrsim> (visited on 06/02/2018).
- [18] London, U. C. *QrSim: Multiplatform quadrotor simulator Manual*. 2014. URL: <https://github.com/UCL-CompLACS/qrsim/raw/master/doc/manual.pdf> (visited on 06/02/2018).
- [19] Mauve, M. et al. “A survey on position-based routing in mobile ad hoc networks”. *IEEE Network* **15.6** (2001), pp. 30–39.
- [20] Nardi, R. D. *The QRSim Quadrotor Simulator*. Tech. rep. RN/13/08. Gower Street, London UK: Department of Computer Science, University College London, 2013.
- [21] Oubbati, O. S. et al. “A survey on position-based routing protocols for Flying Ad hoc Networks (FANETs)”. *Vehicular Communications* **10** (2017), pp. 29–56.
- [22] Ramaswamy, R. et al. “Characterizing network processing delay”. *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*. Vol. 3. 2004, 1629–1634 Vol.3.
- [23] Rothery, R. W. “Car following models”. *Trac Flow Theory* (1992).
- [24] Services, M. D. *What Really Happened with Drones in 2017*. 2018. URL: <https://mydroneservices.com/what-really-happened-with-drones-in-2017/> (visited on 06/12/2018).
- [25] Stojmenovic, I. “Position-based Routing in Ad Hoc Networks”. *Comm. Mag.* **40.7** (2002), pp. 128–134.

- [26] Wang, Z. et al. “A study of spatial packet loss correlation in 802.11 wireless networks”. *IEEE Local Computer Network Conference*. 2010, pp. 568–575.
- [27] Wegener, S. “UAV Over-the-Horizon Disaster Management Demonstration Projects” (2000).

APPENDIX

APPENDIX

A

SAMPLE PERFORMANCE METRICS

In this section, we present the results of a pair of nodes with inter node distance at the median of all the node pairs in the simulation volume. There are 36 nodes randomly distributed the flying volume and the simulation state is the same as in Section 5.1.

In Fig. A.1 we can notice that the PDR exceeds 90% for HTL greater than two. This can also be confirmed in Fig. A.3 where the average number of hops made by successfully delivered packets is approximately equal to two. Due to the small number of hops there is no noticeable difference for the packet delivery ratio in single or diverged transmission zones. Moreover, the average number of transmissions increase for flooding because increase in total number of transmissions is much higher than increase in delivery ratio.

Delivery ratio. Type= random, Iterations= 15,
Node pairs= 1, Avg Distance = 264 m, Packets= 250

