Range-based Relative Navigation for Femto-spacecraft Swarms

Thomas Timmons

Submitted in fulfilment of the requirements
for the Degree of Doctor of Philosophy

James Watt School of Engineering
College of Science and Engineering
University of Glasgow

May 2023

Abstract

Femto-spacecraft, characterised by a total mass of under 100 g, are the smallest category of active spacecraft that are currently classified. If this technology is scaled to use many devices for distributed sensing applications, there can be unique functional benefits over traditional space systems. Determining the location of femto-spacecraft within a networked swarm would add utility for scientific investigation and enable in-orbit navigation without relying on Earth-based tracking or on-board GPS. With extremely limited space and computing power on such platforms, this thesis proposes methods for relative navigation enabled by range estimates obtained via networking. This is experimentally demonstrated using received signal strength indication (RSSI) data as a coarse range metric within a highly interconnected network of devices.

The first part of this thesis is the development of a series of range-based relative positioning algorithms intended for centralised, decentralised, and distributed computation. These are tested in simulation by modelling varying levels of inaccuracy in the range estimates provided. Relative navigation techniques are then presented, modelling the scenario of a swarm of femto-spacecraft deployed into low Earth orbit from a larger carrier spacecraft. Several ejection strategies are proposed using a model of the relative dynamics to propagate trajectories. These scenarios are simulated for real-time relative navigation using Kalman filtering techniques that use the relative positioning algorithms as partial linear state observers.

A network communications protocol enabling full interconnection within the swarm is then developed for experimental testing of these navigation strategies. Small development kits are used as femto-spacecraft proxies for embedded software development. Following this, a path loss model is experimentally developed to characterise the relationship between RSSI and range in an outdoor testing environment over a length-scale of 1-150 m. This model is then used to convert RSSI data into range estimates, used as inputs to the relative positioning algorithms. Relative navigation is demonstrated on an outdoor 120 m $\times$ 60 m sports pitch. This demonstration highlights the ability of the algorithms to fuse coarse proximity data and localise without any additional sensors or equipment.

Finally, the application of distributed in-situ sensing is analysed in simulation to investigate the trade-offs of using a larger number of devices of lower positioning accuracy and sensor measurement accuracy, as a comparison between the use of femto-spacecraft swarms and traditional space systems.

Acknowledgements

I firstly want to thank my supervisor, Colin McInnes, who has made pursuing this PhD an incredibly rewarding and thoroughly enjoyable experience for me with all his guidance, support, and patience. Thank you for giving me the opportunity to work on this project and for all the amazing opportunities and experiences I have had over the past three and a half years. I also want to thank the micro-scale and macro-scale research theme leads in the Emerging Space Technologies programme, James Beeley and Gilles Bailet, for all their insight and support throughout this research and for our many useful discussions.

I would also like say thank you to my examiners, Leonard Felicetti and Euan McGookin, and to the viva convener, Andrea Cammarano, for all your time, consideration and valuable suggestions that have improved this thesis.

Next, I must thank a lot of people who made the experimental work described within this thesis possible and gave up significant amounts of their time to help me. Thank you to Allan Cathcart and all the ground staff team at the University of Glasgow's Garscube Sports Complex for providing access to the outdoor pitches and for all their assistance. Many thanks to Chris, Colin, Bonar, Joe, and Gilles, for helping to set up and conduct the outdoor navigation experiments with me at Garscube over many months. I would also like to thank Afesomeh Ofiare and Thomas O'Hara for providing access to the anechoic chamber facility and all their assistance in characterising the antennas used in testing. Thanks also to James Beeley for making the antennas, and to Cameron Houston for soldering the modifications to the development kits.

I feel very lucky to have met a lot of great people during this project, and I want to thank everyone in the wider SET group for fostering such a fantastic research environment. There are too many people to list here, but to all the colleagues and friends who have helped make the past few years so enjoyable and rewarding, and for all the incredible help and support along the way, I am truly grateful.

Finally, I want to thank all my family and friends. Thank you especially to my mum and dad, who have always supported and encouraged me throughout everything. Thanks to my brothers, Paul and Luke, for all their support, and a special mention and thanks to my sister Anne for all her proofreading. Lastly, thank you to Emma, for all your love.

Declaration

I declare that, except where explicit reference is made to the contribution of others, this thesis is the result of my own work and has not been submitted for any other degree at the University of Glasgow or any other institution.

Thomas Timmons
May 2023

Contents

List of Tables

List of Algorithms

List of Figures

Nomenclature

List of Symbols

A	system matrix
$\mathbf{a}$	position vector of anchor spacecraft
C	covariance matrix
c	linear coefficient
$\mathbb{E}$	mathematical expectation
e	zero column vector
F	matrix of partial derivatives
$\mathcal{F}$	relative coordinate frame
$\mathcal{GP}$	Gaussian process
G	antenna gain
$\mathbf{g}$	vector of partial derivatives
H	observation matrix
h	observation function
I	identity matrix
J	Jacobian matrix
K	Kalman gain
m	mean function
N	number of range estimates **or** number set
$\mathcal{N}$	normal distribution
n	number of femto-spacecraft
P	state covariance estimate **or** probability **or** power
PL	path loss
Q	process noise covariance **or** quadratic form of Z
$\mathbb{R}$	the real numbers
R	rotation matrix **or** sensor noise covariance
R_o	orbital radius
Rx	receive
r	range
$\mathbf{r}$	position vector **or** vector of uncorrelated range estimates
T	translation matrix **or** period
Tx	transmit

t	time
U	unitary matrix
V	volume **or** unitary matrix
$\mathbf{v}$	velocity vector
$\mathbf{w}$	process noise
X	matrix of position vectors
$\mathbf{x}$	position vector **or** state vector
Y	symmetric matrix
Z	symmetric positive semidefinite matrix
$\mathbf{z}$	state measurement
α	slack variable for inter-spacecraft inequality
β	slack variable for anchor-spacecraft inequality **or** training coefficient
γ	path loss exponent **or** gradient descent iterator
δ	error in measurement
ε	error in result
ζ	noise level
θ	longitude **or** polar angle **or** residual
λ	wavelength
μ	standard gravitational parameter of an orbital body **or** mean
ν	observation noise
Σ	diagonal matrix in singular value decomposition
σ	standard deviation
Φ	state transition matrix
ϕ	latitude **or** azimuth angle
ω_n	mean motion of the femto-spacecraft deployer
χ	zero mean normal random variable
∇	gradient
$\triangleq$	is equal by definition to
$\equiv$	is identical to
$\approxeq$	is approximately equal to
$\sim$	is distributed according to
$\dot{\square}$	first derivative with respect to time
$\ddot{\square}$	second derivative with respect to time
$\square^T$	matrix transpose
$\square_k$	at discrete time step k
$\square^-$	predicted
$\square^+$	measurement updated
$\hat{\square}$	estimate
$\lvert\square\rvert$	matrix determinant
$\lVert\square\rVert$	norm

List of Acronyms

ADCS	attitude determination and control system
AWGN	additive white Gaussian noise
API	application programming interface
CCS	Code Composer Studio
COTS	commercial-off-the-shelf
CW	Clohessy-Wiltshire
EKF	extended Kalman filter
FSPL	free space path loss
GPR	Gaussian process regression
GPS	Global Positioning System
IDE	integrated development environment
IMU	inertial measurement unit
IROD	initial relative orbit determination problem
ISM	industrial, scientific, and medical
KF	Kalman filter
LED	light-emitting diode
LEO	low Earth orbit
LMI	linear matrix inequality
MCU	microcontroller unit
MEMS	micro-electro-mechanical systems
NASA	National Aeronautics and Space Administration
NLLS	non-linear least squares
NPDF	normal probability density function
PCB	printed circuit board
PLM	path loss model
RF	radio frequency
RSSI	received signal strength indication
RTOS	real-time operating system
SAIL	sensor and actuator interface layer
SAR	sparse aperture radar
SDK	software development kit
SDP	semidefinite programming
SET	space and exploration technology
SLAM	simultaneous localisation and mapping
SMA	sub-miniature version A
SRAM	static random-access memory
SRP	solar radiation pressure
SVD	singular value decomposition

TI	Texas Instruments
TRR	translation, rotation, and reflection
UAV	uncrewed aerial vehicle
VNA	vector network analyser
WMM	World Magnetic Model
WSN	wireless sensor network

Chapter 1

Introduction

In recent years, there has been a considerable shift towards using small, low-cost satellites and spacecraft for increasingly more ambitious applications [1]. This is made possible largely due to continued advances in miniaturisation for consumer technology. As this trend continues, what is feasible at the smallest of spacecraft length-scales will only increase further, and the question arises of how this technology can be scaled to maximise its utility. At present, an emerging class of centimetre-scale femto-spacecraft, characterised by a total mass of under 100 grams, can in principle be rapidly produced at low cost and in large numbers [2]. Using micro-electro-mechanical-systems technology, femto-spacecraft can be equipped with many of the elements of larger spacecraft, such as attitude determination and control, wireless communications, and sensor suites, contained within a single printed circuit board with a side length of only several centimetres. A wide range of potential applications are possible if many of these devices were to be deployed from a larger carrier spacecraft, such as a CubeSat, and dispersed into orbits neighbouring the carrier [3].

When combined to form a networked swarm, femto-spacecraft may offer many benefits over single small spacecraft for similar applications. A principal advantage is the ability to record large-scale, simultaneous and spatially distributed measurements. The potential to deconvolve spatial and temporal ambiguities in phenomena in a way that is not possible with existing distributed space systems may offer new insights and facilitate novel sensing approaches [4]. Knowledge of femto-spacecraft positions within a swarm would be highly useful, not just for associating measurements to positions for characterisation, but for navigating in-orbit. Primarily, this would enhance the capability of these resource-constrained devices as sensor platforms. This thesis proposes methods for enabling relative navigation for femto-spacecraft swarms when considering the extremely limited space and computing power available on individual platforms. A key focus is practical implementation and experimental demonstration of the methods presented.

1.1 Research Objectives and Contributions

The central aim of this thesis is to develop range-based relative navigation strategies for a networked swarm of femto-spacecraft, from mathematical modelling and simulation based approaches, through to experimental demonstration on hardware. Consequently, the main research objectives follow the central steps in this process:

1. Develop range-based relative positioning algorithms for femto-spacecraft swarms
2. Implement relative navigation filtering approaches to improve real-time state estimation
3. Experimentally demonstrate range-based relative navigation
    (a) Develop network protocols to support swarm communications
    (b) Develop a range metric from received signal strength indication (RSSI) data
4. Investigate distributed sensing strategies for femto-spacecraft swarms

The first main contribution is the development and implementation of range-based relative positioning algorithms for femto-spacecraft swarms. Provided range estimates between swarm members are available, algorithmic approaches can be developed in order to calculate relative positions. Algorithm development in this thesis is not restricted to using RSSI as a range metric and should be applicable to any range-based measurement technique. The objective is also to make the approaches as general as possible, so centralised, decentralised, and distributed approaches are considered. Another key contribution is the development of relative navigation filters that can use the algorithms implemented to provide a real-time relative navigation system. State estimation can be improved with time using a relative dynamics model of the expected swarm motion combined with relative positioning algorithms used as partial linear state observers.

To test and demonstrate relative navigation experimentally, it is first necessary to develop network protocols to facilitate data passing between swarm members using an RF communications link. This is an integral part of this thesis as the interconnection required to obtain all RSSI data via reliable networking underpins the experimental testing. This is achieved through the implementation of the relative navigation methodologies with embedded software on femto-spacecraft scale hardware, representative of what processing power and storage is available for a femto-spacecraft in Earth orbit. Then, the conversion of RSSI data into a range metric that can be used with relative positioning algorithms is necessary before experimental demonstration is possible. RSSI data needs to be scaled with the development of a suitable path loss model for testing, along with antenna modelling and characterisation to account for anisotropy in radiation patterns. With testing, the main contribution in this thesis is experimental demonstration of the capabilities, potential and limitations of this approach.

The final contribution of this thesis is simulation-based analysis of how relative navigation

may be utilised within distributed sensing applications. The objective is to consider operational trade-offs when using femto-spacecraft swarms, characterised by far larger numbers of spacecraft relative to standard distributed space systems. Additionally, collaborative work not contained in this thesis analysed how femto-spacecraft swarms can be used for the atmospheric characterisation of Mars [5] and for in-situ measurements at inaccessible locations in space [6]. It is intended that distributed sensing applications can be shown in combination with the simulation and experimental testing of relative navigation, to bring together all aspects of the utility of femto-spacecraft swarms.

1.2 List of Publications

Published research materials containing work presented within this thesis are listed below [7–9]:

- T. Timmons, J. Beeley, G. Bailet, and C.R. McInnes, "Range-based Relative Navigation for a Swarm of Centimetre-scale Femto-spacecraft", *Journal of Guidance, Control, and Dynamics*, Vol. 45, No. 9, pp. 1583-1597, 2022. doi: 10.2514/1.G006261
- T. Timmons, J. Beeley, G. Bailet, and C.R. McInnes, "Experimental Testing of Range-based Relative Positioning Strategies for a Swarm of Centimetre-scale Femto-spacecraft", *73rd International Astronautical Congress*, IAC-21-B4.IP.68691, Paris, France, 2022.
- T. Timmons, J. Beeley, G. Bailet, and C.R. McInnes, "Massively Parallel In-situ Sensing using Femto-spacecraft Clouds," *72nd International Astronautical Congress*, IAC-21-B4.7.4.64209, Dubai, United Arab Emirates, 2021.

Additionally, relevant collaborative research materials that are not presented within this thesis are listed below [5, 6]:

- T. Timmons, G. Bailet, J. Beeley, and C.R. McInnes, "Mars Atmospheric Characterization with a ChipSat Swarm", *Journal of Spacecraft and Rockets*, Vol. 58, No. 5, 2021, pp. 1453-1460. doi: 10.2514/1.A34970
- T. Timmons, G. Bailet, J. Beeley, and C.R. McInnes, "Solar System Swarm Probes: An Earth-based Technology Demonstrator", *18th International Planetary Probe Workshop*, 2021.

1.3 THESIS OUTLINE

This thesis is divided into eight chapters. As outlined below, Chapters 3-7 present the original research. Chapter 2 provides a literature review highlighting the current state of the art, while Chapter 8 concludes the thesis and provides recommendations for future work.

- **Chapter 3** firstly presents the femto-spacecraft swarm relative positioning problem, discusses the relevant challenges and explains the approaches taken. Range-based algorithms are then developed to operate with centralised, decentralised, and distributed computation with high network interconnectivity. Finally, algorithm performance is analysed via a series of test cases in simulation.

- **Chapter 4** describes relative navigation techniques for a femto-spacecraft swarm. This is presented in the context of a swarm being deployed in LEO from a larger deployer spacecraft using a relative dynamics model. Two predominant swarm ejection strategies are presented and discussed. Then, Kalman filters are implemented by applying the output of algorithms developed in Chapter 3 as discrete-time measurements to improve swarm state estimation with time.

- **Chapter 5** then outlines the implementation of networking in embedded software on small development kits used as femto-spacecraft proxies in experimental testing. This chapter bridges the gap between the modelling and simulation-based work in Chapters 3-4 and the experimental demonstrations in Chapter 6 through the implementation of a fully interconnected network protocol to facilitate testing. The network protocol is presented as an operational process to accumulate swarm data. The embedded software development in C using small development kits is then presented.

- **Chapter 6** details the experimental demonstration of range-based relative navigation using network received signal strength indication (RSSI) data. Firstly, a path loss model (PLM) is derived with discussion of the outdoor testing environment used and characterisation of the antennas used. A series of test cases are then presented for relative positioning and relative navigation demonstrations. Finally, in-orbit implications of the results are discussed.

- **Chapter 7** presents a distributed in-situ sensing strategy using femto-spacecraft swarms. The use of femto-spacecraft swarms for massively parallel distributed sensing is firstly discussed through potential novel applications. This focuses on comparing the utility of using higher numbers of less accurate measurements, representative of a femto-spacecraft swarm contrasted with traditional sensing approaches. A methodology is then developed for characterisation and reconstruction of phenomena such as scalar and vector fields using discrete measurements. Finally, a series of test cases where the number, sensor accuracy and relative navigation accuracy are varied are simulated and discussed.

Chapter 2

Literature Review

This chapter firstly presents a review of existing femto-spacecraft concepts and their potential benefits and applications when used as distributed swarming systems. Then, a review of relative navigation techniques for distributed systems is presented, focusing on the advantages of relative navigation in a femto-spacecraft swarm and potential approaches with current technology.

2.1 Background

There are many known instances in nature of swarms collectively accomplishing objectives that would otherwise be beyond the ability of their individual members. Schools of fish assemble to appear as one very large and fast spinning ball to confuse predators and enhance collective threat detection [10], while many bird species flock to deter predators and improve foraging performance [11]. Key to this type of behaviour is the ability of individual swarm members to self-organise, a trait commonly referred to as swarm intelligence, characterising the emergent behaviours that swarms exhibit [12].

Self-organising systems are not structured by any hierarchy or external commands, but by their own natural processes, and the emergence of order within them is complex [13]. Any patterns that emerge at a global level of a swarming system are the result solely of interactions between the lower-level components. However, not all cooperative distributed systems are by any means self-organising. Migrating geese flying in a 'V formation' emphasise this distinction. Individual geese take turns in position within a precise and highly structured formation that is understood to improve flight efficiency and assist in coordination [14]. Hierarchy is central to the 'V-formation' of migrating geese, yet individual geese are still able to migrate on their own. Flocking is entirely different; there is no apparent structure, and no specific task or role that any individual does that can be considered vital to another member. Despite this, a bird outside the flock cannot deter predators on its own. In short, swarms are unique because their members cannot individually perform the task that a swarming system as a whole is able to accomplish, and do so in a way that does not necessitate a centralised command structure.

In science and engineering, a central motivation for developing artificial swarming systems is to take advantage of these behaviours to offer operational benefits, application-specific improvements over other existing approaches, or both. Swarms are highly robust (reliable and fault tolerant) against system failure. This is not because any of their members need to be robust against failure individually, but because no member of a swarm is essential to the collective operation [12]. In large swarms, individual members are effectively redundant. This is because individual swarm member failure would simply degrade performance but not render the overall system ineffective. As a result, a robust system can be implemented that permits a significant proportion of unit failure. This means that swarm members do not need to be free of single-point failures and can be developed accordingly, with simpler designs that are quicker and cheaper to prototype [15]. The removal of complexity from systems can have operational benefits, mitigate against risks, and streamline process flow. If each swarm member is responsible for simple tasks, swarms can be fully modular and disaggregated (separated into individual elements), and failed units can simply be retired with little impact on the collective swarm objective. Difficult tasks can be accomplished without complex systems. This makes swarms highly scalable and adaptable to a variety of applications in the way that they operate. For distributed systems operating in space, all these attributes increase the probability of mission success [3].

Distributed space systems can be broadly grouped into the classifications of constellations, clusters, and swarms [13, 16]. Satellite constellations are used in global navigation and communications, placing satellites in carefully designed orbits phased to avoid collisions and to require little station-keeping. By contrast, satellite clusters operate in close proximity, requiring a precise orbital determination and control system and formation-flying to maintain their spatial structure. Although entirely different systems, constellations and clusters share the need for a precise and highly controlled structure to maintain their operation. Both constellations and clusters can be contrasted with satellite swarms, which can be widely dispersed without the inherent need of a precise structure or strict hierarchy in fulfilling their objectives [17, 18]. Moreover, if using very small satellites, it can be expected that the number of devices in a swarm could potentially be extremely large. Such systems can be thought of as space-based wireless sensor networks (WSNs) dispersed to perform distributed sensing and collect simultaneous data over a large volume of space [19]. As in nature, these systems are envisaged as performing objectives well beyond the limited capability of any individual satellite member, without a precise and highly controlled structure central to their operation [4].

Over the past two decades, small satellites have become an increasingly popular means of providing fast and low-cost access to space, both for research and industry [20]. The 10 cm $\times$ 10 cm $\times$ 10 cm 1U[1] CubeSat standard [21] has proliferated this trend, proving to be an incredibly popular form factor. Since the launch of the first CubeSat in 1999, there have now been over two thousand CubeSats launched [22], with over a thousand in the past five years

[1] this denotes 'one unit'. For instance, a 3U CubeSat is 30 cm $\times$ 10 cm $\times$ 10 cm.

alone [23]. While CubeSats were originally intended as an educational tool for students, they have now become a standardised form factor for testing new small satellite technologies in low Earth orbit (LEO) [24, 25]. Additionally, CubeSats offer the potential for low-cost and high-risk secondary payloads on flagship missions to deep space. This was highlighted by the twin MarCo CubeSats that launched with NASA's Mars InSight Lander Entry Capsule, successfully providing a communications relay between the Mars Insight rover and Earth [26].

There are many potential applications of CubeSat-based distributed space systems. As distributed space systems, the QB50 project highlighted the potential of a network of CubeSats to conduct thermosphere and re-entry research [27]. CubeSat-based concepts have been proposed to harness the unique ability of swarms in applications as wide ranging from studying the ionosphere in LEO [28] to radio astronomy using swarms orbiting the Moon [29]. CubeSats swarms could also be used to predict space weather by studying the radiation environment around Earth [30]. Despite using the CubeSat standard, the relatively large mass of these platforms makes them less suitable or in some cases impractical for applications such as multi-point sensing, atmospheric characterisation [31], distributed field and particle measurements [32], or surface science [33], where there can be significant spatial and temporal variabilities. For these applications, large numbers of much smaller spacecraft may facilitate new sensing approaches and offer further insights beyond what would be practical with CubeSats. Femto-spacecraft swarms may bridge the gap between what is financially practical and operationally possible using a relatively small number of CubeSats. A swarm of femto-spacecraft deployed from a CubeSat could therefore harness the operational benefits of swarming systems to open up a wide range of potential applications.

2.2 Femto-spacecraft designs and concepts

The availability of miniaturised commercial-off-the-shelf (COTS) components, such as micro-electro-mechanical-system (MEMS) equivalents of traditional technologies [34], makes the development of active femto-spacecraft viable for research groups [2, 35, 36]. At present, several research groups have femto-spacecraft projects under development, including the Space and Exploration Technology Group at the University of Glasgow [37]. In recent years, there have been many design concepts and technology demonstrations from different universities for femto-spacecraft focusing on using COTS and MEMS technology. Many of these have the intention of enabling distributed sensor networks for space use. This section provides an overview of these developments. Application-specific discussions of femto-spacecraft swarms are given in Section 2.3.

Researchers from Cornell University have proposed and developed several different femto-spacecraft concepts over the past fifteen years. Atchison and Peck [17] first proposed a 'spacecraft-on-a-chip' concept as early as 2007, with the aim of demonstrating translation control using the Lorentz force as a means of electromagnetic propulsion. Later work proposed using solar radiation pressure (SRP) with a 25 µm thick spacecraft-on-a-chip concept as a millimetre-scale

solar sail [38]. By 2008, Cornell's Sprite ChipSat[2] concept was in development with the aim of demonstrating the capabilities of spacecraft-on-a-chip devices [39]. A 'Sprite', as shown in Fig. 2.1, fits a solar cell, microcontroller, radio, 3-axis magnetometer, and 3-axis MEMS gyroscope onto a 3.5 cm × 3.5 cm printed circuit board (PCB) with a total mass of 5 g. By 2011, Manchester [18] had founded the KickSat project, developing a CubeSat-based deployer for ChipSats and pioneering the concept of using femto-spacecraft swarms in LEO deployed by a larger carrier spacecraft. After a failed deployment attempt in 2014 (KickSat-1), over one hundred ChipSats were successfully deployed in LEO from the 3U CubeSat Kicksat-2 (Fig. 2.1) in 2019 [40]. In this technology demonstration, the ChipSats were able to transmit RF signals before re-entering the atmosphere. Recent developments from Adams and Peck [41] present the biologically inspired 5 cm × 5 cm 2.5 g Monarch spacecraft, with additional sensors and technological capabilities over earlier Sprite designs.

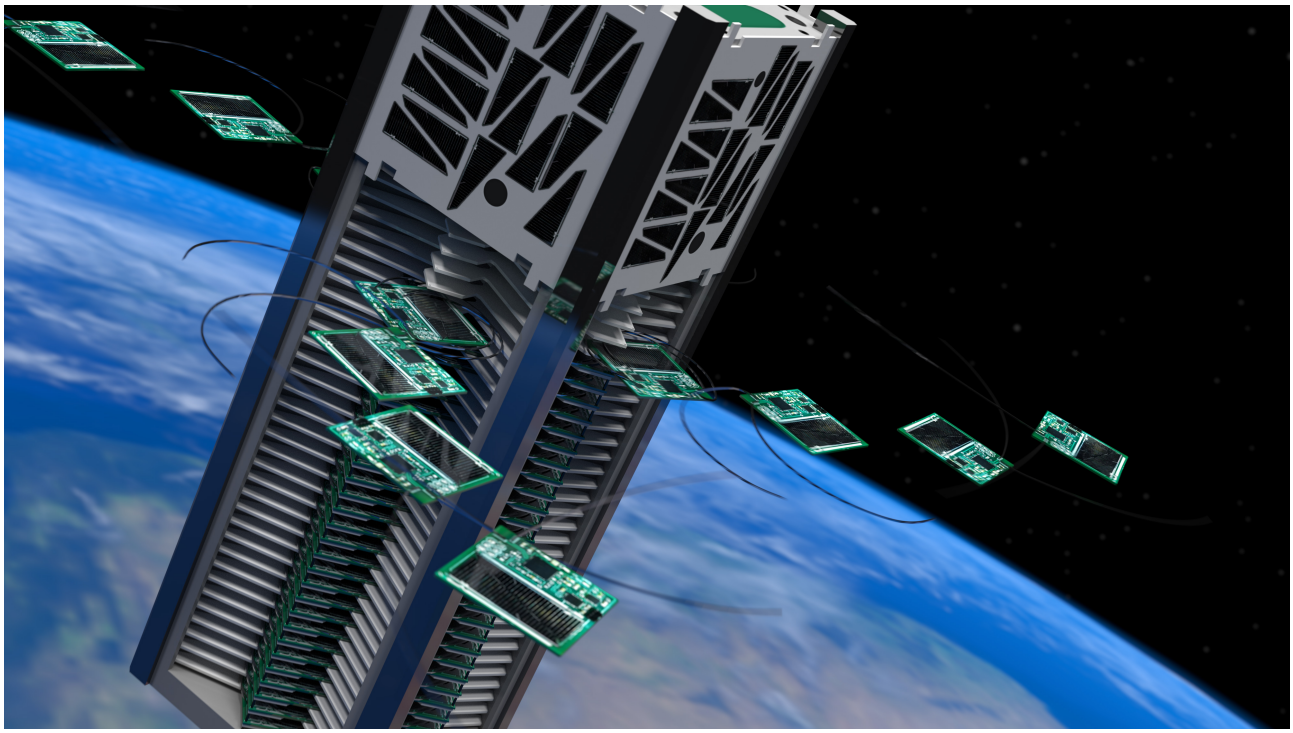


Figure 2.1: Render of the Sprite ChipSat deployment from KickSat-2[3] [42]

Other early developments in femto-spacecraft COTS technology come from Barnhart et al. [19, 43], who developed a satellite-on-a-PCB design - the 'PCBSat' - with the aim of demonstrating what capabilities could be achieved with a femto-spacecraft package used in distributed space applications. The design has a total mass of 70 g and takes a PC/104 card form factor (dimensions of 9 cm × 9.5 cm × 1 cm). Their research from 2009 found that for several applications of distributed systems, PCB-based designs offer a more cost-effective approach than CubeSats. As discussed in Section 2.3, the authors also propose several applications that may be enabled by femto-spacecraft sensor networks. By 2010, Stuurman and Kumar [44] had prototyped Ryerson University's femto-spacecraft, 'RyeFemSat'. This concept aimed to fur-

[2]satellite-on-a-chip
[3]Image credit: NASA

ther femto-spacecraft technology capabilities by demonstrating active attitude control at this length-scale using magnetorquers. Additionally, RyeFemSat featured a small optical camera, for either low-resolution Earth observation or high-resolution host satellite inspection tasks. The total mass of RyeFemSat is 25 g, with dimensions of 9 cm × 9 cm × 1 cm. This design's mass is much lower than the PCBSat but shares relatively large dimensions relative to the smaller Sprite and Monarch concepts.

In December 2013, researchers at the Pontificia Universidad Catolica del Peru [45] deployed a 8 cm × 5 cm × 2 cm 97 g femto-spacecraft called 'Pocket-PUCP' in LEO. This femto-spacecraft transmitted temperature data to a larger host satellite that deployed it. Yang et al. [46] developed the 'Stardust' femto-spacecraft, which was deployed in LEO in 2015 along with two larger satellites. In-orbit results showed that Stardust could network with other satellites and operate effectively.

There have been a considerable number of further concepts proposed over the past few years. This includes the WikiSat project from the Universitat Politecnica de Catalunya, which involved a number of student teams developing several versions of femto-spacecraft designs under 20 g between 2011-2015 [47, 48], presenting design budgets and prototypes for this strict mass limit. Hadaegh et al. [4] proposed the 'SWIFT' (swarms of silicon wafer integrated femto-satellites) concept in 2014, focusing on potential applications and swarm array configurations for distributed space systems as discussed in Section 2.3. In 2018, a team of students at UC Berkeley designed a KickSat-compatible femto-spacecraft similar to Sprites called 'SpinorSat' [35, 49].

The Space and Exploration Technology group at the University of Glasgow have been developing PCB-based femto-spacecraft concepts since 2018 [37]. The current platform under development comprises a 3.5 cm × 3.5 cm PCB containing a microprocessor with integrated wireless communications, an active 3-axis attitude determination and control system (ADCS) using miniaturised magnetorquers and a battery for power.

The concepts proposed to date for femto-spacecraft that have been developed into hardware have mainly focused on technology demonstration. This leaves application-specific use cases that would necessitate femto-spacecraft development in the first place open for further research and in-orbit demonstrations. This can be compared with the earliest CubeSat launches, as contrasted with how CubeSats are used in many applications today. At present, femto-spacecraft represent an emerging technology with the potential to offer improvements over existing approaches, but this has yet to be demonstrated. As would be expected, there are different advantages and limitations to all existing concepts based on the researchers' aims and envisioned applications. Common to all, however, is the desire to maximise functionality within as small a platform as possible.

2.3 Femto-spacecraft swarm applications

A swarm of femto-spacecraft may render many novel applications feasible by facilitating large-scale distributed sensing. This section provides an overview of applications and concepts that have been proposed for femto-spacecraft networks from the literature. The main applications include the potential for improved investigation and characterisation of near-Earth space phenomena, such as the atmosphere and magnetosphere, for space weather monitoring and to improve satellite operations and reliability with improved modelling and observational data. Other concepts include using femto-spacecraft swarms for Earth observation via distributed sparse aperture radar (SAR) interferometry, gravity field mapping, host satellite inspection, ballistic coefficient analysis for improved space debris modelling, and atmospheric re-entry analysis. Beyond near-Earth space, concepts proposed include impactor missions to asteroids and moons, planetary atmospheric characterisation, and even interstellar travel.

Early work by Barnhart et al. [50] reviewed previously proposed distributed space mission for small satellites and considered specific applications requiring simultaneous multi-point sensing that could only be enabled by large networks of extremely small satellites. The authors note applications such as: disaster monitoring for pre-emptive warning of volcanic eruptions and earthquakes, high spatial resolution imaging, and short-lived fast-response sensing networks in LEO and the upper atmosphere. Finally, the authors present a mission concept for low-resolution imaging of Earth using their satellite-on-a-chip 'PCBSat' design. In later work, the same authors proposed several femto-spacecraft sensor network missions to study ionospheric phenomena termed 'plasma bubbles' that are known to adversely affect satellite communications and navigation via Global Positioning System (GPS) receivers [19]. This is a form of in-situ space weather monitoring, offering the potential to improve understanding of regions of the ionosphere with many femto-spacecraft in a way not possible with conventional single-point measurements. This concept was later furthered by Balthazor et al. [51], where the authors found that forecasting ionospheric weather at lower latitudes could be improved by further observational data from several femto-spacecraft architectures.

Janson and Barnhart [3] later outline further proposed femto-spacecraft enabled mission concepts, including upper atmospheric density monitoring, terrestrial gamma ray flash monitoring, and satellite inspection tasks. In the case of atmospheric density modelling, the highly dynamic and variable density in the upper atmosphere makes aerodynamic drag difficult to predict. This can shorten the orbit lifetime of a spacecraft in LEO, such as when increased solar activity temporarily increases upper atmospheric density. The authors propose sampling the upper atmosphere directly with low-drag femto-spacecraft networks deployed over the course of a few years. It is proposed that individual femto-spacecraft could inflate to form high-drag spherical shells that would be able to de-orbit from an approximately 500 km orbit altitude over the course of a month. The in-situ observational data accumulated would be able to improve orbital drag estimates for improved satellite operations in the future. A clear downside to this concept would be the potential for space debris if (relatively) low-drag femto-spacecraft

are dispersed in LEO and fail to deploy a high-drag shell. However, the authors predict a worst-case de-orbit within a year. Most other concepts for LEO femto-spacecraft swarms focus on short-lived temporary networks that would re-enter the atmosphere and burn up within a few weeks of deployment. In the case of ChipSats, this is ensured by extremely low ballistic coefficients (for the Sprite ChipSat, approximately $1\,\mathrm{g\,cm^{-2}}$) [5, 18]. In the same paper, Janson and Barnhart also outline a femto-satellite network to monitor terrestrial gamma ray flashes, where the concept proposes that tens or hundreds of sensor points would potentially improve understanding of this phenomenon.

The concept of a satellite inspector is of course not limited to femto-spacecraft [52]. However, the prospect of being able to quickly eject a small disposable imager to assess the overall health of the host satellite by examining, for example, an antenna deployment failure could provide invaluable insights to extend satellite operations. A femto-spacecraft could be ejected from a CubeSat into a neighbouring orbit, and with translation control available from SRP or differential drag, be able to control its position relative to the host satellite. Attitude control could then enable the pointing necessary for imaging techniques. In 2008, Vladimorova et al. [53] proposed a concept using multiple CubeSats in part for inspection tasks that may be scalable to femto-spacecraft technology today. It is also possible that a number of femto-spacecraft in a swarm deployed from a larger carrier could provide host satellite inspector capability, even if this is not the main intended application.

Another potential application of femto-spacecraft swarms is for sparse aperture interferometry. Hadaegh et al. [4] propose the SWIFT space system architecture, presenting a random sparse aperture array configuration using thousands of femto-spacecraft. Cost analysis suggested this is a feasible approach over monolithic spacecraft systems to achieve substantially large effective apertures. Cao et al. [54] present a remote sensing concept using femto-spacecraft swarms enabled by SRP for orbit control. In addition to remote sensing, the concept proposed using the swarm as a space-based radar for target detection of, for instance, ships and aircraft. This approach focuses on offering low-cost but high-resolution radar imaging, achieved by using the swarm as a sparse way to receive radar echoes in orbit.

Impactor mission concepts to other planets, moons and asteroids have been proposed to obtain in-situ measurements throughout the solar system. These concepts harness the benefits of robustness in swarming systems with probabilistic approaches to mission success. Utilising the low ballistic coefficients of PCB-based femto-spacecraft concepts, Atchison et al. [38] proposed using their Sprite ChipSat firstly for Earth-based atmospheric re-entry analysis. The results found that Sprites could be designed to re-enter Earth's atmosphere from LEO at a low enough temperature to continue radio communications sharing atmospheric sensor measurements. Barker and Salazar [33] investigated the use of self-organising femto-spacecraft scale sensors for planetary exploration. Manchester and Peck [15] propose an asteroid impactor mission, modelling the swarm stochastically to estimate the likelihood of a proportion surviving impact. Further combined work from Draper and Cornell universities [55] later proposed a dual mission architec-

ture for planetary moon science with femto-spacecraft dispersed to combine remote and in-situ measurements by conducting gravity field mapping at Europa from a larger deployer prior to femto-spacecraft deployment and landing. Adams and Peck [41] conducted durability tests on the 'Monarch' ChipSat through emulation of lunar surface impact experiments, indicating the survivability of components. Collaborative research by this author (outside the scope of this thesis) proposed using a femto-spacecraft swarm for Mars atmospheric characterisation [5]. The concept proposed atmospheric characterisation during descent and surface science upon landing. Recent work by Castro et al. [56] presents an orbit-to-ground model for Monarch ChipSats used for planetary atmospheric re-entry analysis.

Other approaches take advantage of the ability to produce many femto-spacecraft at relatively low cost for highly ambitious applications and novel orbit control strategies. A study by Hein et al. [57] presented a feasibility analysis accelerating femto-spacecraft using lasers to one tenth the speed of light in a fifty-year fly-by mission to Alpha Centauri. This approach differs from previous self-propelled 'smart dust' devices approaches using SRP for propulsion [32, 58]. The clear operational and functional advantages with femto-spacecraft swarms mitigate traditional component failure concerns associated with conventional distributed space systems.

An underpinning motive behind developing this concept is to discover what functionality and new applications can be delivered at the smallest of spacecraft length-scales. Scaling the technology to large, networked swarms dispersed over a large volume of space is a desirable extension to current capabilities. This could enable compelling new applications for a range of mission scenarios.

2.4 In-orbit relative navigation

In-orbit relative navigation between a networked swarm of centimetre-scale femto-spacecraft would add considerable value to a range of space mission concepts and applications discussed in the previous section. Determining the location of femto-spacecraft relative to one another is essential in adding value to the data gathered in many mission applications [50], and in enabling swarm members to operate in close proximity to each other. Furthermore, this could prove invaluable for reliable spacecraft tracking in-orbit [3]. Relative positioning can maximise the utility of each swarm member, not only for scientific investigation, but also to navigate in orbit without relying on Earth-based tracking [59–61]. It is anticipated that absolute navigation would be enabled by the known position of the carrier spacecraft that is used to deploy the swarm. Addressing the specific hardware limitations at this length-scale, this thesis seeks to address how a swarm of femto-spacecraft could establish relative positioning and networking to enable a diverse range of applications.

Concepts for the in-orbit relative navigation between at least two spacecraft can be broadly grouped into GPS-based, vision-based, and RF-based approaches [13]. However, previous research and development has largely focused on small numbers of spacecraft, typically for ren-

dezvous and docking or navigation and control for formation-flying. In expanding a swarm from tens to hundreds of femto-spacecraft, there is a need to consider operational constraints such as computational and network scalability, real-time operations, and the small size and hardware limitations.

GPS-based approaches to relative navigation operate on the principle of differencing two absolute positions to obtain the relative positions of each satellite with respect to the other(s). Equipping femto-spacecraft with GPS receivers has been proposed for several femto-spacecraft concepts, including the Sprite and Monarch [15,41]. An obvious drawback to this is the restriction of only being able to operate in LEO. There are further issues of how power intensive this would be and the scalability of this approach for large swarms. GPS use currently represents a large fraction of CubeSat power budgets [62,63]. The design of energy management techniques may make this a more efficient option for femto-spacecraft, which have extreme power limitations. When operating in LEO, having a small fraction of GPS-equipped femto-spacecraft within a swarm could be a more feasible approach, as discussed in Chapter 3. For applications beyond LEO, and for scalability and power usage considerations, the question of how relative localisation is achieved for the situation where few or no swarm members have access to GPS needs to be addressed [64].

Recent work on vision-based methods for on-orbit relative navigation proposes using cameras and optical sensors for relative pose[4] estimation via image processing and computer vision techniques for clusters of small satellites such as CubeSats [65, 66]. Femto-spacecraft can be equipped with small COTS cameras, but the ADCS pointing accuracy required for relative state estimation with this approach, along with the limited computational and power resources available, makes such approaches presently impractical [37, 46].

Conventionally, range-based relative-navigation methods between cooperative spacecraft have been implemented as part of a chain of available resources in a satellite's sensor suite, used to accompany or back-up other more precise measurement techniques [67, 68]. With femto-spacecraft however, a large number of limited devices can be considered, using only range estimates for on-orbit relative navigation. This idea is discussed by Christian [69], who presents how range data could be applied for the initial relative orbit determination problem (IROD) of small satellite formations, with no a-priori information on the formation's state.

In this thesis, it is proposed that using range estimates from communication within a highly interconnected network would make it possible to calculate relative positions of swarm members directly. This would be of utility in the following circumstances:

1. Where there is an estimate of the a priori swarm state from the known ejection impulse and time from a deployer spacecraft and a model of the relative dynamics of the swarm is available. Processing ranging estimates to determine swarm relative positioning could be used to improve the swarm's state estimate over time and bound growing uncertainties.

[4] position and attitude

2. Where there is little or no understanding of the a priori swarm state in a given scenario. Relative positioning information would characterise the dispersal of the swarm and the spatial density of the swarm. This information could then be post-processed by the deployer spacecraft to enhance the utility of sensed data from the swarm.

As discussed, there are unique operational benefits for swarms that can be harnessed without significant complexity in individual swarm member operations. The approach taken in this thesis is to build on developments into range-based cooperative localisation methods that have found application across many disciplines, such as in mobile robotics, uncrewed aerial vehicle (UAV) applications, static WSNs, application of simultaneous localisation and mapping (SLAM) techniques, and navigation in GPS-denied environments [70–74].

In this thesis, it is proposed that range-based relative positioning methods may utilise the wireless communication link between swarm members to infer ranges, specifically using the received signal strength indication (RSSI) as a proxy for a direct range measurement between two femto-spacecraft using undirected antennas. A signal sent from a device transmitting at a known power can be converted to a range estimate with an understanding of the path loss between the two devices. This is an appealing solution for femto-spacecraft as it would not require additional sensors when resources are already constrained, is usable in essentially any orbit type, does not require attitude control, and is available by virtue of the swarm carrying out its primary mission application when communicating data between swarm members. Furthermore, this could find use within the smallest of femto-spacecraft design concepts as it only requires the ability to communicate data.

There are other existing methods for deriving range estimates from wireless communications, such as time difference of arrival techniques [75–77]. These approaches make use of synchronisation and timing processes to estimate the difference between the time a signal was sent at the transmitting device and the time a signal was received at the receiving device. This requires highly accurate synchronisation throughout the network, and introduces further complexity and feasibility issues with large networks of many devices. As a result, these approaches are not considered in this work.

Network structure and computation sharing would vary widely depending on the mission application, so this research is not limited to one particular approach. Experimental demonstration of relative positioning is necessary to show practical viability, to determine performance limitations and gain experimental insights. In this thesis, the practical limitations of using RSSI data as a range metric are addressed within the steps taken to convert this into a suitable ranging metric.

Chapter 3

Relative Positioning Algorithms

This chapter presents the development and implementation of relative positioning algorithms designed to use range estimates available from wireless communications within highly interconnected femto-spacecraft swarm networks. The challenge of localising an entire swarm of femto-spacecraft is firstly presented as the relative positioning problem, with discussion of the challenges associated with solving this problem in the context of relative navigation for femto-spacecraft. Then, algorithms are developed to work with centralised, decentralised and distributed computation of relative positioning. In all cases, however, it is assumed that the swarm network has either full or high interconnectivity. Finally, a series of test cases are simulated to demonstrate the algorithms operating with varying levels of inaccuracy in the range estimates available.

3.1 The Relative Positioning Problem

The position of any object is relative to the coordinate system that it is described in. For navigating in orbit around Earth, a global, Earth-centred coordinate system is normally used [78]. This can be referred to as absolute positioning. Relative positioning is the process of locating an object within a coordinate system that is not absolute. This is predominantly used in situations where it is difficult or even impossible to estimate absolute position given the sensor technology available, or where absolute position is not of importance or relevance (e.g. spacecraft rendezvous and docking).

In this thesis, the position of femto-spacecraft are described in a Cartesian frame relative to a larger spacecraft that deploys the swarm in space. Chapter 4 addresses the local relative reference frame used for relative navigation in LEO. For the purposes of describing the relative positioning problem, implementing algorithms to solve it, and testing these algorithms' capabilities, it is sufficient to assume that a relative Cartesian reference frame is fully defined with respect to the femto-spacecraft swarm. The primary challenge in this approach to relative positioning is to determine where individual members of a larger swarm are located using only estimates of ranges between one another.

3.1.1 Mathematical description

The relative positioning problem can be formulated mathematically for this purpose with the following assumptions. As the femto-spacecraft are moving through space, positions are estimated at discrete time steps. This is a valid approach provided the range estimates can be obtained via networking and positions can be computed in real-time with a sampling rate within the network that keeps up with the changing dynamics of the swarm. Secondly, as certain spacecraft, such as the deployer(s) of the swarm, would be much larger than individual femto-spacecraft, it is assumed that these spacecraft locations are known to the swarm. Similarly, in LEO a proportion of the femto-spacecraft within the swarm could be equipped with GPS receivers. For the scope of this work, these spacecraft are referred to as 'anchor spacecraft' of known position. At a minimum, a single deployer spacecraft would be the only known anchor spacecraft.