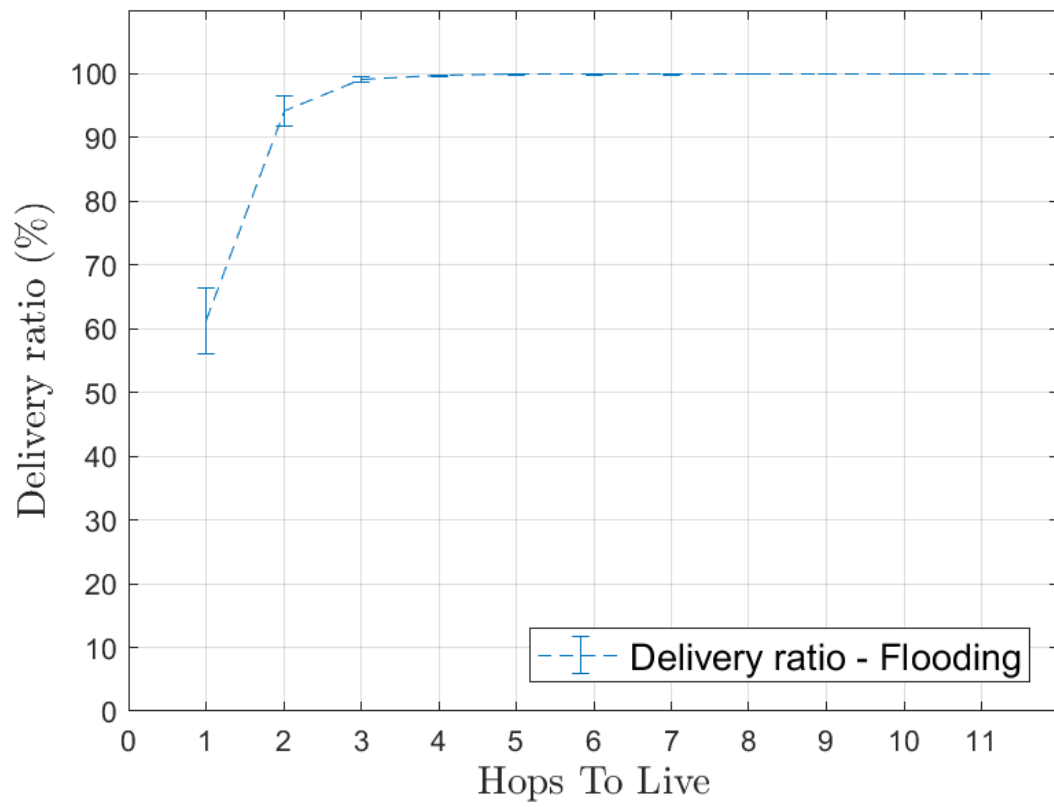


Figure A.1 Packet delivery ratio in random node distribution for flooding algorithm for median inter drone separation

Delivery ratio. Type= random, Iterations= 15
Node pairs= 1, Avg Distance = 264 m, Packets= 250

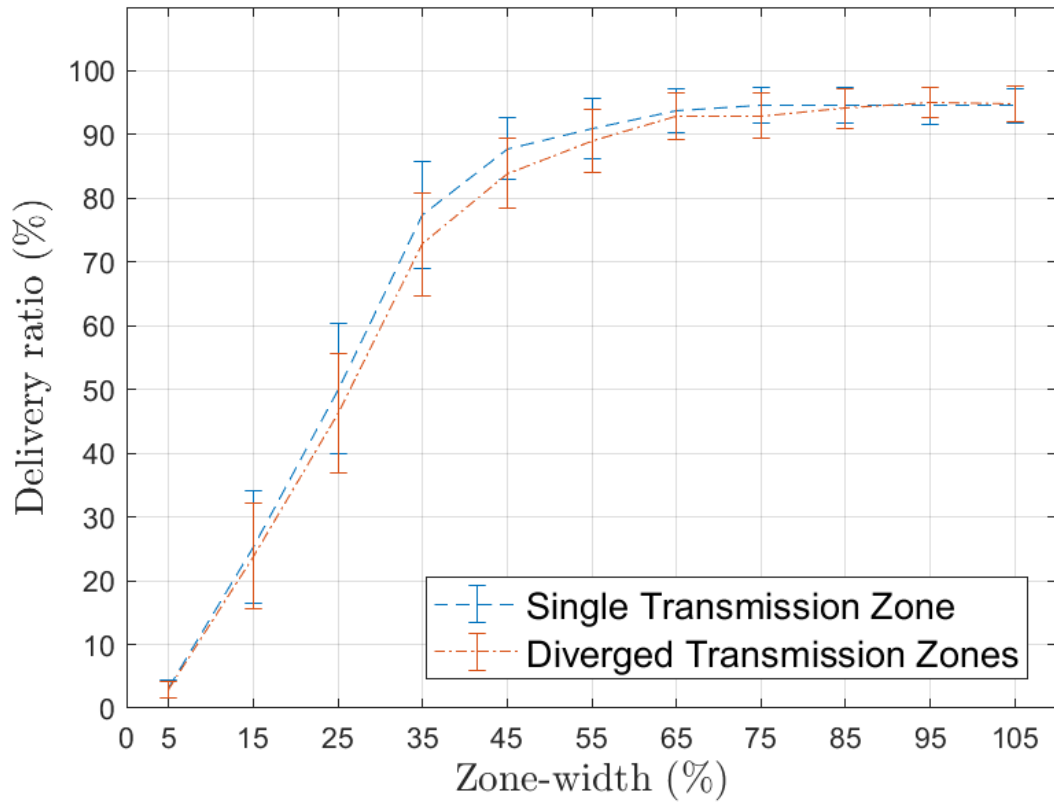


Figure A.2 Packet delivery ratio in random node distribution for petal routing for median inter drone separation

Hop Count. Type= random, Iterations= 15
Node pairs= 1, Avg Distance = 264 m, Packets= 250