The parameters of this problem are therefore: n unknown femto-spacecraft position vectors $\mathbf{x}_i \in \mathbb{R}^3$, where $i = 1, ..., n$; m known anchor spacecraft positions vectors $\mathbf{a}_k \in \mathbb{R}^3$, where $k = 1, ..., m$; known range estimates between femto-spacecraft, where $\hat{r}_{ij}$ is the range estimate between femto-spacecraft positions $\mathbf{x}_i(x_i, y_i, z_i)$ and $\mathbf{x}_j(x_j, y_j, z_j)$, and $1 \leq i < j \leq n$; and known range estimates between femto-spacecraft and anchor spacecraft, where $\hat{r}_{ik}$ is the range estimate between femto-spacecraft $\mathbf{x}_i$ and anchor spacecraft $\mathbf{a}_k$, and $i \leq k \leq m$.

The objective of this problem is to find the n estimated femto-spacecraft positions that were previously unknown. Arranging the true femto-spacecraft position vectors into a $3 \times n$ matrix X, in the absence of measurement error, the objective is to find:

$$X = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \\ z_1 & z_2 & \dots & z_n \end{bmatrix} \tag{3.1}$$

subject to:

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = r_{ij}^2 \quad \forall\, (i,j) \in N_1 \tag{3.2}$$

$$\|\mathbf{x}_i - \mathbf{a}_k\|^2 = r_{ik}^2 \quad \forall\, (i,k) \in N_2 \tag{3.3}$$

where the set N_1 contains the pairs of femto-spacecraft (i, j) that have a true range r_{ij} between them, and the set N_2 contains the pairs of femto-spacecraft i and anchor k that have a true range r_{ik} between them. This scenario is shown in Fig. 3.1.

3.1.2 Solution approach considerations

As the exact ranges r_{ij} and r_{ik} are in practice unknown, any algorithm used to solve the relative positioning problem has the objective of minimising the difference between true and estimated positions obtained from the range estimates $\hat{r}_{ij}$ and $\hat{r}_{ik}$. In this context, the approach must

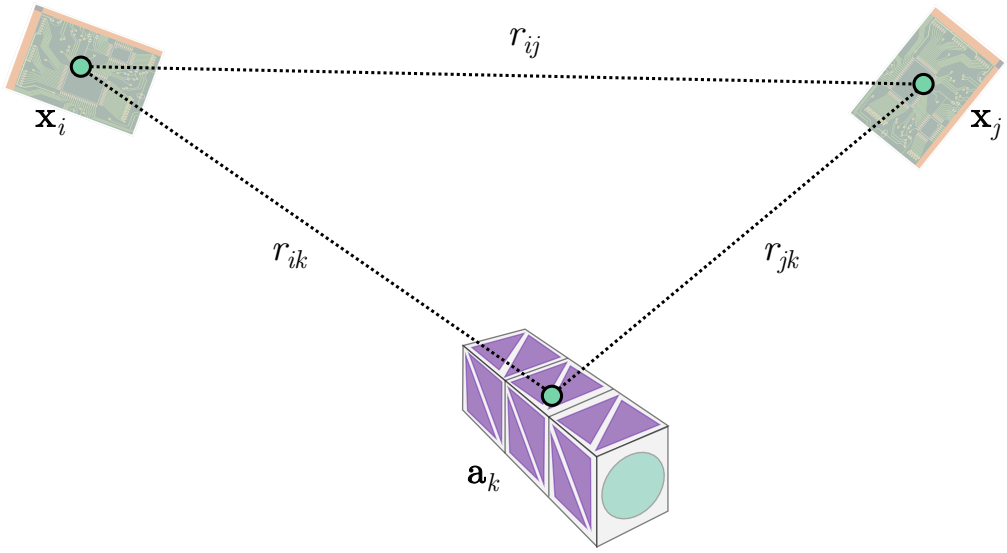


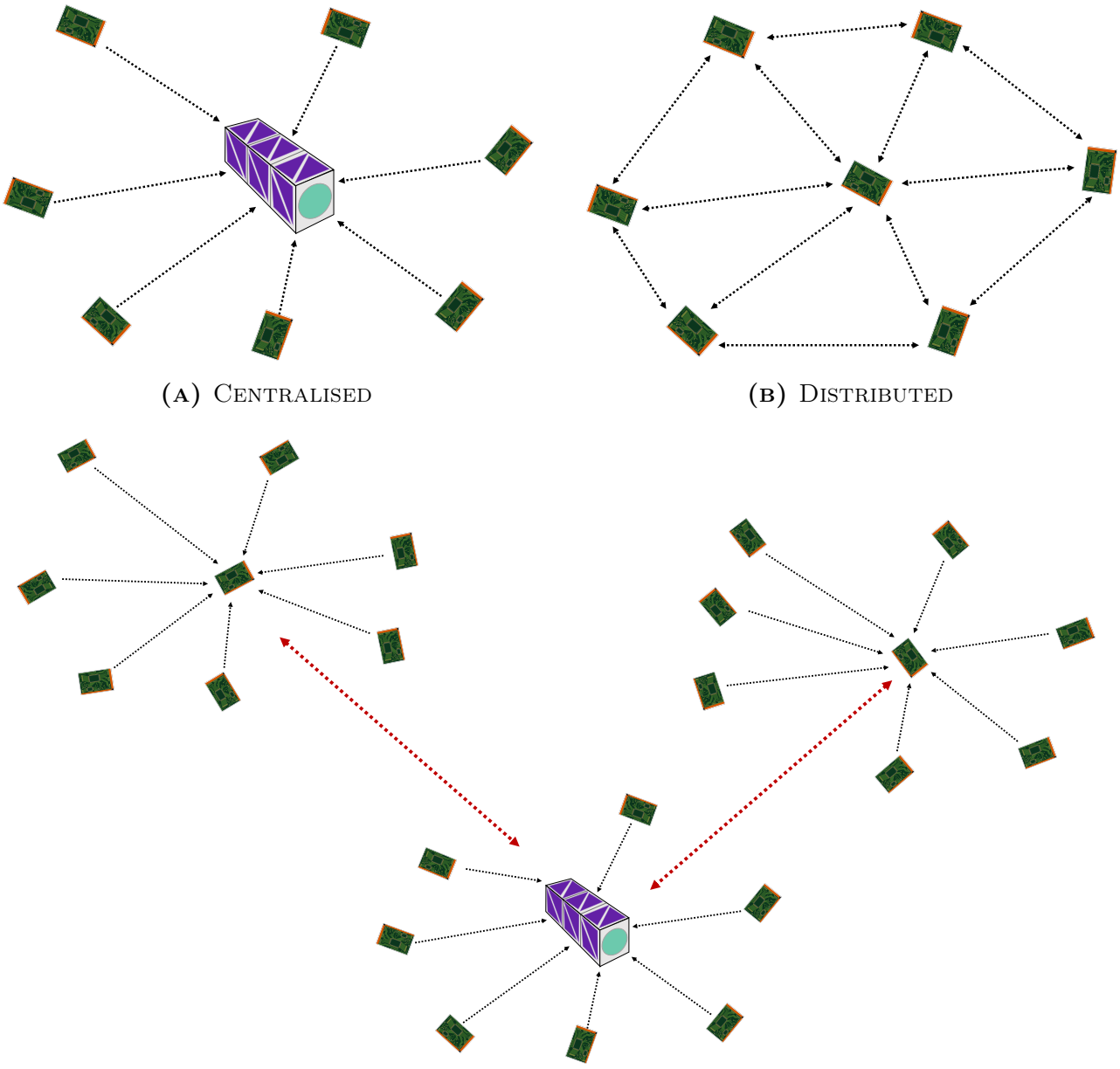
Figure 3.1: The relative positioning problem

also be able to operate in real-time, that is, be able to update position estimates in discrete time steps faster than the changing dynamics of the femto-spacecraft swarm. Finally, the approach must also be scalable to large networks of potentially hundreds of inter-connected femto-spacecraft [79, 80].

Different algorithmic approaches to solving this problem are therefore necessitated by scalability in terms of computation to run in real-time, but also network communication range [81]. If the swarm spatial structure extends beyond the communications range limit of single femto-spacecraft, there would be a need for distributed communication and computation throughout the network to enable relative positioning and navigation for the entire swarm. With this considered, configurations within a network can be broadly expressed in the three primary categories of centralised, distributed, and decentralised computation [82], as shown in Fig. 3.2.

The direction of the arrowheads indicate communication between spacecraft for the purposes of relative positioning computation (primarily, sending range estimates wirelessly to be processed). In a centralised configuration (Fig. 3.2a), femto-spacecraft send ranging data back to a central spacecraft (e.g. the swarm deployer) that handles all computation and optimises relative position estimates for the entire swarm. In a distributed configuration (Fig. 3.2b), this computation is shared; the femto-spacecraft share ranging information and attempt to localise relative to one another. In a decentralised configuration (Fig. 3.2c), distributed 'cluster heads' may act in a centralised way with nearby femto-spacecraft, but relate to other cluster heads in a distributed fashion. Different strategies would suit various in-space applications and swarm sizes, both in terms of the number of femto-spacecraft and the length-scales they are deployed over. Developing different algorithms which may be more suitable depending on the utility, scale, and application of the swarm, is therefore the approach taken in this work.

It is important to note the distinction that is made here between centralisation in computation and communication. Centralised computation does not mean that the swarm cannot or does



(a) Centralised

(b) Distributed

(c) Decentralised

Figure 3.2: Swarm relative positioning computation configurations

not have a highly interconnected network topology for communications. As relative navigation for femto-spacecraft swarms is primarily a key enabling technology for distributed sensing applications, it is assumed that swarms will in most cases operate with high degrees of interconnectivity between each femto-spacecraft. However, the extent of distributed computation for the purposes of navigation or sensing is presumed to vary considerably depending on the swarm size and application. Centralised computation would place less constraints on individual swarm members and be simpler to implement, while distributed computation offers operational benefits for sensing applications and mitigating the effect of component degradation.

Therefore, the relative positioning algorithms developed in this chapter are designed to be implemented in different ways. Section 3.2 presents an algorithm using convex optimisation that

relies on a central unit for computation. This could also be implemented in a decentralised manner with clusters as described here. Section 3.3 presents a distributed algorithm designed to rely on shared computation within the swarm using non-linear least squares (NLLS) trilateration. Section 3.4 presents an algorithm using trilateration via normal probability density functions, intended for use in small swarms to make best use of less accurate range estimates and the significantly fewer constraints available. As has been noted in the author's previous work [5], these approaches may be adapted for terrestrial applications within wireless sensor networks (WSNs) that share the severe resource constraints of femto-spacecraft and in environments where GPS is unavailable.

3.2 Semidefinite Programming

With a centralised spacecraft able to process the entire swarm's range data centrally, the relative positioning problem becomes an optimisation challenge: minimise the error in all position estimates with the constraints available. By relaxing the relative positioning problem constraints in Eqs. 3.2-3.3 to satisfy convex optimisation bounds, geometric constraints between femto-spacecraft can be represented by linear matrix inequalities (LMIs) combined to form a single semidefinite program (SDP) [83]. In an SDP, a linear function is minimised subject to the LMI constraint that a linear combination of symmetric matrices is positive semidefinite. By definition, if Z is a symmetric matrix, and $Q(\mathbf{x}) = \mathbf{x}^T Z \mathbf{x}$ is the corresponding quadratic form, Z and Q are positive semidefinite if $Q(\mathbf{x}) \geq 0$ for all $\mathbf{x}$ [84]. This has been applied for localisation in static two-dimensional (2D) WSNs in [85–88].

The feasible regions of SDPs are spectrahedra, and this requires the constraints to be convex functions. The relative positioning problem can be reformulated to find the symmetric positive semidefinite matrix Z containing the matrix X of all unknown femto-spacecraft positions. Using the definition of the relative positioning problem from Eqs. 3.1-3.3, the terms can be restated in matrix form in a way that sets the problem up to be solved using convex optimisation techniques [87]. Introducing the terms $e_{ij} \in \mathbb{R}^n$ as a zero column vector with the value of 1 at index i and the value of -1 at index j, $e_i \in \mathbb{R}^n$ as a zero column vector with the value of 1 at index i, and the symmetric matrix $Y = X^T X \in \mathbb{R}^{n \times n}$, Eqs. 3.2-3.3 become:

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = e_{ij}^T Y e_{ij} \quad \forall\,(i,j) \in N_1 \tag{3.4}$$

$$\|\mathbf{x}_i - \mathbf{a}_k\|^2 = \begin{bmatrix} e_i \\ a_k \end{bmatrix}^T \begin{bmatrix} I_3 & X \end{bmatrix}^T \begin{bmatrix} I_3 & X \end{bmatrix} \begin{bmatrix} e_i \\ a_k \end{bmatrix} \quad \forall\,(i,k) \in N_2 \tag{3.5}$$

where I_3 is a 3×3 identity matrix. Now the relative positioning problem can be restated to find the matrix of unknown relative positions X such that:

$$e_{ij}^T Y e_{ij} = r_{ij}^2 \quad \forall\,(i,j) \in N_1 \tag{3.6}$$

$$\begin{bmatrix} e_i \\ a_k \end{bmatrix}^T \begin{bmatrix} I_3 & X \end{bmatrix}^T \begin{bmatrix} I_3 & X \end{bmatrix} \begin{bmatrix} e_i \\ a_k \end{bmatrix} = r_{ik}^2 \quad \forall\,(i,k) \in N_2 \tag{3.7}$$

$$Y = X^T X \tag{3.8}$$

By relaxing the constraint $Y = X^T X$ to $Y \succcurlyeq X^T X$, this problem becomes an SDP [83, 89], where the inequality constraint '$\succcurlyeq$' indicates that Y is positive semidefinite. The positive semidefinite symmetric matrix Z can be defined as:[1]

$$Z = \begin{bmatrix} I_3 & X \end{bmatrix}^T \begin{bmatrix} I_3 & X \end{bmatrix} = \begin{bmatrix} I_3 & X \\ X^T & Y \end{bmatrix} \tag{3.9}$$

Semidefinite programs, as a generalisation of linear programs, only allow linear objective functions. In this case, there is no obvious candidate for a linear objective that would optimise this problem [85]. Without an objective function, this could be represented as a feasibility problem, effectively with the objective of finding any feasible solution to the relative positioning problem without considering the objective of minimising positioning error. The effect of this would be to solve for random feasible positions bounded by the feasible regions, with an empty objective function in the convex optimisation solver when computed. As the algorithm will in practice be calculating the position matrix subject to noisy range estimates, a straightforward way to solve the SDP is to minimise the sum of squared positioning errors ε^2 [86], i.e. to minimise:

$$\varepsilon^2 = \sum_{(i,j)\in N_1} \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{r}_{ij}^2 \right| + \sum_{(i,k)\in N_2} \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{r}_{ik}^2 \right| \tag{3.10}$$

where the absolute values of these errors are taken into account for range estimates being either higher or lower than the true range. This can be addressed by defining the slack variables α_{ij} and α_{ik}:

$$\alpha_{ij} = \alpha_{ij}^+ + \alpha_{ij}^- = \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{r}_{ij}^2 \right| \tag{3.11}$$

$$\alpha_{ik} = \alpha_{ik}^+ + \alpha_{ik}^- = \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{r}_{ik}^2 \right| \tag{3.12}$$

In addition to a range estimate $\hat{r}$, upper and lower constraints $\hat{\bar{r}}$ and $\hat{\underline{r}}$ can be placed based on either or a combination of: the confidence in the range estimate around its mean, the minimum and maximum possible ranges that the estimate could correspond to (based on the magnitude and direction of the deployment impulse, for example) and the maximum communication range of the radio. For RSSI data, as is used experimentally in this thesis, these additional constraints can reflect the inaccuracy of individual range estimates from modelling. This can help alleviate the inaccuracy of RSSI, at worst having no impact on positioning accuracy but offering the potential for improvement over the consideration of $\hat{r}$ alone [86, 89]. Again, upper and lower range estimates could be constrained by the maximum and minimum possible ranges between femto-spacecraft (which would vary for different scenarios considered). For example,

[1] As X is also a matrix and ${I_3}^T = I_3$, $[I_3 \;\; X]^T = \begin{bmatrix} I_3 \\ X^T \end{bmatrix}$

an upper bound could be placed based on maximum possible radio communications range, or a lower bound on the minimum possible distance femto-spacecraft could be apart from one another based on a particular deployment sequence. Adding these four additional constraints to minimise:

$$\begin{aligned}\varepsilon^2 = &\sum_{(i,j)\in N_1} \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{\bar{r}}_{ij}^2 \right| + \sum_{(i,k)\in N_2} \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{\bar{r}}_{ik}^2 \right| \\ &+ \sum_{(i,j)\in N_1} \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{\underline{r}}_{ij}^2 \right| + \sum_{(i,k)\in N_2} \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{\underline{r}}_{ik}^2 \right|\end{aligned} \tag{3.13}$$

and introducing the corresponding slack variables β_{ij} and β_{ik}:

$$\beta_{ij}^{+} = \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{\bar{r}}_{ij}^2 \right| \tag{3.14}$$

$$\beta_{ij}^{-} = \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{\underline{r}}_{ij}^2 \right| \tag{3.15}$$

$$\beta_{ik}^{+} = \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{\bar{r}}_{ik}^2 \right| \tag{3.16}$$

$$\beta_{ik}^{-} = \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{\underline{r}}_{ik}^2 \right| \tag{3.17}$$

This problem can now be expressed as an SDP optimisation problem:

Find:

$$Z = \begin{bmatrix} I_3 & X^T \\ X & X^T X \end{bmatrix} \tag{3.18}$$

to minimise:

$$\varepsilon^2 = \sum_{(i,j)\in N_1} (\alpha_{ij}^{+} + \alpha_{ij}^{-}) + \sum_{(i,k)\in N_2} (\alpha_{ik}^{+} + \alpha_{ik}^{-}) + \sum_{(i,j)\in N_1} \beta_{ij}^{+} + \sum_{(i,k)\in N_2} \beta_{ik}^{+} + \sum_{(i,j)\in N_1} \beta_{ij}^{-} + \sum_{(i,k)\in N_2} \beta_{ik}^{-} \tag{3.19}$$

subject to:

$$\begin{bmatrix} e_{ij} \\ 0 \end{bmatrix}^T Z \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix} - \alpha_{ij}^{+} + \alpha_{ij}^{-} = \hat{r}_{ij}^2 \quad \forall\, (i,j) \in N_1 \tag{3.20}$$

$$\begin{bmatrix} e_i \\ a_k \end{bmatrix}^T Z \begin{bmatrix} e_i \\ a_k \end{bmatrix} - \alpha_{ik}^{+} + \alpha_{ik}^{-} = \hat{r}_{ik}^2 \quad \forall\, (i,k) \in N_2 \tag{3.21}$$

$$\begin{bmatrix} e_{ij} \\ 0 \end{bmatrix}^T Z \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix} - \beta_{ij}^{+} \leq \hat{\bar{r}}_{ij}^2 \quad \forall\, (i,j) \in N_1 \tag{3.22}$$

$$\begin{bmatrix} e_{ij} \\ 0 \end{bmatrix}^T Z \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix} + \beta_{ij}^{-} \geq \hat{\underline{r}}_{ij}^2 \quad \forall\, (i,j) \in N_1 \tag{3.23}$$

$$\begin{bmatrix} e_i \\ a_k \end{bmatrix}^T Z \begin{bmatrix} e_i \\ a_k \end{bmatrix} - \beta_{ik}^{+} \leq \hat{\bar{r}}_{ik}^2 \quad \forall\, (i,k) \in N_2 \tag{3.24}$$

$$\begin{bmatrix} e_i \\ a_k \end{bmatrix}^T Z \begin{bmatrix} e_i \\ a_k \end{bmatrix} + \beta_{ik}^- \geq \underline{\hat{r}}_{ik}^2 \quad \forall\, (i,k) \in N_2 \tag{3.25}$$

where:

$$Z, \alpha_{ij}^+, \alpha_{ij}^-, \alpha_{ik}^+, \alpha_{ik}^-, \beta_{ij}^+, \beta_{ij}^-, \beta_{ik}^+, \beta_{ik}^- \succcurlyeq 0 \tag{3.26}$$

Once solved, X can be extracted from Z and be further refined with a gradient descent method [90]. The basis of this is to minimise the difference between the range estimates and the ranges that can be found from the current position estimates, i.e. to minimise (now without the slack variables as the position estimates are known):

$$f(X) \triangleq \sum_{(i,j)\in N_1} \left| \|\mathbf{x}_i - \mathbf{x}_j\|^2 - \hat{r}_{ij}^2 \right| + \sum_{(i,k)\in N_2} \left| \|\mathbf{x}_i - \mathbf{a}_k\|^2 - \hat{r}_{ik}^2 \right| \tag{3.27}$$

by refining each position estimate in the negative gradient direction of the objective $f(X)$. Note that the upper and lower bound constraints on $\hat{r}$ can also be used in $f(X)$ but for brevity are not shown. The gradient of $f(X)$ with respect to femto-spacecraft i is given by:[2]

$$\nabla_i f(X) = \sum_{(i,j)\in N_1} \left(1 - \frac{\hat{r}_{ij}}{\|\mathbf{x}_i - \mathbf{x}_j\|}\right)(\mathbf{x}_i - \mathbf{x}_j) + \sum_{(i,k)\in N_2} \left(1 - \frac{\hat{r}_{ik}}{\|\mathbf{x}_i - \mathbf{a}_k\|}\right)(\mathbf{x}_i - \mathbf{a}_k) \tag{3.28}$$

Then the position matrix X can be updated as:

$$X(\gamma) = [\mathbf{x}_1 - \gamma \nabla_1 f(X), ..., \mathbf{x}_n - \gamma \nabla_n f(X)] \tag{3.29}$$

This can be iterated starting with $\gamma = 1$, whereby if $f(X(\gamma)) < f(X)$, the next iteration becomes $X(\gamma)$, otherwise γ is reduced in steps towards zero and the last X is used for the next iteration, until convergence within a given tolerance level between two iterations (or the lower limit of zero on γ is reached) [90]. The SDP optimisation output is taken as the first iteration for this gradient descent method, which is an optional refinement step in the implementation. Adding this refinement adds to the computation time but will in general improve accuracy. Therefore, a trade off would exist for practical implementation.

This formulation is used as the basis for implementing a three-dimensional (3D) SDP algorithm, developed in MATLAB with the convex optimisation modelling system CVX and using the solver SDPT3 [91]. This approach is necessarily centralised because it would require one device to optimise for the entire swarm given all the range estimates and problem constraints. Such computation could be handled by the swarm deployer, such as a CubeSat carrier. In this role, the central unit would accumulate swarm range estimates communicated to it via networking and use this information to form the SDP constraints, optimise for the entire swarm and then estimate the relative positions. In practice, as the space-based WSN that the femto-spacecraft form would be dynamic, gradually drifting away from the deployer, the algorithm

[2] as $\nabla_i \|\mathbf{x}_i - \mathbf{x}_j\| = \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|} \quad \forall i \neq j$

would therefore be sampled at regular intervals to update the overall swarm state estimates dynamically using a relative navigation filter, as described later in Chapter 4.

The frequency with which this optimisation is performed would be a design choice that depends upon the spatial density, relative motion and number of femto-spacecraft in the swarm. For the purposes of centralised operations, a key assumption is that the time taken to accumulate range data and calculate relative positions is sufficiently low enough to neglect time differences in range data obtained and to keep up with the dynamics of the swarm. This is discussed in more detail in Chapter 5, where the time taken to accumulate data via networking is addressed.

Algorithm 3.1 presents the implementation as pseudocode, while the flowchart in Fig. 3.3 summarises the straightforward centralised algorithm logic flow. The flowchart displays a decision case for a situation where anchors are unavailable. As will be discussed in Section 3.3, five femto-spacecraft can be localised in an arbitrary reference frame relative to one another to unambiguously describe their relative positions in three-dimensional space up to a global transformation (translation and rotation). This can provide the algorithm with a relative coordinate system to solve within and a set of proxy anchor positions if anchors are unavailable. Otherwise, anchor positions would be provided using a small proportion of the swarm equipped with GPS receivers, or communications to other spacecraft (such as in the case of multiple deployers) of known position. From the description of the NLLS algorithm techniques, the combination of techniques to enable relative frame definition to solve SDP is a straightforward extension to the algorithm presented here. This is not however implemented in simulation or tested, as it is not the intended use of this approach.

Algorithm 3.1 SDP algorithm

Inputs: r_{ij}, r_{ik}, **a**
Output: X

Define SDP constraints
Solve SDP problem:
if *convergence* **then**
    extract X from Z
else
    iterate
end if
Set $\gamma = 1$ and refine X with gradient descent method:
while $\gamma > 0$ **do**
    if $f(X(\gamma) < f(X)$ **then**
        Set $X = X(\gamma)$
        if *convergence* **then**
            extract X as final position estimates
        else
            iterate through $\gamma_{k+1} \leftarrow \gamma_k$
        end if
    end if
end while

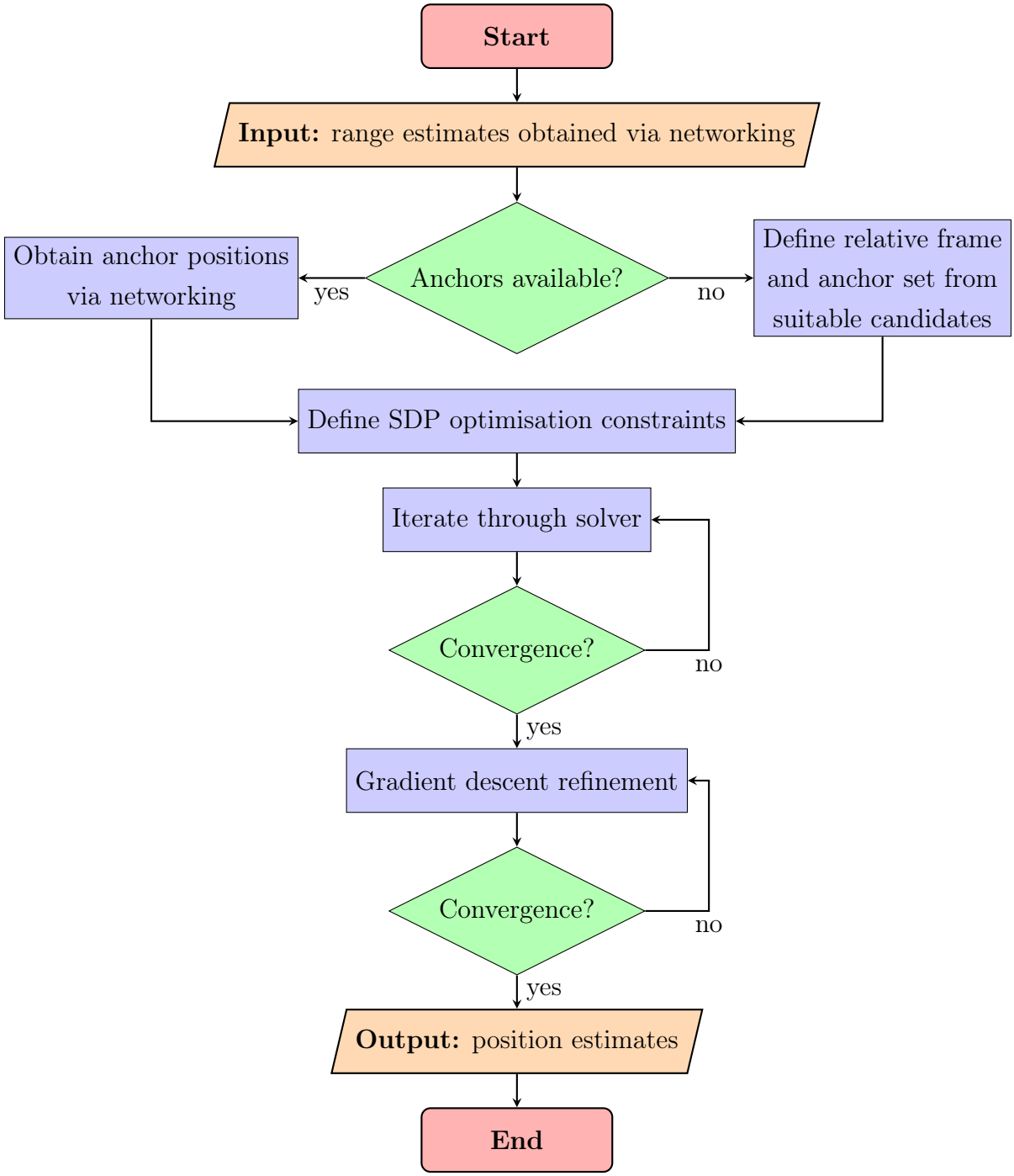


Figure 3.3: SDP algorithm flowchart

3.3 Non-linear Least Squares Distributed Trilateration

A centralised approach relies on one unit being supplied all information necessary to optimise the estimate of the relative positions for the swarm members. A decentralised or distributed approach instead relies on data and computation sharing between swarm members. Using a distributed positioning algorithm, there are far fewer constraints available to individual units, requiring a completely different approach. With this algorithm, non-linear least squares (NLLS) distributed trilateration is used to determine the relative positions of femto-spacecraft with respect to their closest neighbours in the network.

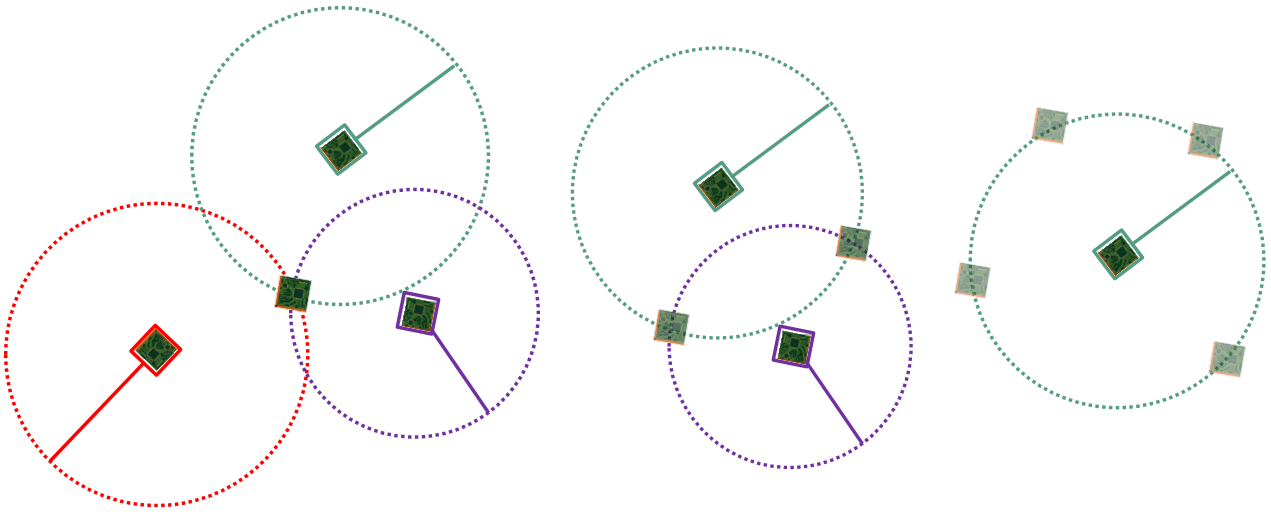
The implementation challenge for this strategy is developing an algorithm that is robust to measurement noise and the consequent ambiguities in relative positioning that can otherwise arise. The key advantage of this method however is its inherent scalability and ability to operate in an anchorless way. This is because local clusters of femto-spacecraft would be able to determine their positions relative to one another in a coordinate system local to them, which can be mapped onto other groups or anchors at a later stage. Crucially though, when this mapping takes place, the computation of the relative spatial structure has already been done, saving later computation for other swarm members. The structure of the algorithm in computational implementation and in communication within the network could vary considerably depending on the application or swarm size. In the simplest case, a single femto-spacecraft would be able to determine its position relative to at least four of its neighbours.

The algorithm developed in this section is designed to be scalable for distributed computation using varying extents of shared computation between femto-spacecraft. Like the SDP algorithm, the intention is to be as general as possible and not restrict implementation to specific use cases or the computational resources of individual spacecraft. Unlike the SDP algorithm, which with current technology would not be feasible to compute in real-time on femto-spacecraft, the algorithm developed here could be used in networking the results of simple embedded computation between swarm members.

3.3.1 Relative ambiguities

To develop this algorithm, it is first important to consider the ambiguities than can arise when a small number of femto-spacecraft attempt to localise themselves relative to one another in a shared fashion. This is easier to describe firstly in two-dimensional (2D) space, where in the absence of any measurement errors, knowledge of the ranges between three femto-spacecraft of known position would be sufficient to localise a fourth unknown object precisely and uniquely, as shown in Fig. 3.4a.

The unknown femto-spacecraft's position is found to be the only common intersection of the three circles that bind the known ranges of other femto-spacecraft to it. If there were only two femto-spacecraft of known position, the subsequent two intersections of the range circles would result in two possible positions for the unknown femto-spacecraft, causing an obvious



(a) One unique intersection (b) Two intersections (c) Infinite positions

Figure 3.4: Range-based trilateration in 2D

ambiguity as to which position is correct (Fig. 3.4b). If there was only one femto-spacecraft of known position, the unknown femto-spacecraft could lie anywhere on the circumference of the range circle (Fig. 3.4c). Extending this principle to three-dimensional (3D) space, the ranges between four non-coplanar femto-spacecraft of known position, again in the absence of any measurement errors, is sufficient to uniquely localise a fifth femto-spacecraft of unknown position, as shown in Fig. 3.5.

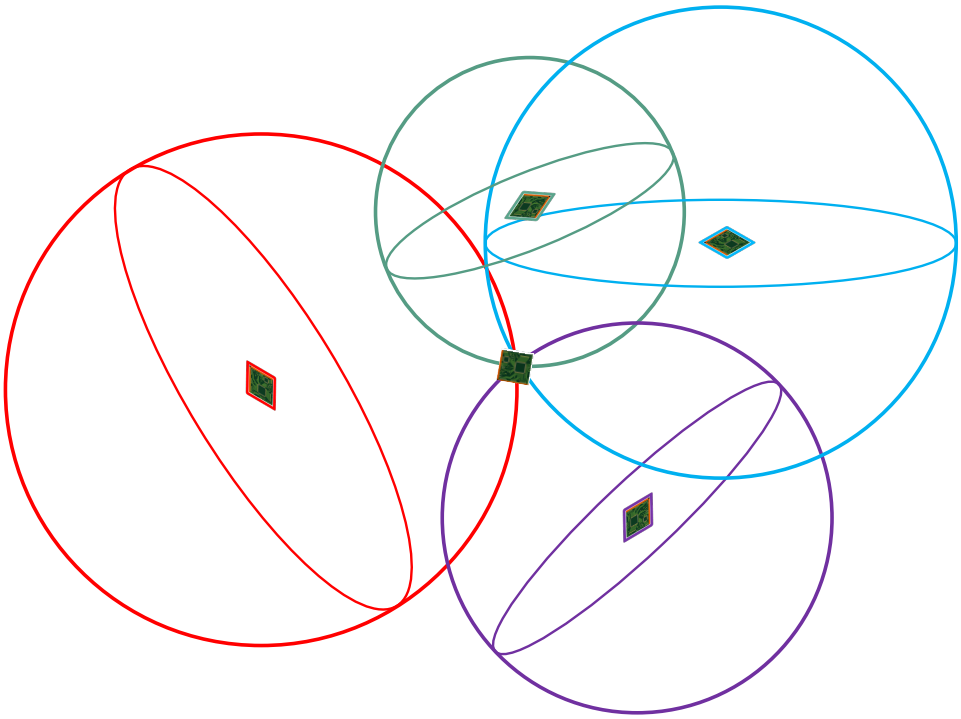


Figure 3.5: Range-based trilateration in 3D

In practice, any error in the range estimates requires an approximation technique to estimate position. This is a simplified description of range-based trilateration, the fundamental principle on which the global positioning system (GPS) operates. Any smartphone user can determine

their location anywhere on the globe to within several metres with a GPS receiver. The GPS receiver is able to determine the time difference between a GPS satellite's broadcast and reception to determine the range between the receiver and the satellite. This is possible because the satellite has an atomic clock that can estimate the current time extremely accurately. As described above for 3D trilateration, with a connection to at least four GPS satellites, a user can estimate their current location anywhere on Earth.

For relative navigation, without a global coordinate system, there is a need to define a relative coordinate system that is common to all femto-spacecraft to be localised within that system. In this thesis, a Cartesian frame is always used. Again, it is more straightforward to describe and visualise how this can be done and the challenges that arise firstly in 2D space. A 2D Cartesian coordinate system can be fully defined in relative space with as few as three femto-spacecraft with known range estimates between one another. Assigning the position of one femto-spacecraft at the origin, another a distance along the x-axis, as defined by the range between itself and the origin, and a third somewhere on the xy-plane from basic trigonometry, this relative coordinate system can be defined.

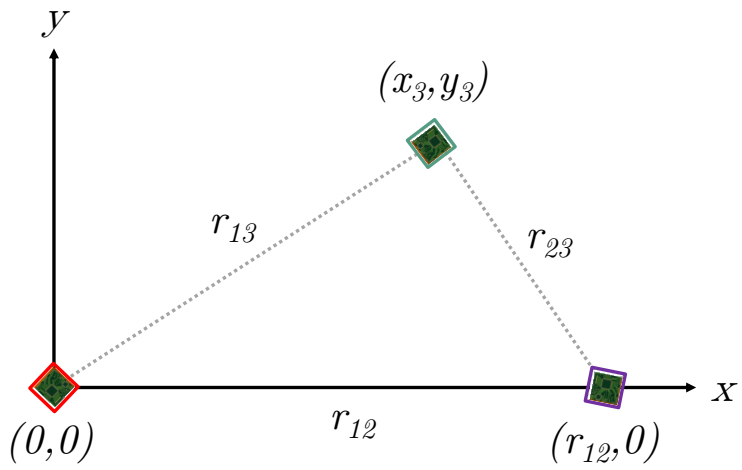


Figure 3.6: Defining a 2D Coordinate System

Now attempting to localise many unknown femto-spacecraft positions within this coordinate system using range measurements alone requires knowledge of what relationships between femto-spacecraft are sufficient to remove ambiguities in their relative positions. Figure 3.7 shows three femto-spacecraft that can communicate with one another, storing and sharing range data (nested circles). Each is therefore able to determine where it is relative to the other two, forming a relative triangle from the ranges.

Assuming that this is possible, and it will be seen that this is not always the case due to measurement noise, this is still insufficient in itself for constructing a network of further relative positions, owing to the positioning ambiguities that would arise, as illustrated in Fig. 3.8. Even with a shared coordinate system, this triangle can be located anywhere and rotated in any direction in 2D space, known as translational and rotational ambiguity. This is true of any relative coordinate system, so all that matters is that the femto-spacecraft localised within

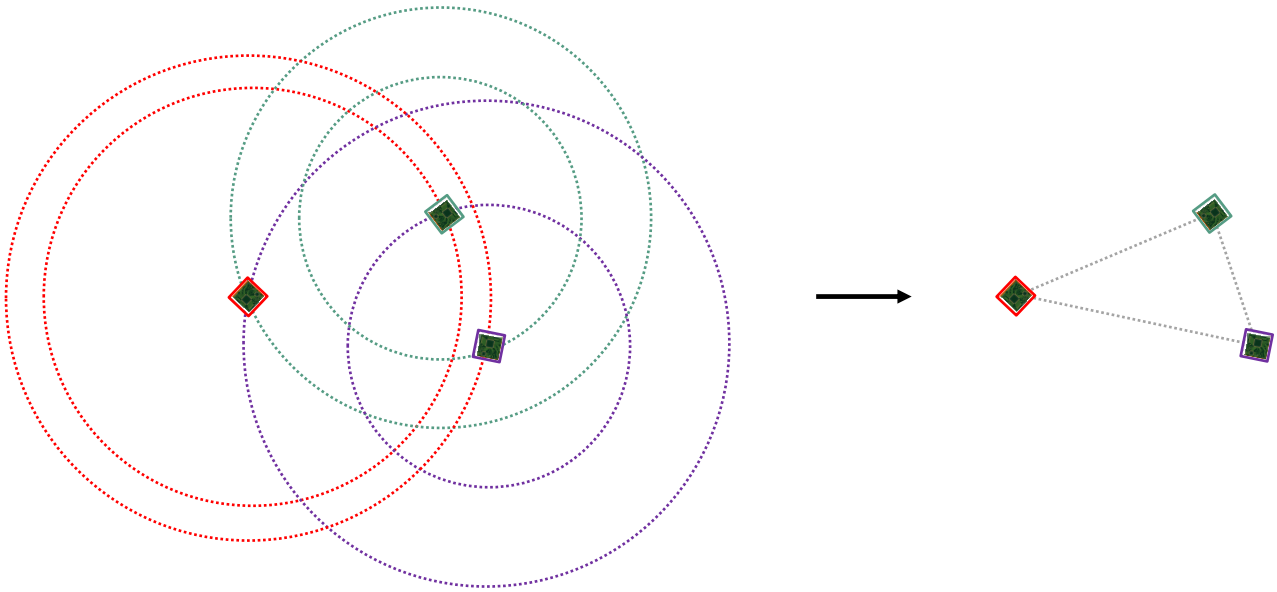
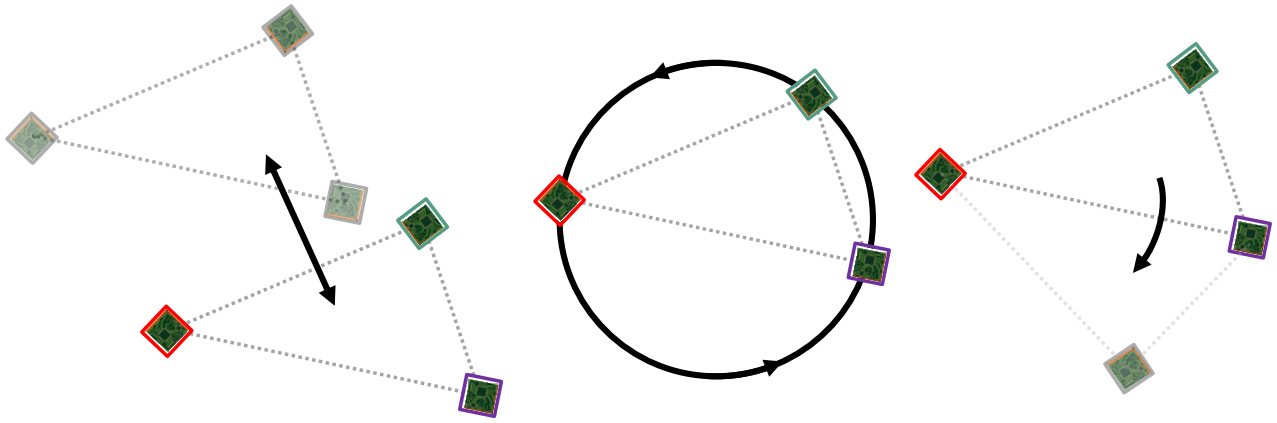


Figure 3.7: Relative positioning triangle formed via communication

it can be unambiguously described up to a single translation, rotation and reflection (TRR) common to all positions. Reflective ambiguity exists if the solution can be 'flipped' about any side and still remain valid - where there is an equally valid 'ghost' solution. As described above, the initialisation of the relative frame by defining those three femto-spacecraft positions prevents this initially, but as more femto-spacecraft are localised into the frame, knowledge of the ranges between just three nodes in 2D space is insufficient to prevent reflective ambiguity in the relative frame.