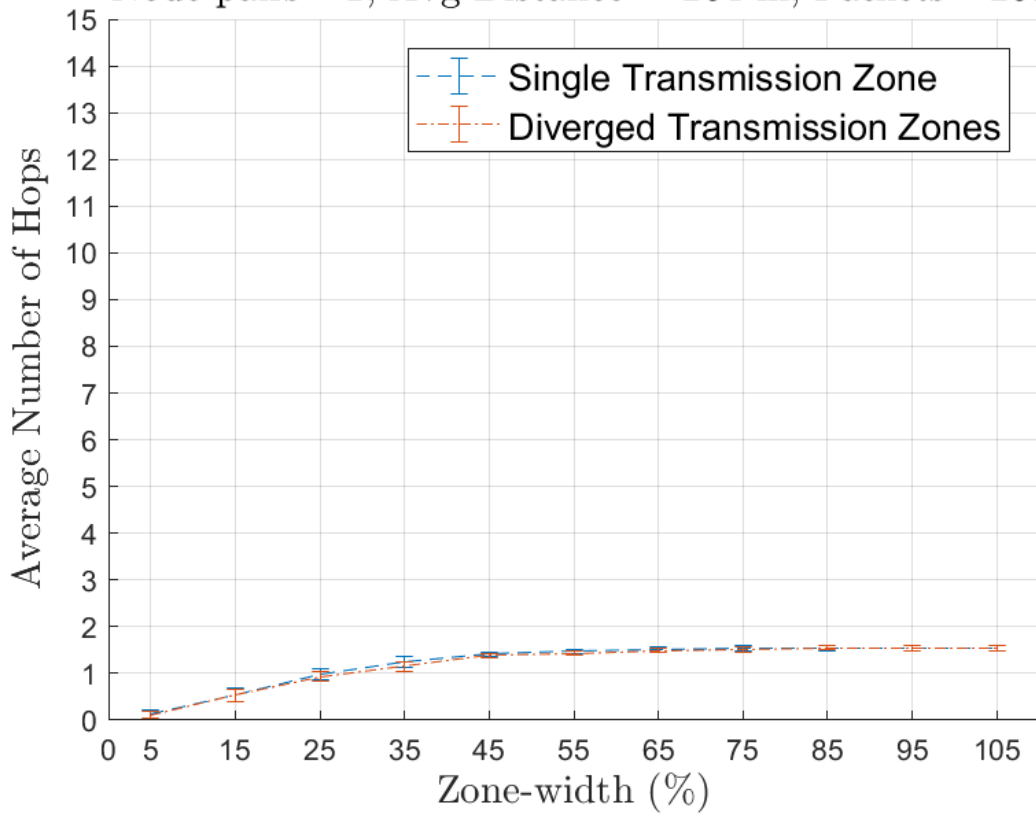


Figure A.3 Average number of hops by successfully delivered packets

Overhead. Type= random, Iterations= 15,
Node pairs= 1, Avg Distance = 264 m, Packets= 250