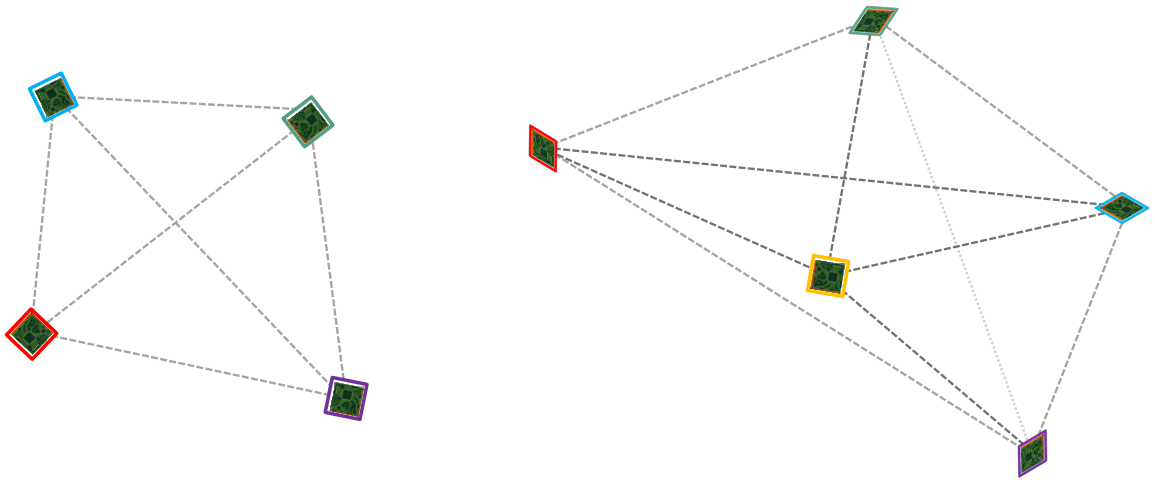


(a) Translation (b) Rotation (c) Reflection

Figure 3.8: Translational, rotational and reflective ambiguities in 2D

Clearly, triangles cannot be used as a linking mechanism for constructing a larger network of connections in two-dimensional space or reflective ambiguities will arise. A rigid relative solution must be unique for the entire swarm up to a single translation, rotation, and reflection (TRR). In two-dimensional space, quadrilaterals formed with the maximum six ranges between four femto-spacecraft are globally rigid and can be used as a linking mechanism [92, 93]. The three-

dimensional equivalents of the triangle ambiguities are found with tetrahedra, and the rigid linking mechanism is found to require the maximum ten ranges between five femto-spacecraft. This would often take the form of a 6-sided triangular bipyramid, with the tenth range link through the middle of the structure, as shown in Fig. 3.9. Alternatively, this structure would be some form of 5-sided polygon if the two tetrahedra forming it were aligned in the same way above the xy-plane.



(a) Quadrilateral (2D) (b) Triangular bipyramid (3D)

Figure 3.9: Rigid relative linking mechanisms using ranges

This fully connected linking condition removes the possibility for reflection ambiguity when localising. When contained within the boundaries of a larger network and shared coordinate system (solved in translation and rotation), this is a robust way to localise further femto-spacecraft onto algorithmically. Relative coordinate system initialisation in 3D space therefore requires a minimum of five non-coplanar femto-spacecraft with all ten range estimates between them to be known to fully define the three axes unambiguously. A relative coordinate system can be mapped to other coordinate systems (including a global one) as long as some nodes are known in both coordinate systems. With these ambiguities addressed, the distributed algorithm is now developed.

3.3.2 NLLS algorithm

Provided that knowledge of the range estimates between four other femto-spacecraft that are not co-planar is in principle sufficient to uniquely identify a fifth femto-spacecraft as the only possible intersection, as shown in Fig. 3.10, trilateration is possible in a frame relative to these five femto-spacecraft.

The fifth femto-spacecraft position $\mathbf{x}_5$ is shown as the only possible intersection of the other four ranging spheres. A least-squares approach (or similar) is required to estimate position with this method in practice due to measurement noise. In its usual implementation, trilateration also requires the absolute coordinates of the four points to be known. With a relative approach, however, this concept is reversed to determine relative positions where there are many unknown

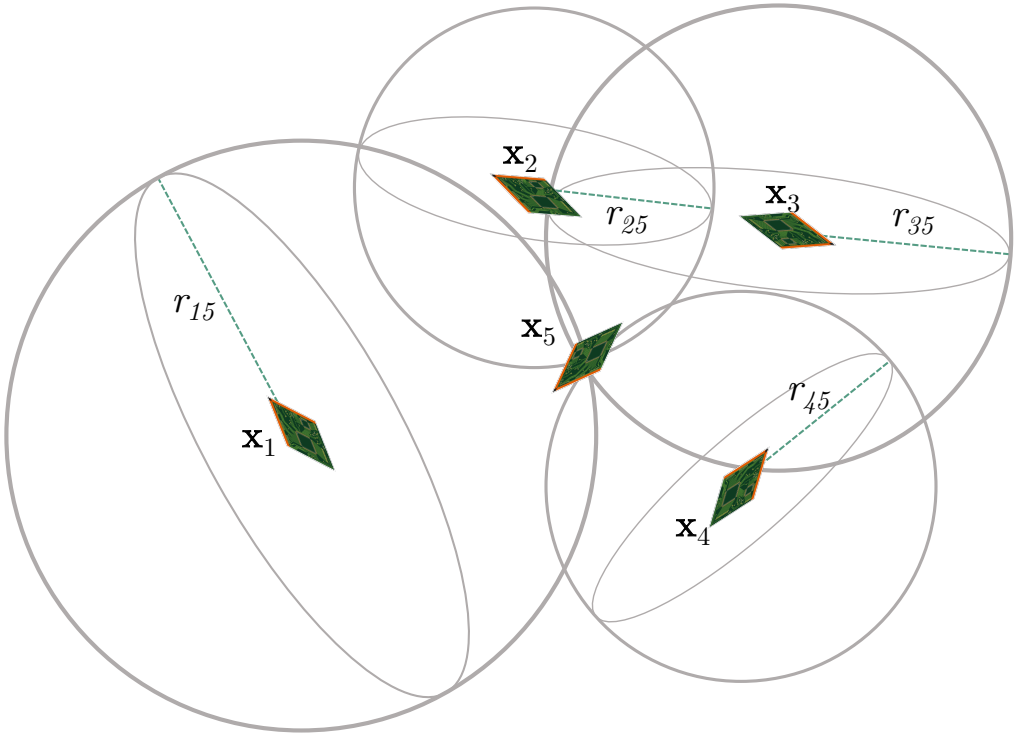
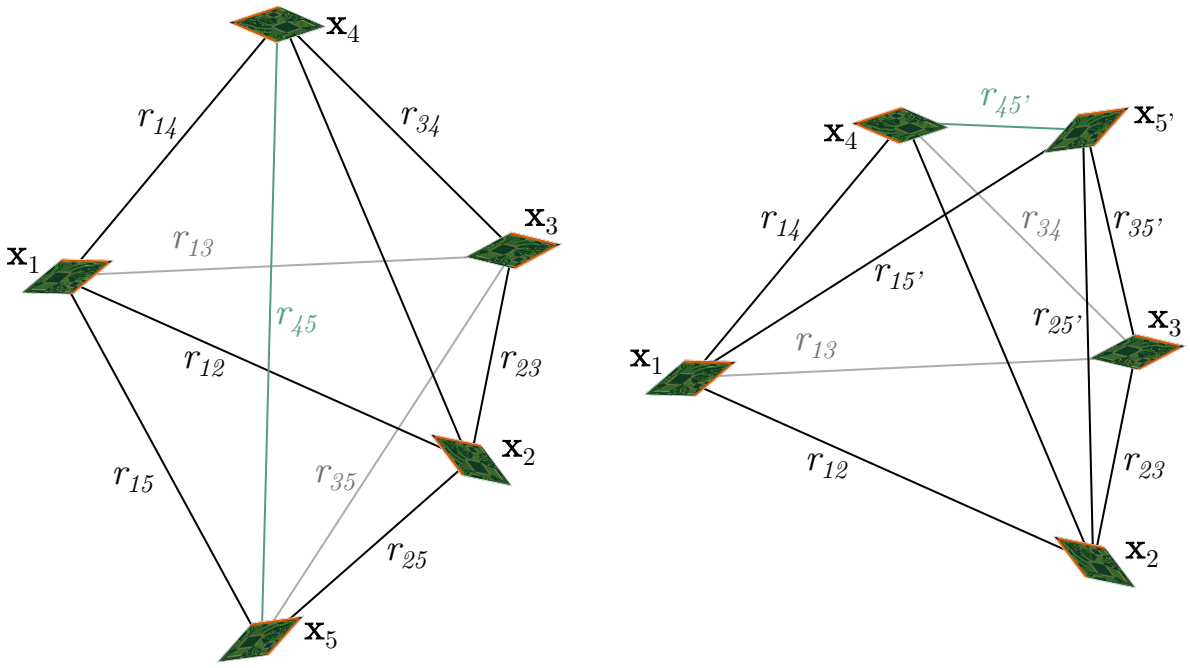


Figure 3.10: 4 ranging spheres to 5th femto-spacecraft

points but there are estimates of the ranges between them. Provided that these five femto-spacecraft can localise in the presence of measurement error, reflective ambiguity may still exist. It is, as discussed, essential to avoid this ambiguity in constructing a network of further relative positions. Consider the scenario shown in Fig. 3.11.



(a) Localisation (b) Alternate localisation

Figure 3.11: Reflective ambiguity between five femto-spacecraft

From Fig. 3.11a, consider the scenario where positions $\mathbf{x}_1$ to $\mathbf{x}_4$ are known in relative space, and that each range measurement to position $\mathbf{x}_5$ is used to trilaterate and locate femto-spacecraft 5 relative to the first four positions. It is possible that in the presence of measurement noise both the solutions shown in Figs. 3.11a and 3.11b are valid without considering the range r_{45} between $\mathbf{x}_4$ and $\mathbf{x}_5$. Even when r_{45} is considered, it is possible that the difference between r_{45} and the potential $r_{45'}$ is not sufficient in the presence of noise to rule out one solution or the other. While Fig. 3.11 illustrates an extreme case where the range estimate r_{45} would need to be completely anomalous to mistake the correct configuration (as the difference is so large), this is not always the case for random geometries, particularly in the presence of noise. In developing a distributed algorithm that uses relative trilateration it is essential to be robust against this kind of uncertainty and rule out candidate solutions that exhibit it. This is especially important for the first set of positions used to start the algorithm and trilaterate new solutions to that first solution.

Consider Fig. 3.11a for an illustration of how relative trilateration would work as the basis of starting the algorithm. A femto-spacecraft can be arbitrarily assigned at position $\mathbf{x}_1$ as the origin of a new relative Cartesian coordinate system, position $\mathbf{x}_2$ with the x-coordinate r_{12} and positions $\mathbf{x}_3$ and $\mathbf{x}_4$ using basic trigonometry (forming a relative tetrahedron of four femto-spacecraft). This is described later (see Eq. (3.39)). It is then essential to perform the same operation with positions $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_5$, and then the relationship between both tetrahedra can be confirmed using the range measurement r_{45}. With this structure formed subject to strict ranging conditions that avoid reflective ambiguity, new femto-spacecraft can be freely trilaterated onto this cluster in a simple way as a sufficient number of robust relative locations estimates are now known.

The first five femto-spacecraft start the algorithm by defining a relative orientation and position that newly trilaterated positions are found relative to. This process continues until confidence limits of new femto-spacecraft positions are reached based on an estimate of the size of the measurement noise. Intuitively, lower confidence in the range measurements results in the ability to form smaller clusters.

Clusters could form in an ad-hoc fashion based on proximity or identification tags of individual femto-spacecraft. In large networks, this would result in clusters in different relative coordinate systems that can be combined by transformation (translation and rotation) into a single relative reference frame. There are several ways of achieving this [94], provided the clusters share a degree of overlap with localised femto-spacecraft in common (at least four in three-dimensional space). Singular value decomposition (SVD) [95] has been found to be the most stable in computation, so this is the method for frame transformations used in this distributed algorithm. As this can be used in many algorithmic implementations (such as for decentralised SDP computation), the details of this method are described in general terms in Section 3.5.

To describe how the algorithm determines the swarm members' relative positions, the process

is best summarised in its constituent stages, and then each stage can be explained in detail. This can be thought of as the three stages of cluster formation, expansion, and combination. The following steps outline the process:

1. Trilaterate the first set of femto-spacecraft that pass volumetric and ranging tests against positioning ambiguity as positions $\mathbf{x}_1$ - $\mathbf{x}_5$ in the relative frame $\mathcal{F}_1$.
2. Trilaterate further positions onto this cluster using a non-linear least squares iteration process:
   (a) Subject new configurations to the same volumetric and ranging tests for positioning ambiguity
   (b) If a configuration passes volumetric tests, start with a first guess linear least-squares estimate that solves the range equations
   (c) Refine through iteration of a non-linear case with the Newton-Raphson method
   (d) If the position found agrees with ranging test results against flip ambiguity, the femto-spacecraft is localised onto the cluster.
3. Continue to add new femto-spacecraft as in Step 2 until a (pre-determined) threshold on the error propagation in newly trilaterated positions is reached for the cluster.
4. Repeat Steps 1-3 with a new cluster of positions for the entire swarm, ensuring a degree of overlap in clusters for subsequent frame transformations.
5. Transform c clusters in c relative frames $\mathcal{F}_1 - \mathcal{F}_c$ into a single swarm localisation in one relative frame using singular value decomposition.

3.3.2.1 Cluster formation

To start a localisation cluster, five femto-spacecraft must pass both a volumetric test (a check that the solution is valid) and a reflective ambiguity test to work in the presence of range errors from measurements. For these tests, a method similar in three-dimensional space is used to those presented in [96]. As implemented, the two tests work as follows:

1. Volumetric test: prevents poor geometry and measurement noise allowing trilateration of non-robust structures (e.g. in two-dimensional space, the equivalent would be three range measurements failing the triangle inequality). If the probability that a tetrahedron formed by four femto-spacecraft encloses a negative volume is above a pre-determined value (set at 1% in simulation), then the femto-spacecraft are not localised.
2. Reflective ambiguity test: prevents flip/reflective ambiguity by using the otherwise redundant tenth range estimate r_{45} between five femto-spacecraft (Fig. 3.11). A statistical two-tailed z-test [97] is used to determine within a 95% confidence interval that their positions are robust against this reflective ambiguity.

The volume of the tetrahedron $V_{(1,2,3,4)}$ formed by the four femto-spacecraft positions $\mathbf{x}_1$ to $\mathbf{x}_4$ can be found using the ranges between its vertices using the Cayley-Menger determinant [98] as:[3]

$$V_{(1,2,3,4)} = \frac{1}{288}\begin{vmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & r_{12}^2 & r_{13}^2 & r_{14}^2 \\ 1 & r_{12}^2 & 0 & r_{23}^2 & r_{24}^2 \\ 1 & r_{13}^2 & r_{23}^2 & 0 & r_{32}^2 \\ 1 & r_{14}^2 & r_{24}^2 & r_{34}^2 & 0 \end{vmatrix} \tag{3.30}$$

Note here that the notation $\mathbf{x}_i$ to $\mathbf{x}_{i+3}$ would describe the general case for this scenario with four arbitrary relative positions used, but $\mathbf{x}_1$ to $\mathbf{x}_4$ is used in these descriptions and subsequent equations in this description for clarity in the expressions. To determine the probability that $V_{(1,2,3,4)}$ encloses a negative volume, the variance of the volume σ_V^2 is calculated to test this case. Let $\mathbf{r}$ denote the 1×6 vector of the 6 uncorrelated range estimates that make up $V_{(1,2,3,4)}$:

$$\mathbf{r} = \begin{bmatrix} r_{12} & r_{13} & r_{14} & r_{23} & r_{24} & r_{34} \end{bmatrix} \tag{3.31}$$

with an associated covariance matrix $C_\mathbf{r}$ given by:[4]

$$C_\mathbf{r} = \begin{bmatrix} \sigma_{r12}^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_{r34}^2 \end{bmatrix} \tag{3.32}$$

and let F be the matrix of partial derivatives of $V_{(1,2,3,4)}$ with respect to each range (expansion of each term is given in Appendix A.1):

$$F = \begin{bmatrix} \frac{\partial V}{\partial r_{12}} & \frac{\partial V}{\partial r_{13}} & \frac{\partial V}{\partial r_{14}} & \frac{\partial V}{\partial r_{23}} & \frac{\partial V}{\partial r_{24}} & \frac{\partial V}{\partial r_{34}} \end{bmatrix} \tag{3.33}$$

then the variance σ_V^2 of $V_{(1,2,3,4)}$ is given by:

$$\sigma_V^2 = F C_\mathbf{r} F^T \tag{3.34}$$

Now representing the volume estimate as a normally distributed random variable y, with mean V and variance σ_V^2, the probability function is:

$$P(y) = \frac{1}{\sigma_V^2\sqrt{2\pi}} e^{-\frac{(y-V)^2}{2\sigma_V^2}} \tag{3.35}$$

which means the probability that the estimated volume is negative is given by:

$$P_{V<0} = \int_{-\infty}^{0} P(y)dy \tag{3.36}$$

[3] where the determinant of a matrix is given by $|\square|$

[4] As the range estimates are uncorrelated, only the matrix diagonal (containing the variance terms) is non-zero.

Choosing a limit for acceptance would in practice depend on the confidence in the accuracy of the range estimates. This can be set strictly with the expectation of fewer but more accurate and robust localisation, or loosely with the expectation of more but less accurate localisation. In simulation, the condition $P_{V<0} \leq 0.01$ is used for acceptance in cluster formation.

If two tetrahedra $V_{(1,2,3,4)}$ and $V_{(1,2,3,5)}$ pass this volumetric test, the ranging test is performed against reflective ambiguity. For robustness against reflective ambiguity, a statistical two-tailed z-test is used to determine within a 95% confidence interval that the positions in two tetrahedra can be uniquely and correctly aligned. From the calculated geometry obtained from the other range measurements, estimates of the true range r_{45} denoted by $\tilde{r}_{45}$ and $\tilde{r}_{45'}$ can be calculated based on whether the fifth femto-spacecraft is above or below the plane formed with $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ using the geometry shown in Fig. 3.12, where $\tilde{r}_{45} = \sqrt{\Delta h^2 + \Delta q^2}$ and $\tilde{r}_{45'} = \sqrt{\Delta h'^2 + \Delta q^2}$.

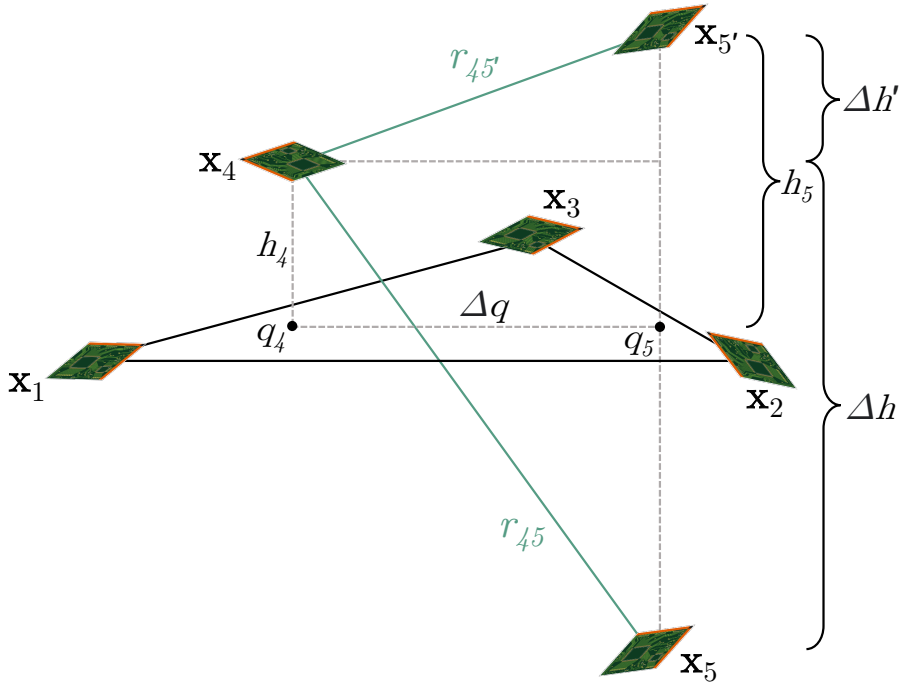


FIGURE 3.12: CALCULATION OF $\tilde{r}_{45}$ AND $\tilde{r}_{45'}$

Derivations for these expressions are provided in Appendix A.2. Now, $\tilde{r}_{45}$ and $\tilde{r}_{45'}$ can be compared with the range measurement $\hat{r}_{45}$ for the ranging test, by setting:

$$\theta_{45} = |\tilde{r_{45}} - \hat{r}_{45}| \tag{3.37}$$

$$\theta_{45'} = |\tilde{r_{45'}} - \hat{r}_{45}| \tag{3.38}$$

and by setting $\theta_{min} = min(\theta_{45}, \theta_{45'})$ and $\theta_{max} = max(\theta_{45}, \theta_{45'})$. Five femto-spacecraft can pass the procedure if and only if θ_{min} is small enough and θ_{max} is large enough; that is, if one can be categorically accepted or rejected. This requires calculation of their respective variances $\sigma_{\theta_{45}}$ and $\sigma_{\theta_{45'}}$, in order to assess if either can be accepted. There are three possible outcomes: both are accepted, in which case it cannot be said which configuration is correct; both are rejected; or the minimum case is accepted while the maximum case is rejected. If this final condition is

met, the five positions are considered robust to form a cluster.

3.3.2.2 CLUSTER EXPANSION

If the volumetric and ranging tests are passed, the relative localisation in the arbitrary local frame $\mathcal{F}_1$ of the first five femto-spacecraft positions of the cluster ($\mathbf{x}_1(x_1, y_1, z_1)$ to $\mathbf{x}_5(x_5, y_5, z_5)$) can be assigned simply from trigonometry:

$$\mathbf{x}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.39}$$

$$\mathbf{x}_2 = \begin{pmatrix} r_{12} \\ 0 \\ 0 \end{pmatrix} \tag{3.40}$$

$$\mathbf{x}_3 = \begin{pmatrix} \frac{r_{13}(r_{23}^2 - r_{12}^2 - r_{13}^2)}{-2r_{12}r_{13}} \\ \sqrt{r_{13}^2 - x_3^2} \\ 0 \end{pmatrix} \tag{3.41}$$

$$\mathbf{x}_4 = \begin{pmatrix} \frac{r_{14}^2 - r_{24}^2 + r_{12}^2}{2r_{12}} \\ \frac{r_{14}^2 - r_{34}^2 + x_3^2 + y_3^2 - 2x_3x_4)}{2y_3} \\ \sqrt{r_{14}^2 - x_4^2 - y_4^2} \end{pmatrix} \tag{3.42}$$

$$\mathbf{x}_5 = \begin{pmatrix} \frac{r_{15}^2 - r_{25}^2 + r_{12}^2}{2r_{12}} \\ \frac{r_{15}^2 - r_{35}^2 + x_3^2 + y_3^2 - 2x_3x_5}{2y_3} \\ \pm\sqrt{r_{15}^2 - x_5^2 - y_5^2} \end{pmatrix} \tag{3.43}$$

where $\mathbf{x}_4$ defines the positive third dimension and the relative orientation of $\pm z_5$ is determined by the reflective ambiguity test. Further femto-spacecraft are then trilaterated onto this cluster using a non-linear least squares refinement process. Using the Newton Raphson method, this starts with a first iteration guess from a linear least-squares solution of the spherical ranging equations which is refined with non-linear least squares. First, the trilateration problem is solved with a system of linear equations involving the new femto-spacecraft to be localised $\mathbf{x}_i(x_i, y_i, z_i)$ and at least four other femto-spacecraft already trilaterated within the cluster.

Importantly, this method uses the range estimates between the femto-spacecraft used that are already localised to the one that is not. This requires at least the four ranges between each of the localised femto-spacecraft and the new femto-spacecraft to be localised to add this position estimate to the cluster. Its formation is still subject to the same volumetric and ranging tests as used for cluster formation. Expressed for the unknown position $\mathbf{x}_i(x_i, y_i, z_i)$ and any position known to the cluster $\mathbf{x}_l(x_l, y_l, z_l)$, where n_c is the number of femto-spacecraft of known position

in the cluster and $l = 1, 2, ..., n_c$, the range equations (Eq. (3.2)) can be written as:

$$\|\mathbf{x}_i - \mathbf{x}_l\|^2 = \sqrt{(x_i - x_l)^2 + (y_i - y_l)^2 + (z_i - z_l)^2} = r_{il}^2 \qquad (l = 1, 2, ..., n_c) \tag{3.44}$$

This can be rearranged into a standard system of linear equations of the form:

$$H\mathbf{y} = \mathbf{b} \tag{3.45}$$

where:

$$H = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ \vdots & \vdots & \vdots \\ x_{n_c} - x_1 & y_{n_c} - y_1 & z_{n_c} - z_1 \end{bmatrix} \tag{3.46}$$

$$\mathbf{y} = \begin{bmatrix} x_i - x_1 \\ y_i - y_1 \\ z_i - z_1 \end{bmatrix} \tag{3.47}$$

$$\mathbf{b} = \frac{1}{2} \begin{bmatrix} (r_{1i}^2 - r_{2i}^2 + r_{12}^2) \\ (r_{1i}^2 - r_{3i}^2 + r_{13}^2) \\ \vdots \\ (r_{1i}^2 - r_{n_c i}^2 + r_{1n_c}^2) \end{bmatrix} \tag{3.48}$$

and solved in a linear least-squares sense as:

$$\mathbf{y} = (H^T H)^{-1} H^T \mathbf{b} \tag{3.49}$$

Solving for $\mathbf{y}$, the newly trilaterated femto-spacecraft position is extracted as $\mathbf{x}_i(x_i, y_i, z_i)$. This is refined using a non-linear least squares method that as a first iteration starts with the initial linear least squares solution. This method minimises the sum of the squares of the range errors, which is achieved by minimising:

$$F(x, y, z) = \sum_{l=1}^{n_c} (\hat{r}_{il} - r_{il})^2 = \sum_{l=1}^{n_c} f_l(x_i, y_i, z_i)^2 \tag{3.50}$$

where $\hat{r}_{il}$ is the estimated range between femto-spacecraft l and the femto-spacecraft i and r_{il} is the true range. Calculating the partial derivatives of Eq. (3.50) with respect to x_i, y_i and z_i yields:

$$\mathbf{g} = 2J^T \mathbf{f} \tag{3.51}$$

where $\mathbf{g}$ is the vector of partial derivatives, J is the Jacobian matrix and $\mathbf{f}$ is a $1 \times n_c$ vector:

$$\mathbf{g} = \begin{pmatrix} \frac{\partial F}{\partial x_i} \\ \frac{\partial F}{\partial y_i} \\ \frac{\partial F}{\partial z_i} \end{pmatrix} \tag{3.52}$$

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_i} & \frac{\partial f_1}{\partial y_i} & \frac{\partial f_1}{\partial z_i} \\ \frac{\partial f_2}{\partial x_i} & \frac{\partial f_2}{\partial y_i} & \frac{\partial f_2}{\partial z_i} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_{nc}}{\partial x_i} & \frac{\partial f_{nc}}{\partial y_i} & \frac{\partial f_{nc}}{\partial z_i} \end{bmatrix} \tag{3.53}$$

$$\mathbf{f} = \begin{pmatrix} \hat{r_{1i}} - r_{1i} \\ \hat{r_{2i}} - r_{2i} \\ \vdots \\ \hat{r_{n_c i}} - r_{n_c i} \end{pmatrix} \tag{3.54}$$

Then, iterating through $(k+1) \leftarrow k$ using the Newton Raphson method for $\mathbf{x}_i$:

$$\mathbf{x}_{i_{k+1}} = \mathbf{x}_{i_k} - (J_k^T J_k)^{-1} J_k \mathbf{f}_k \tag{3.55}$$

This procedure can be repeated for a set number of iterations or until convergence within a given tolerance level. The number of attempts that a femto-spacecraft of unknown position would have to localise onto the cluster would vary according to networking and real-time constraints. While the above strategy works with a minimum of four known positions, if extra positions are available to a femto-spacecraft within the cluster already, these would be available, and in simulation these are used. Other femto-spacecraft can continue to be localised onto a cluster until the solutions exceed noise bounds or confidence levels. At this stage new femto-spacecraft yet to be localised would find another cluster to localise to or would start another cluster altogether.

A distributed computation strategy would utilise the fact that when networking with a cluster of femto-spacecraft of known position, a femto-spacecraft of unknown position would be able to attempt to localise itself relative to that cluster using its own processing power, and then calculate its position estimate algorithmically and communicate this to other femto-spacecraft. Depending on the awareness of other femto-spacecraft of the others' positions, very few additional transmissions would be required to share positioning data and expand the cluster further.

3.3.2.3 Cluster combination

To combine clusters of known positions, a suitable amount of overlap in relative positions is required for the transformation procedure, as described in Section 3.5. Therefore, it is necessary to know some femto-spacecraft positions in at least two reference frames for robust transformations. This would be possible in a step where two clusters communicate and either utilise or establish a necessary degree of overlap in positions using the methods for cluster expansion described.

So far, this algorithm has been described in general terms to highlight the ad-hoc and distributed fashion it could work in, without explicitly considering the role of anchor femto-spacecraft. In the case where anchors are available to particular femto-spacecraft, the intended implementation

of this algorithm is that cluster formation would use anchors to the unknown positions, in order to achieve the best possible accuracy. In particularly large networks, potentially consisting of hundreds of widely dispersed femto-spacecraft where direct communication between all femto-spacecraft may not be practical or possible, it is necessary to consider the case where these clusters localise themselves in relative frames where anchors are, at least directly, unavailable. In such a case, a series of frame transformations could be used to associate cluster positions to a frame shared by anchors. This is discussed in Section 3.5.

The NLLS algorithm is implemented as series of functions in MATLAB. The implementation is explained as pseudocode in Algorithm 3.2, while the logic flow is shown in Fig. 3.13.

Algorithm 3.2 NLLS algorithm

Inputs: r_{ij}, k = 1, (optional: r_{ik}, $\mathbf{a}$)
Output: X

for *cluster formation* **do**
  trilaterate $\mathbf{x}_i$ - $\mathbf{x}_{i+4}$:
  volumetric (solution validity) test
  if *pass* **then**
    ranging (reflective ambiguity) test
    if *pass* **then**
      form cluster with $\mathbf{x}_i$ - $\mathbf{x}_{i+4}$ in relative frame $\mathcal{F}_k$
      $k \leftarrow k + 1$
    end if
  else
    $i \leftarrow i + 1$
  end if
end for
for *cluster expansion* **do**
  while threshold on error propagation not met **do**
    introduce femto-spacecraft j to cluster
    volumetric (solution validity) test
    if *pass* **then**
      trilaterate initial estimate of $\mathbf{x}_j$ using linear least squares
      refine $\mathbf{x}_j$ using non-linear least squares & the Newton-Raphson method
      ranging (reflective ambiguity) test
      if *pass* **then**
        add $\mathbf{x}_j$ onto cluster
        $j \leftarrow j + 1$ (success)
      end if
    end if
    $j \leftarrow j + 1$ (failure)
  end while
end for
for *cluster combination* **do**
  Transform k clusters into 1 relative frame using SVD (Sec. 3.5)
  extract X as the final position estimates
end for

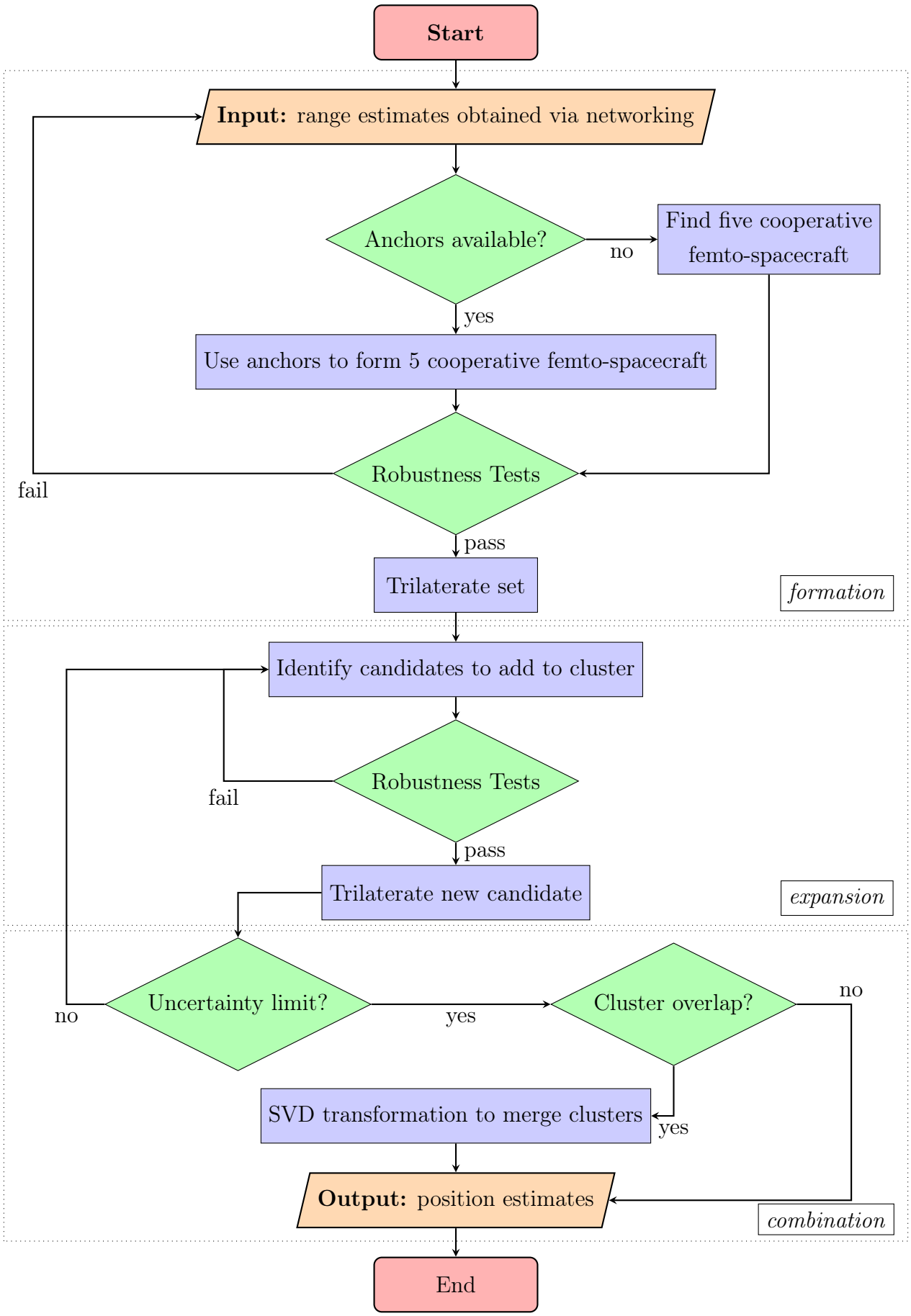


Figure 3.13: NLLS algorithm flowchart

3.4 Normal Probability Density Function Trilateration

As discussed in Section 3.2, the conversion of an RSSI value into a single approximate range estimate is not necessarily as suitable an approach as placing lower and upper bounds on the range approximation. This is particularly true in the case of small networks where the number of constraints may not be sufficient to neglect outlier measurements. This of course applies to all normal measurement types with uncertainty around the mean value, but particularly in the case of experimental testing with a small number of femto-spacecraft proxies. As will be shown in Chapter 6, outliers in experimental data of a small number of devices can massively degrade performance. The SDP algorithm makes use of all available constraints to achieve relative positioning as an optimisation-based feasibility problem. The NLLS algorithm, in its intended distributed implementation, makes use of a large number of available constraints in a distributed network to be selective and rigorous about conditions for estimating relative positions to avoid errors.

For experimental testing and demonstration, as discussed in Chapter 6, the following algorithm using normal probability density functions (NPDFs) was also implemented. In 2D space, the confidence bounds of the range estimates are directly treated to create range annuli, where the intersections of annuli represent the highest likelihood of location [99]. This compensates for the inaccuracy of RSSI as a ranging metric by focusing instead on creating binding regions where a femto-spacecraft may lie within. This process estimates positions of unknown femto-spacecraft using knowledge of their range estimates to already known positions. As the known positions could have been found from the algorithmic methods in the previous sections, this could work as a cluster expansion method or make use of anchors. For generality, the known positions in the following descriptions are given the notation of anchors. For the unknown position $\mathbf{x}_i(x_i, y_i, z_i)$ and anchor position $\mathbf{a}_k(x_k, y_k, z_k)$, with range estimate $\hat{r}_{ik}$ and standard deviation in the range estimate $\sigma_{r_{ik}}$, the NPDF takes the form:

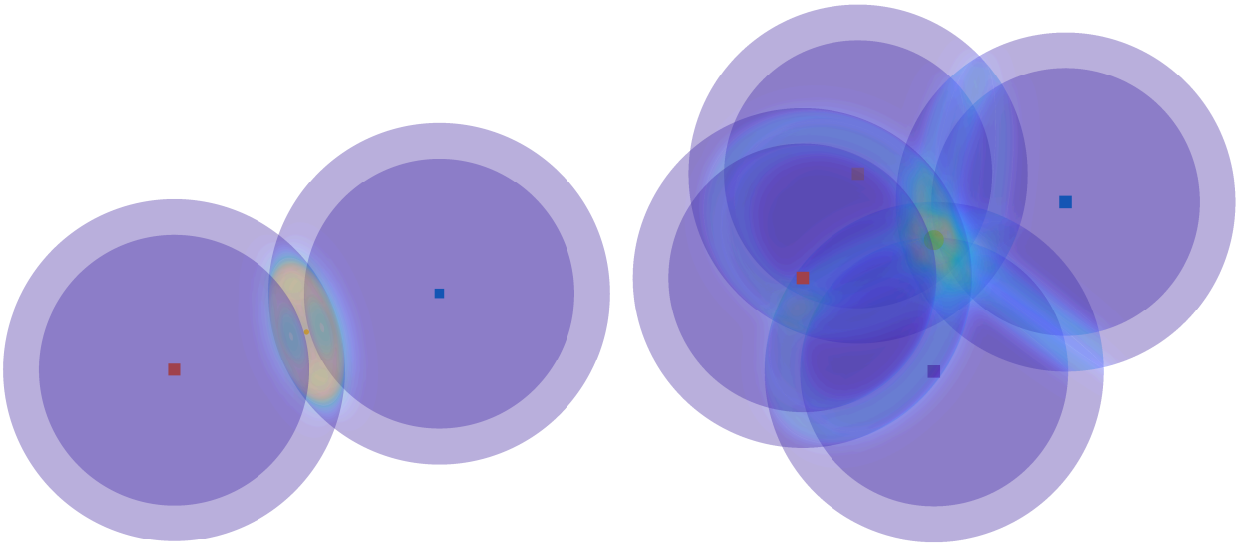
$$P_k(x_i, y_i, z_i) = \frac{1}{\sqrt{2\pi\sigma_{r_{ik}}^2}} e^{-\frac{\left(\sqrt{(x_i-x_k)^2+(y_i-y_k)^2+(z_i-z_k)^2}-\hat{r}_{ik}\right)^2}{2\sigma_{r_{ik}}^2}} \tag{3.56}$$

where P is the representation of a spherical shell volume (the 3D generalisation of an annulus) of probability that binds the estimate of the unknown position in 3D space. This treats the range estimate as the mean value around which the confidence in the range estimate is measured. The NPDF localisation method can be visualised as a normalised heat map whereby finding:

$$P(x_i, y_i, z_i) = \sum_{k=1}^{m} P_k(x_i, y_i, z_i) \tag{3.57}$$

results in a region of highest probability emerging to localise the unknown position relative to the other known position. This can be normalised between 0 and 1 by dividing through by the

maximum P_k value. As in Section 3.1, m denotes the total number of localised anchor positions. This can be visualised in general terms with an unknown node's relationship between known positions as iso-surfaces in 3D space, with a heat map representative of probability, as shown in Fig. 3.14.



(a) 3D torus of probability (b) Unique region

Figure 3.14: Visual representation of $P(x_i, y_i, z_i)$ as a 3D heat-map

The hottest regions show where the unknown position can be estimated as lying within. With a sufficient number of intersecting femto-spacecraft range estimates, the centroid of this probable region can be taken as the estimate of the unknown position. As shown in Fig. 3.14a, two known positions' ranges to the unknown position results in a 3D torus of probability, while in Fig. 3.14b, four known positions result in a unique region inside which the unknown position lies. This approach works best when the confidence in range estimate being used is well defined. In the simplest case, this can be used as a step for trilateration of further femto-spacecraft onto clusters. Its usage with inter-spacecraft relative positions using r_{ij} is however limited, owing to the high uncertainty in positions that would be found without using the conditions of approaches like the SDP or NLLS algorithms. This is implemented in MATLAB for experimental testing comparisons and visualisation of RSSI data in a path loss model, which for small-scale testing offers insight into the reliability of RSSI as a range metric.

Algorithm 3.3 NPDF Algorithm

Inputs: r_{ik}, $\sigma_{r_{ik}}$, $\mathbf{a}$

Output: $\mathbf{x}_{ik}$

1: Generate a 3D Cartesian mesh-grid that spans the feasible search space for the unknown position

2: Calculate all P_k for k = 1,2,...,m as 4D arrays (3D positions and 1D normalised probability)

3: Sum all $P = \sum P_k$, and normalise with $P = \frac{P}{max(P_k)}$

4: Find and extract coordinates of max(P) as $\mathbf{x}_{ik}$

3.5 Singular Value Decomposition Frame Transformation

For the relative positioning algorithms described, any distributed computation would facilitate real-time relative positioning in networks of greater sizes by sharing the computational requirement of each spacecraft within the swarm. For the SDP algorithm, which is an inherently centralised approach, decentralisation can be introduced with several spacecraft acting in a centralised way. Frame transformations between clusters of spacecraft would enable faster computation of a larger swarm's relative positions. For the NLLS algorithm, which is intended to work in a distributed fashion, frame transformations are an essential step in cluster expansion and combination.

To combine two clusters of positions known in two different reference frames into a single cluster known in one with a transformation, it is necessary to have some proportion of positions known in both sets, i.e. a degree of overlap. Singular value decomposition (SVD) frame transformation can be used given the coordinates of a minimum of the same four non-coplanar positions known in two separate frames of reference, such that a transformation (rotation matrix and translation vector) between the two frames can be found [95, 100].