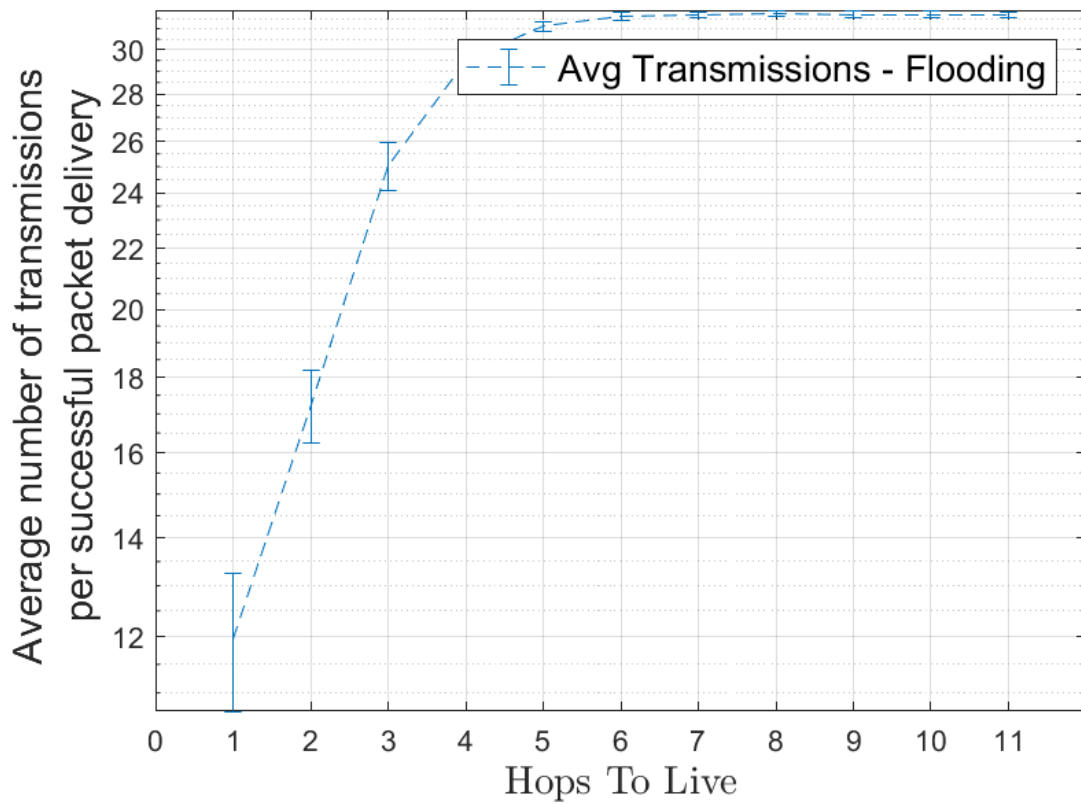


Figure A.4 Average number of transmissions per successful packet delivery in random node distribution for flooding algorithm

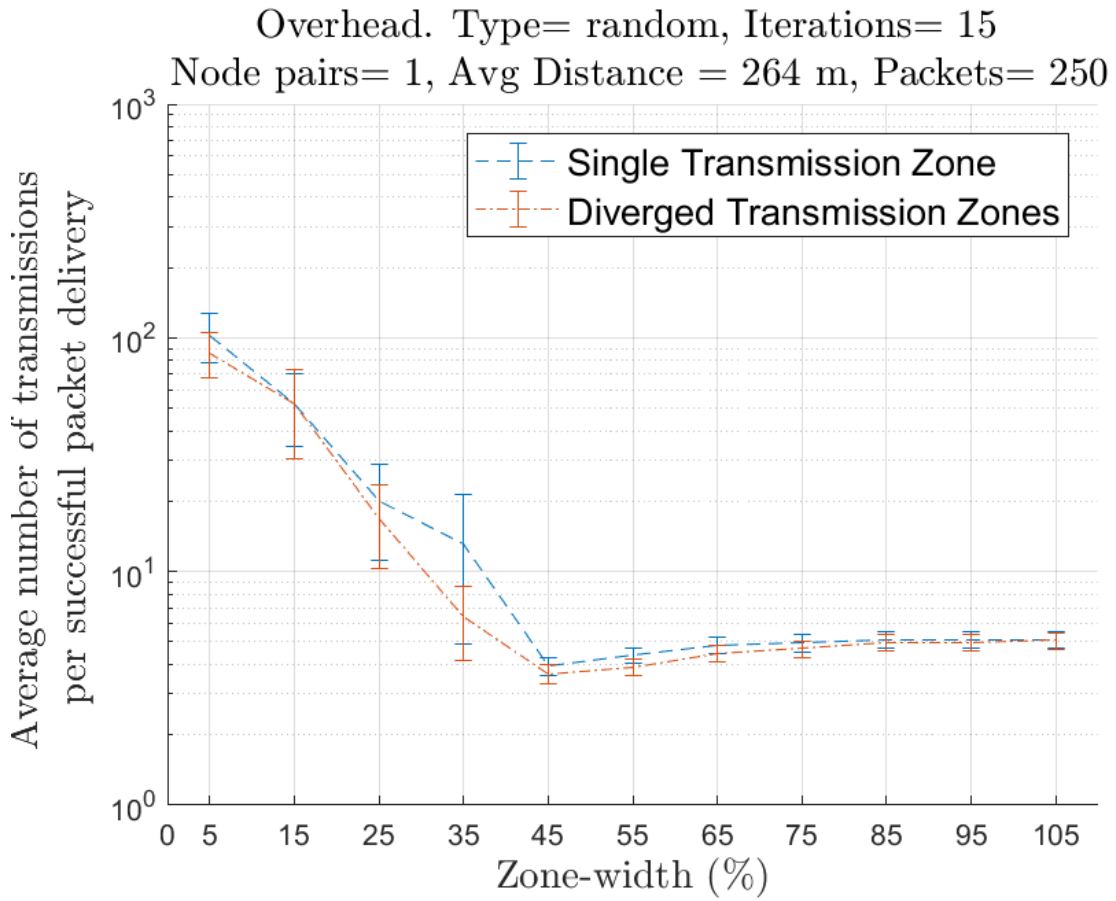


Figure A.5 Average number of transmissions per successful packet delivery in random node distribution for petal routing