Consider the two sub-clusters of femto-spacecraft positions $X_{\mathcal{F}_1}$ and $X_{\mathcal{F}_2}$, representing two $3 \times n$ matrices of the same $n \geq 4$ position vectors expressed in the two different reference frames $\mathcal{F}_1$ and $\mathcal{F}_2$. Note here that n represents a number of femto-spacecraft known in both frames, and not the total number in the swarm. In the absence of any errors, the sets are related by a 3×3 rotation matrix R and a 3×1 translation vector T such that:

$$X_{\mathcal{F}_1} = RX_{\mathcal{F}_2} + T \tag{3.58}$$

As in practice, both sets of positions will have inaccuracies in the position estimates unique to each set, an objective of this transformation is instead to find an optimal $\hat{R}$ and $\hat{T}$ to minimise the least squares error criterion:

$$\sum_{i=1}^{n} \left\| X_{i_{\mathcal{F}_1}} - \hat{R}X_{i_{\mathcal{F}_2}} - \hat{T} \right\|^2 \tag{3.59}$$

Consequently, if a solution $\hat{R}$ and $\hat{T}$ exists then the sets $X_{\mathcal{F}_1}$ and $X_{\mathcal{F}_2}$ have the same centroids [95]:

$$\bar{X}_{\mathcal{F}_1} = \frac{\sum_{i=1}^{n} X_{i_{\mathcal{F}_1}}}{n} \tag{3.60}$$

$$\bar{X}_{\mathcal{F}_2} = \frac{\sum_{i=1}^{n} X_{i_{\mathcal{F}_2}}}{n} \tag{3.61}$$

giving the centred vectors of:

$$X_{c_{i_{\mathcal{F}_1}}} = X_{i_{\mathcal{F}_1}} - \bar{X_{\mathcal{F}_1}} \tag{3.62}$$

$$X_{c_{i_{\mathcal{F}_2}}} = X_{i_{\mathcal{F}_2}} - \bar{X_{\mathcal{F}_2}} \tag{3.63}$$

simplifying the criterion in Eq. (3.59) to:

$$\sum_{i=1}^{n} \left\| X_{i_{\mathcal{F}_1}} - \hat{R} X_{i_{\mathcal{F}_2}} - \hat{T} \right\|^2 = \sum_{i=1}^{n} \left(X_{c_{i_{\mathcal{F}_1}}}^T X_{c_{i_{\mathcal{F}_1}}} + X_{c_{i_{\mathcal{F}_2}}}^T X_{c_{i_{\mathcal{F}_2}}} - 2 X_{c_{i_{\mathcal{F}_1}}}^T \hat{R} X_{c_{i_{\mathcal{F}_2}}} \right) \tag{3.64}$$

which is minimised by maximising the trace of $\hat{R}H$ [100], where H is a correlation matrix given by:

$$H = \sum_{i=1}^{n} X_{c_{i_{\mathcal{F}_2}}} X_{c_{i_{\mathcal{F}_1}}}^T \tag{3.65}$$

Finding the singular value decomposition of H:

$$H = U \Sigma V^T \tag{3.66}$$

the optimal rotation matrix $\hat{R}$ and translation vector $\hat{T}$ are found to be:

$$\hat{R} = V U^T \tag{3.67}$$

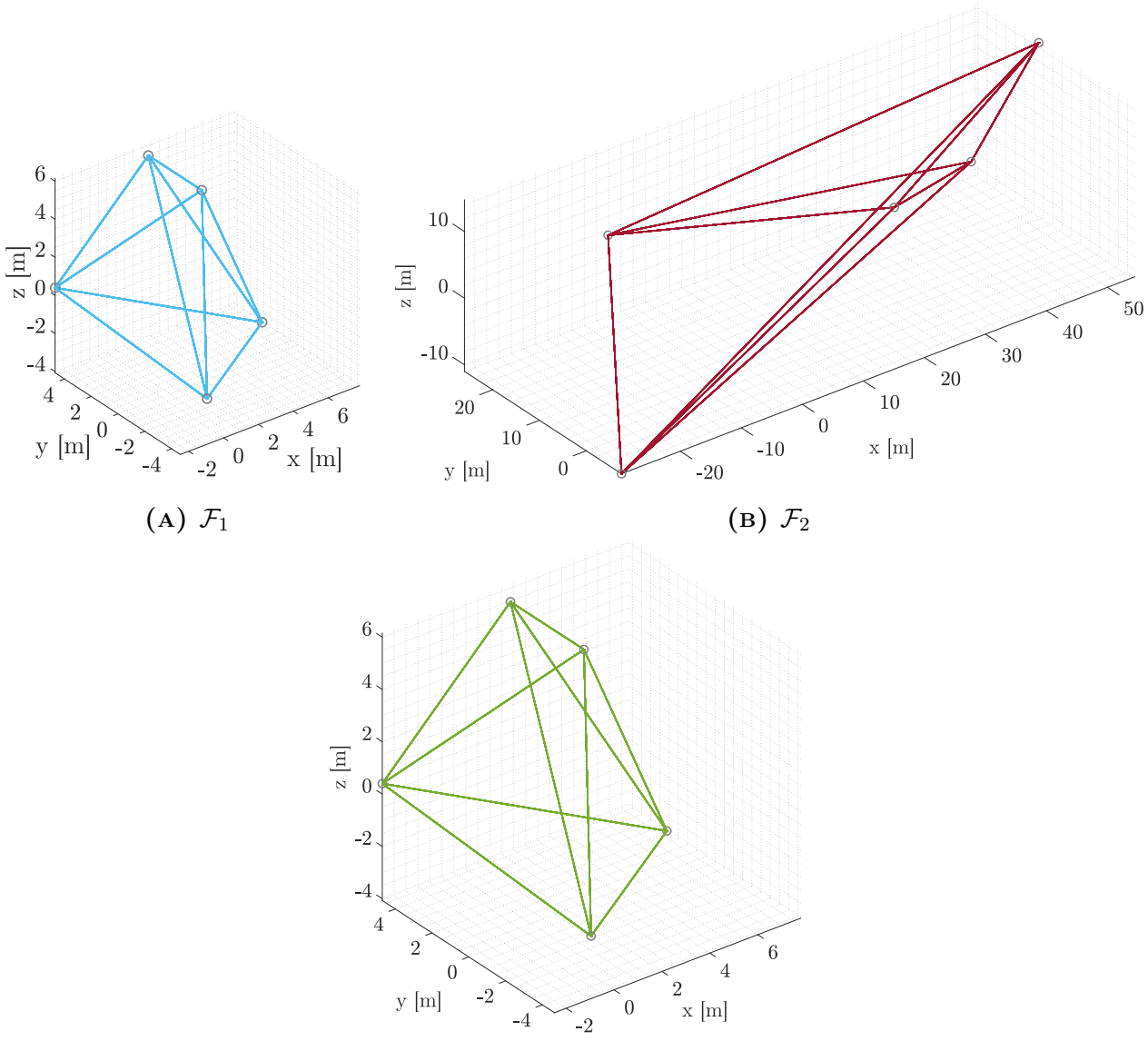
$$\hat{T} = \bar{X_{\mathcal{F}_2}} - \hat{R} \bar{X_{\mathcal{F}_1}} \tag{3.68}$$

providing an optimal transformation between the two sets of positions. Then, the general approximation for the matrix of femto-spacecraft positions X can be made:

$$X_{\mathcal{F}_1} \approxeq \hat{R} X_{\mathcal{F}_2} + \hat{T} \tag{3.69}$$

This can be tested for robustness by finding the determinant $|\hat{R}|$. If $|\hat{R}| = 1$, $\hat{R}$ represents a rotation and can be used, otherwise $|\hat{R}| =$ -1, indicating a reflection and preventing the transformation from being useful. There have been approaches outlined to address this in the literature [100], but in simulation and testing the author has not found these to work. A reflection case indicates outliers in the relative positioning, which if not already addressed by algorithmic techniques, can likely not be dealt with otherwise. In such (rare) failure cases, different point set combinations can be tried to establish an optimal transformation. Issues such as randomly selected position sets either being (or close to being) coplanar accounts for any known instances of this error in testing. For demonstration of this method, consider the test case for $n = 5$ randomly placed femto-spacecraft known in the two reference frames $\mathcal{F}_1$ and $\mathcal{F}_2$ shown in Fig. 3.15. Given $\hat{R}$ and $\hat{T}$, the positions in $\mathcal{F}_2$ can be transformed into $\mathcal{F}_1$ as shown in Fig. 3.15c.

In the singular value decomposition, U and V are 3×3 unitary matrices and Σ is a 3×3 diagonal matrix. The main advantage of this rotation and translation is that further positions known in either reference frame can be transformed as required, so it only requires the small amount of overlap in clusters to know all positions in both clusters in either frame. This is implemented as a function in MATLAB as an integral step in the NLLS algorithm, for decentralisation in the SDP algorithm, and for general usage. This is outlined as pseudocode below.



(a) $\mathcal{F}_1$

(b) $\mathcal{F}_2$

(c) $X_{\mathcal{F}_1} = \hat{R}X_{\mathcal{F}_2} + \hat{T}$

Figure 3.15: SVD transformation

Algorithm 3.4 SVD Frame Transformation Algorithm

Inputs: $X_{\mathcal{F}_1}$,$X_{\mathcal{F}_2}$
Outputs: $\hat{R}, \hat{T}$

find the centroids of $X_{\mathcal{F}_1}$ and $X_{\mathcal{F}_2}$ as $\bar{X}_{\mathcal{F}_1}$ and $\bar{X}_{\mathcal{F}_2}$
find the centred vectors $X_{c_{i_{\mathcal{F}_1}}}$ and $X_{c_{i_{\mathcal{F}_1}}}$, as arrays of vector combinations
find the 3×3 correlation matrix $H = \sum_{i=1}^{n} X_{c_{i_{\mathcal{F}_2}}} X_{c_{i_{\mathcal{F}_1}}}^T$
compute the singular value decomposition $H = U\Sigma V^T$
determine the optimal rotation matrix $\hat{R} = VU^T$
if $|\hat{R}| = 1$ **then**
    return $\hat{R}$
    return the optimal translation vector $\hat{T} = \bar{\mathbf{X}}_{\mathcal{F}_2} - \hat{R}\bar{\mathbf{X}}_{\mathcal{F}_1}$
else
    if $|\hat{R}| = -1$ **then**
        return 0 - transformation failed
    end if
end if

3.6 Simulation-based Algorithmic Performance Analysis

For algorithmic performance analysis, a series of tests are simulated for femto-spacecraft generated in randomised positions within a controlled volume of space. These are a centralised, decentralised, and distributed test case of $n = 100$ femto-spacecraft and $a = 10$ anchor spacecraft randomly generated within a cubic volume $V = 200$ m^3. In all tests, the true ranges between femto-spacecraft are distorted with additive white Gaussian noise (AWGN), varying its standard deviation (σ_r) to examine how the algorithm performs. The mean squared error in range estimates, ${\sigma_r}^2$, is given by:

$${\sigma_r}^2 = \sum_{i=1}^{N} \frac{(\hat{r}_i - r_i)^2}{N} \tag{3.70}$$

where N is the number of range estimates, $\hat{r}_i$ is the range estimate supplied to the algorithm and r_i is the true range. This treatment of the range measurement in simulation is done so as to not restrict the analysis to RSSI or any particular range metric and its inaccuracies or biases. The algorithms are supplied only with the noisy range estimates and any anchor positions. For simplicity, anchors are assumed to be located without any error in position. The mean squared error of the localised femto-spacecraft ${\sigma_X}^2$ is used to analyse the algorithm's localisation accuracy with the true positions, such that:

$${\sigma_X}^2 = \sum_{i=1}^{n} \frac{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 + (\hat{z}_i - z_i)^2}{n} \tag{3.71}$$

where n is the number of femto-spacecraft, X is the matrix of all known position estimates, position $\mathbf{x}_i$ has true coordinates (x_i, y_i, z_i) and the algorithm estimates these coordinates to be $(\hat{x}_i, \hat{y}_i, \hat{z}_i)$. Inaccuracies in the range measurements are varied by introducing a noise level and random error proportionate to the true range such that:

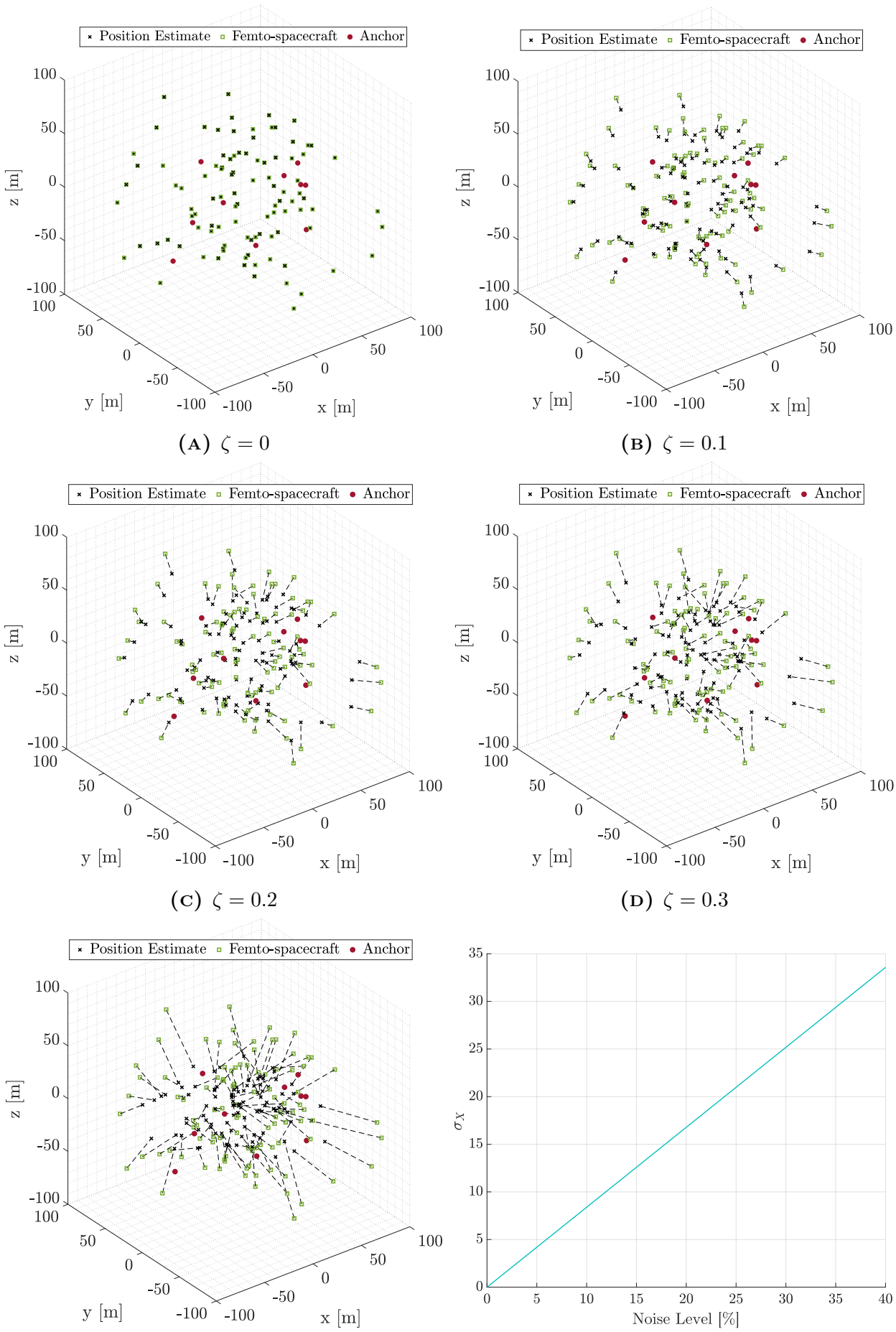
$$\hat{r}_{ij} = r_{ij}(1 + \zeta\chi) \tag{3.72}$$

where χ is a zero mean normally distributed random variable of variance one, and ζ is a noise level greater than or equal to zero. For example, a noise level of 10% corresponds to $\zeta = 0.1$.

3.6.1 SDP algorithm centralised test case

For a test case of the SDP algorithm centralised performance analysis, the noise level is varied from 0-40%, and full interconnectivity within the swarm is assumed, i.e. that there is a range estimate between every femto-spacecraft and to the anchors. The results shown in Fig. 3.16 show single-run results of the SDP algorithm estimating the same positions with noise levels of $\zeta = 0$, 0.1, 0.2, 0.3 and 0.4.

In Fig. 3.16, the anchor coordinates are marked by red circles, the true coordinates (unknown



(A) $\zeta = 0$

(B) $\zeta = 0.1$

(C) $\zeta = 0.2$

(D) $\zeta = 0.3$

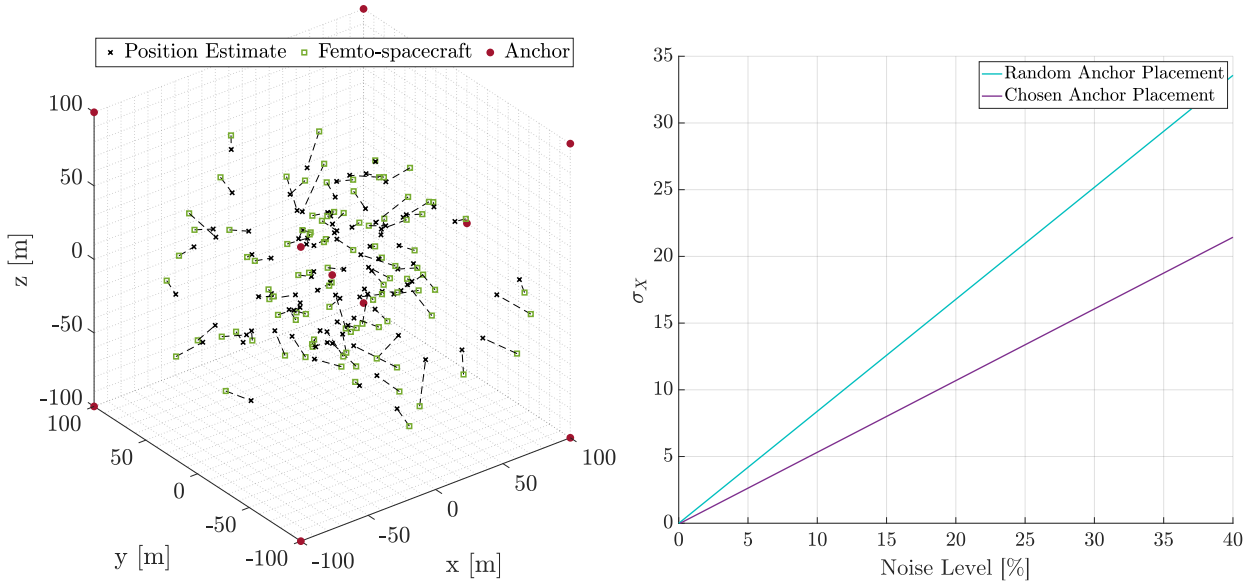
(E) $\zeta = 0.4$

(F) σ_X with noise level

Figure 3.16: SDP algorithm performance in centralised test case

to the algorithm) are marked by green squares and the algorithm output position estimates are marked by black crosses. The dashed lines indicate the difference between an estimate and the true position for that point. Figures 3.16a-3.16e show the sample results of this test case, while Fig. 3.16f shows the linear interpolation of the trend in the mean squared error in position σ_X with increasing noise levels as a percentage of r_{ij}. As can be expected, this shows that the errors in position increase directly in proportion to increases in range errors. With randomised anchor placement, this test case demonstrates that due to the convex constraints, estimated positions tend to be least accurate towards and at the outer edges of the swarm, with Fig. 3.16e highlighting the inaccuracy in estimating positions far nearer to the centre of the swarm than the true positions.

The effect of this can be mitigated to an extent with consideration of anchor placement within the swarm. Placing anchors towards the edges of the swarm is found to improve the accuracy in general, due to minimising any effects of poor geometry for the convex constraints. For example, repeating this test case, instead now with nine anchors located at the boundaries of the test volume and one at the centre, as shown in Fig. 3.17, this effect is demonstrated.



(A) $\zeta = 0.3$ (B) σ_X WITH NOISE LEVEL

FIGURE 3.17: SDP ALGORITHM PERFORMANCE IN CENTRALISED TEST CASE WITH DELIBERATE ANCHOR PLACEMENT

It is found that the positioning accuracy is consistently better across all noise levels. Only the full $\zeta = 0.3$ result is displayed in full in Fig. 3.17a for brevity, with the trend in Fig. 3.17b highlighting the general performance improvement across all noise levels. It is clear that this anchor placement improves the general algorithm positioning performance at all noise levels, implying that where possible, anchors should be selected to be positioned at the edges of the swarm for an expected improvement in positioning accuracy. In practice, if anchors were for example to be GPS-equipped femto-spacecraft, this placement may be achieved by ejecting anchor spacecraft either before the rest of the swarm or at higher ejection velocities. Ejection

strategies are discussed in Chapter 4. While the test case here shows anchors placed deliberately, in reality, anchors could still take quite random positions, but if congregating around the edges of the swarm this could in principle achieve better accuracy than with random placement.

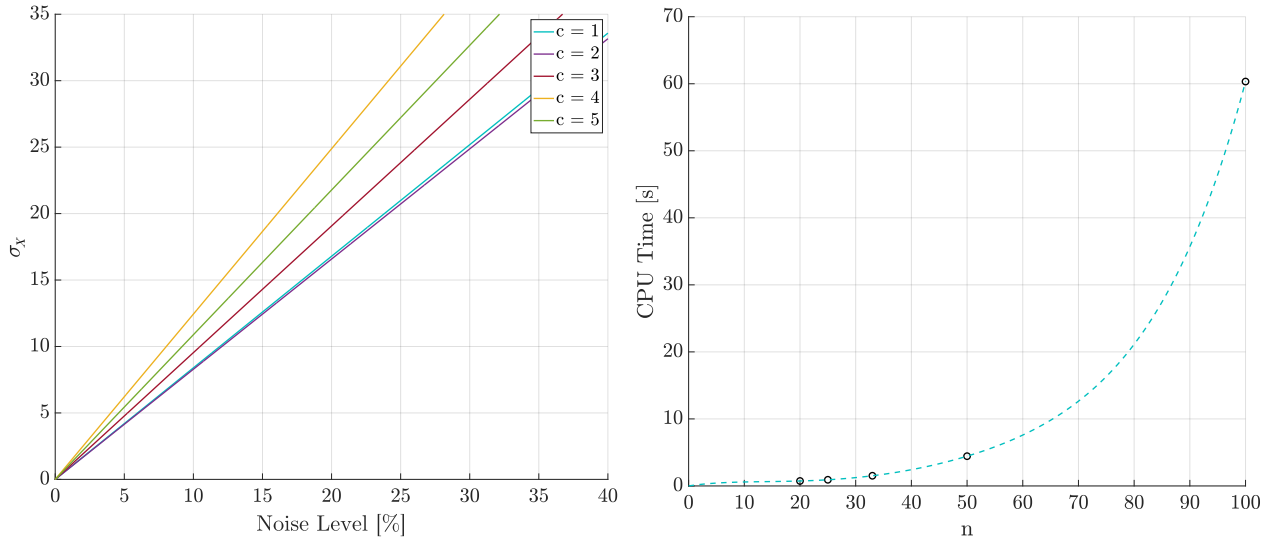
3.6.2 SDP algorithm decentralised test case

The centralised test case can be straightforwardly decentralised by partitioning the swarm into separate clusters, assuming the case of the swarm either having multiple deployers or being composed of a small fraction of larger spacecraft able to perform the centralised computation. Again for $n = 100$ and $a = 10$, the random positions are partitioned into between two to five clusters c of fifty to twenty femto-spacecraft respectively. All anchor positions are known to each cluster head. As an operational comparison to the single cluster case, all anchor positions are known, so unlike the NLLS algorithm, there is no need for overlap in the femto-spacecraft known by each cluster requiring SVD frame transformation.

For relative positioning performance, this comparison is essentially a trade-off between accuracy and computation time. Considering full interconnection of n femto-spacecraft, the number of communication links (and therefore range constraints for the algorithm) scales with $\frac{n(n-1)}{2}$, implying quadratic time complexity scaling with the number of constraints (i.e. $\mathcal{O}(n^2)$). As implemented in MATLAB with cvx, the SDP algorithm works using the solver SDPT3 [101]. For each iteration of the solver, the worst-case time complexity is theoretically $\mathcal{O}(n^6)$ [89, 102]. However, in practice this scaling can be substantially reduced with constraints on connectivity and communications range limits for substantially larger networks, typically bounding the typical worst-case complexity to $\mathcal{O}(n^3)$ [89,103]. The polynomial-time complexity ultimately means that decentralised clustering will enable substantially faster computation in larger networks, at the cost of positioning accuracy due to the decreased number of constraints.

Considering the four test case partitions of c = 2-5 clusters, the accuracy and computation time can be assessed from the simulation results using the same test positions as in the centralised test case, as shown in Fig. 3.18. For this test case, this shows that in general, larger clusters with greater numbers of range constraints can be expected to offer improved relative positioning performance, at the cost of computation time. The noise level in the range estimates was found in testing to have a small effect on the total computation time, with the results in Fig. 3.18b showing the case for the noise level of $\zeta = 0.3$. This is negligible in the order of magnitude compared to the effect of n, but naturally becomes a greater consideration with higher n. The full computational run-times simulated were 0.68-0.75 s for $n = 20$, 0.90-0.98 s for $n = 25$, 1.36-1.54 s for $n = 33$, 4.02-4.61 s for $n = 50$ and 53.54-65.26 s for $n = 100$.

The CPU times given here are simply indicative of the time complexity scaling and are shown for comparison. Note that all simulations were performed on a MacBook Pro with a 1.4 GHz Quad-Core Intel Core i5 processor and 8 GB RAM, running MATLAB 2021B. These values correspond only to the simulation parameters tested. However, it is clear that due to the time complexity of the algorithm, clustering is a desirable extension for larger networks to operate



(A) σ_X with noise level for c = 1-5 (B) CPU Time with n (shown for $\zeta = 0.3$)

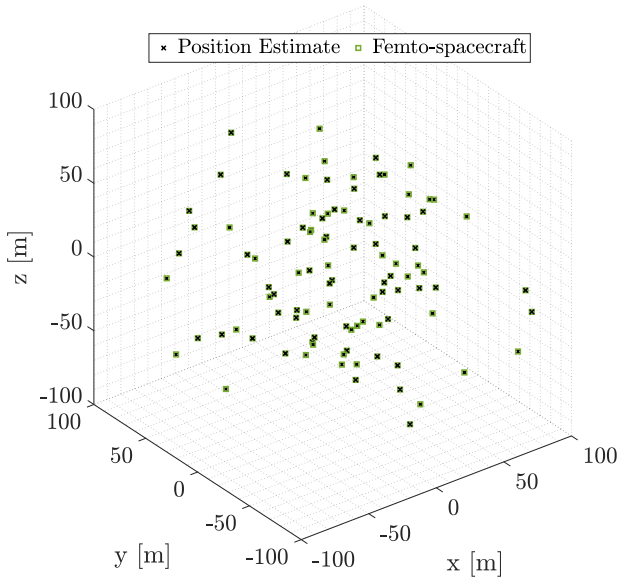
Figure 3.18: SDP algorithm performance in decentralised test case

in providing relative navigation in real time.

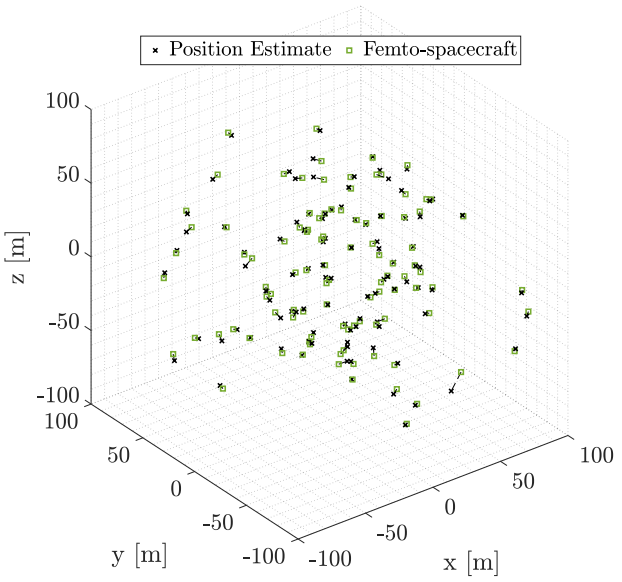
3.6.3 NLLS algorithm distributed test case

To assess the NLLS algorithm performance in a particular test case, the same random positions from the centralised and decentralised test cases are used, without use of the anchor positions. This is done to demonstrate and assess the NLLS algorithm's ability to work distributively in an anchorless way. All positions are localised into a single relative frame in one cluster. Finally, after the algorithm has output the positions estimates, all are transformed into the same frame of reference used to generate the random points for a performance analysis of the algorithm's accuracy. The utilisation of the NLLS algorithm in the presence of anchors has been discussed. However, as a key benefit of this approach is to work without anchors for larger network sizes or where the swarm has been deployed in orbit with varying communications ranges between all femto-spacecraft, this test case highlights its performance in a general way, without the use of anchors. Consider the distributed test case shown in Fig. 3.19.

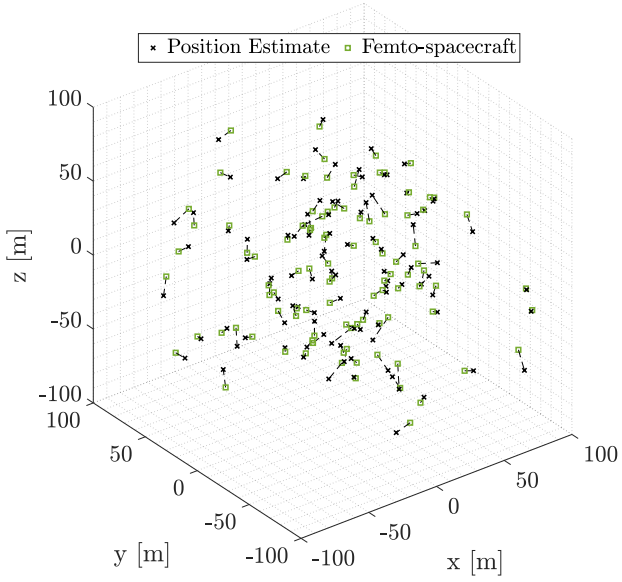
This test case demonstrates the ability of the distributed algorithm to localise the femto-spacecraft up to a noise level of $\zeta = 0.2$, which, with the tolerances of 99% certainty of robustness against volumetric and ranging ambiguity in simulation, is found to be the upper limit at which this algorithm can operate for a swarm of this size in this test scenario. This represents the best performance that can be expected in simulation using the thresholds described for the volumetric and ranging tests to prevent incorrect localisation of femto-spacecraft. Ultimately, there is a demonstrable trade-off between positioning accuracy and the fraction of femto-spacecraft localised in a particular sample at a point in time when implemented using real data. As the noise level increases, there is an expected general trend towards fewer first attempt localisations within the algorithm (i.e. femto-spacecraft localised using range estimates between the first four femto-spacecraft attempted). In practice, frequent sampling of the al-



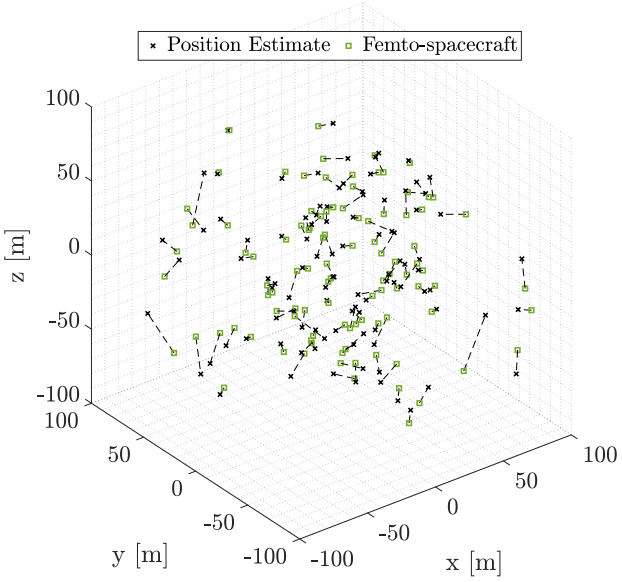
(A) $\sigma_r = 0$



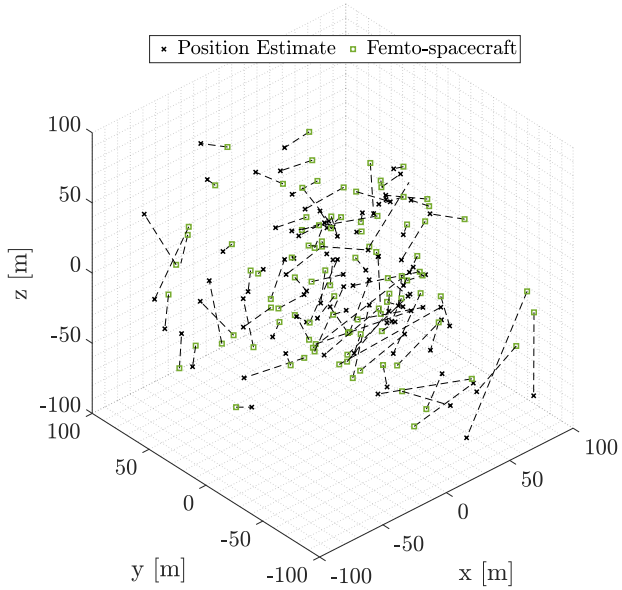
(B) $\sigma_r = 5\%\bar{r}$



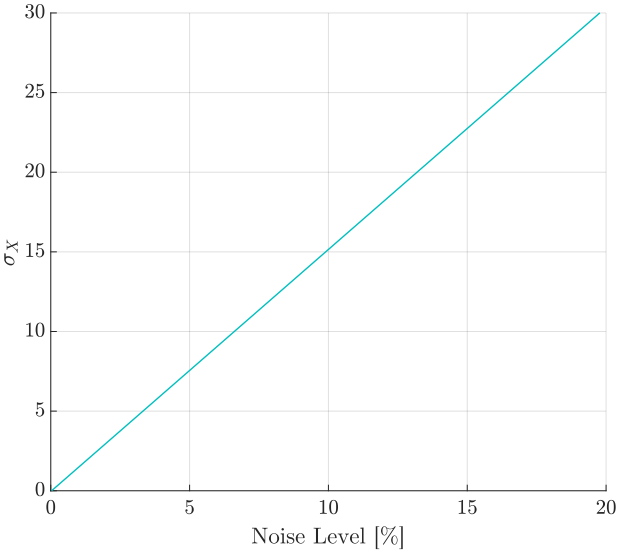
(C) $\sigma_r = 10\%\bar{r}$



(D) $\sigma_r = 15\%\bar{r}$



(E) $\sigma_r = 20\%\bar{r}$



(F) σ_X with noise level

Figure 3.19: NLLS algorithm performance in distributed test case

gorithm would update the relative position estimates of the swarm to account for the swarm state changing in space with time. Femto-spacecraft that fail to localise well on a particular algorithm cycle would have the opportunity to do so on the next cycle.

It can be expected that fewer localisations would occur in the presence of higher noise levels. This is left as a point of practical implementation with the use of experimental data and a well understood accuracy of the range metric when this algorithm is utilised with larger networks than those experimented with in this work. In simulation, these results demonstrate its ability to localise all femto-spacecraft up to a large degree of measurement noise relative to the average range between femto-spacecraft. This emphasises the contrasting approaches between the SDP algorithm and the NLLS algorithm, in that centralised approaches can make use of far more constraints for optimisation.

3.7 Chapter Summary

This chapter has presented several approaches for enabling relative positioning for a swarm of femto-spacecraft using only range estimates from network communications. The algorithms implemented have been developed to facilitate centralised, decentralised, and distributed computation within the swarm. Simulation-based results of the SDP algorithm and NLLS algorithm in randomly generated test cases demonstrate their utility in localising the swarm with range estimates of varying accuracy. Discussion of the accuracy of the range metric available has not been limited to RSSI in this chapter so as to highlight the general applicability of range-based positioning methods the algorithms offer for large networks of computationally limited devices, where other (or a combination of) range metrics may be of use.

The algorithms presented here are used as measurement updates at discrete time points within relative navigation filters presented in Chapter 4. These are then experimentally tested in Chapter 6.

Chapter 4

Relative Navigation Techniques

In this chapter, the relative positioning approach from Chapter 3 is applied to femto-spacecraft swarm dispersal and evolution scenarios using a relative dynamics model of the swarm's motion with respect to its deployer. The case of a swarm being dispersed in orbit from a single deployer spacecraft (such as a larger CubeSat carrier) is considered. This enables the development of relative navigation filters, to demonstrate the utilisation of the relative positioning algorithms in practice. Algorithm outputs at discrete time steps act as the measurement update stage in Kalman filtering techniques.

Firstly, the relative dynamics model used to propagate the dynamics of the swarm with respect to its deployer is presented. The scenario considered is of a swarm of femto-spacecraft being deployed in LEO from a larger carrier spacecraft. Then, several swarm dispersal and evolution strategies are considered, which can be categorised broadly into randomised and controlled ejection procedures. Finally, Kalman filter approaches are applied to demonstrate the utility of the algorithm outputs simulated as being sampled in low Earth orbit (LEO) for both dispersal procedures.

4.1 Relative Dynamics Model

The Clohessy-Wiltshire (CW) equations [104] provide a linearised approximation of the relative motion of a 'chaser' spacecraft with respect to a 'target' spacecraft in a target-centred reference frame. These equations are primarily used in the context of spacecraft rendezvous and docking manoeuvres, or the general relative dynamics between two satellites operating in close proximity. In this work, the terms 'femto-spacecraft' and 'deployer' will be used to describe the chaser and target spacecraft respectively. As such, this enables analysis of how a swarm of many femto-spacecraft could be dispersed from a larger carrier spacecraft and how the dynamics would evolve.

The CW equations assume the deployer to be in a circular orbit while the femto-spacecraft drift passively relative to it. Perturbations to the two-body problem are neglected in this analysis.

The CW equations are defined as:

$$\ddot{x} - 2\omega_n \dot{y} - 3\omega_n^2 x = 0 \tag{4.1}$$

$$\ddot{y} + 2\omega_n \dot{x} = 0 \tag{4.2}$$

$$\ddot{z} + \omega_n^2 z = 0 \tag{4.3}$$

where:

$$\omega_n = \sqrt{\frac{\mu}{R_o^3}} = \frac{2\pi}{T_o} \tag{4.4}$$

The mean motion of the deployer ω_n is expressed in terms of the standard gravitational parameter of the central body μ and the orbital radius R_o of the deployer's orbit, which has an orbital period T_o. In this deployer-centred reference frame, the x-axis points outwards along the radius vector of the deployer spacecraft (radial motion), the y-axis points forwards along the velocity vector (along-track motion), and the z-axis completes the right-handed set by pointing along the deployer's orbital angular momentum vector (cross-track motion). This means that the central orbital body (in this scenario, Earth) is towards the negative x-direction of the deployer. This is illustrated in Fig. 4.1.

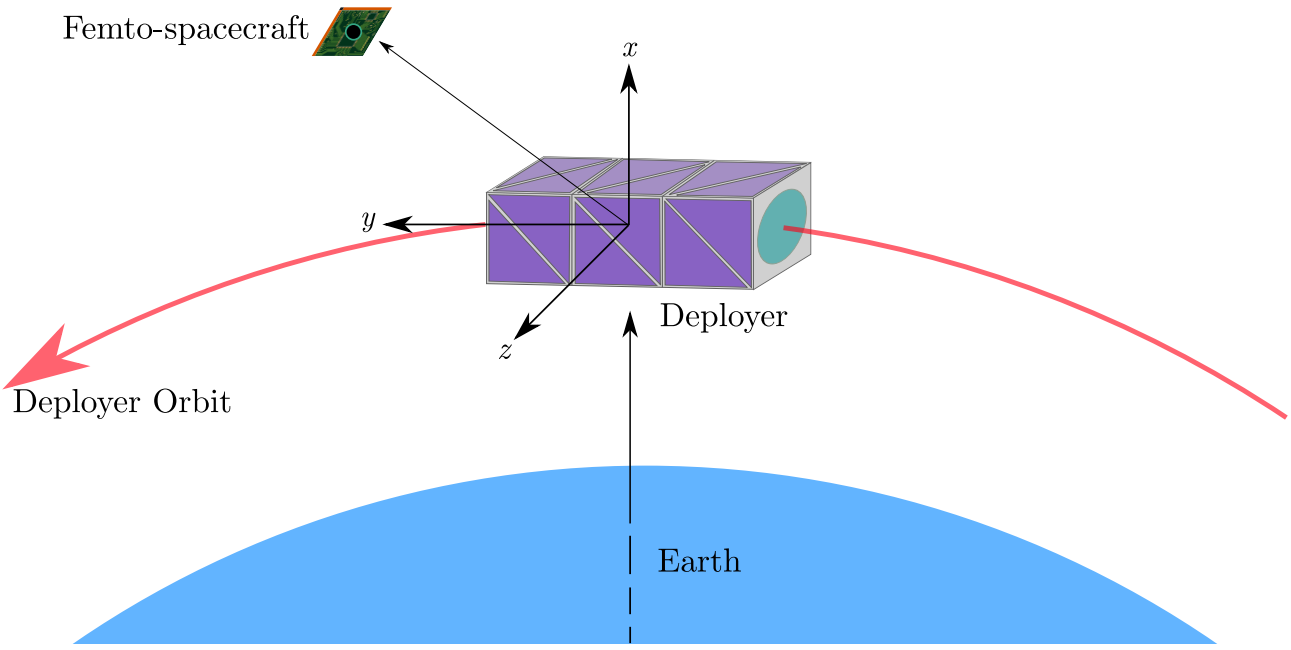


FIGURE 4.1: CLOHESSY-WILTSHIRE REFERENCE FRAME

Expressing the CW equations in state space form allows a closed form solution to be expressed such that:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) \tag{4.5}$$

where the system matrix A and the state vector $\mathbf{x}(t)$ are given by:[1]

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega_n^2 & 0 & 0 & 0 & 2\omega_n & 0 \\ 0 & 0 & 0 & -2\omega_n & 0 & 0 \\ 0 & 0 & -\omega_n^2 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{x}(t) = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{4.6}$$

The CW equations can then be solved in terms of a state transition matrix Φ and the initial conditions $\mathbf{x}(t_0)$ such that:

$$\mathbf{x}(t) = e^{A(t-t_0)}\mathbf{x}(t_0) = \ \Phi\mathbf{x}(t_0) \tag{4.7}$$

Abbreviating $\sin(\omega_n t) = s$ and $\cos(\omega_n t) = c$:

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \end{bmatrix} = \begin{bmatrix} 4-3c & 0 & 0 & \frac{s}{\omega_n} & \frac{2}{\omega_n} - \frac{2c}{\omega_n} & 0 \\ 6s - 6\omega_n t & 1 & 0 & \frac{2c}{\omega_n} - \frac{2}{\omega_n} & \frac{4s}{\omega_n} - 3t & 0 \\ 0 & 0 & c & 0 & 0 & \frac{s}{\omega_n} \\ 3\omega_n s & 0 & 0 & c & 2s & 0 \\ 6\omega_n c - 6\omega_n & 0 & 0 & -2s & 4(c-3) & 0 \\ 0 & 0 & -\omega_n s & 0 & 0 & c \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ \dot{x_0} \\ \dot{y_0} \\ \dot{z_0} \end{bmatrix} \tag{4.8}$$

With this closed form solution, the state of a femto-spacecraft can be propagated in time using the state transition matrix. The CW equations are implemented as a relative dynamics model in MATLAB. This model is applied to consider how a femto-spacecraft swarm ejected from a deployer may disperse and drift with time, and how the relative positioning algorithms proposed can estimate the location of each swarm member. The initial state vector of each femto-spacecraft deployed is always coincident with the deployer, with some initial ejection velocity for each femto-spacecraft relative to the deployer.

In this analysis, it is assumed that the femto-spacecraft have no means of controlling their relative positions. It is also assumed that the initial conditions for the dynamics are defined entirely by the deployer. This is done to simplify the discussion of relative navigation for the purposes of demonstrating the utility of the relative positioning algorithms used in this approach. However, one application of relative navigation in orbit could be to help maintain the swarm's spatial structure using differential air drag [105], or solar radiation pressure [54], for example. This would require a degree of attitude control of individual femto-spacecraft, such as by using miniaturised magnetorquers to interact with the ambient magnetic field in

[1] In Chapter 3, $\mathbf{x}$ denotes a position vector, but here the state vector describes both position and velocity.

orbit [37]. Additionally, the deployment of larger femto-spacecraft networks could be enabled by multiple deployers. This analysis is limited to a single deployer spacecraft to study dispersal and evolution, as well as demonstrate navigation filters.

4.2 Swarm Dispersal and Evolution Strategies in LEO

In this section, swarm dispersal and evolution strategies in LEO from a single deployer spacecraft are presented. Only the ejection impulse is modelled as contributing to the swarm dispersal, in the absence of any perturbing forces affecting the swarm dynamics beyond the relative dynamics model. The swarm is composed of femto-spacecraft without any form of attitude or translational control (be this passive or active).

Two primary deployment methods are considered. The first approach is a controlled, sequential ejection of the swarm with fixed ejection velocities comprised of along-track and cross-track elements. This results in trajectories that both lead and trail the deployer, with cyclical displacement that includes radial motion. The second approach is to instantaneously scatter femto-spacecraft at a constant speed but in many (random) ejections around the deployer. This highlights the possible spatial structure a swarm could occupy at a given time. Both of these approaches could be used in applications such as sparse aperture interferometry or massively parallel sensing of near-Earth phenomena, as discussed in Chapter 1. In each scenario presented, the deployer is in a circular LEO of altitude 400 km. Both scenarios consider small ejection velocities of a realistic order of magnitude for ejecting a femto-spacecraft from a CubeSat deployer. The exact velocities are selected for a comparison with experimental work, described later in Chapter 6, and represent a demonstratory sample case for each scenario.

4.2.1 Sequential swarm ejection strategy

The sequential ejection of a swarm of femto-spacecraft is simulated over one Earth orbit relative to the deployer. The swarm is ejected along two relative trajectories, one leading and one trailing the deployer. Firstly, consider two femto-spacecraft, both initially coincident in position with the deployer:

$$\mathbf{r}_{0_1} = \mathbf{r}_{0_2} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{m} \tag{4.9}$$

and with ejection velocities given by:

$$\mathbf{v}_{0_1} = \begin{bmatrix} \dot{x_0} \\ \dot{y_0} \\ \dot{z_0} \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 4 \end{bmatrix} \times 10^{-3}\,\text{m}\,\text{s}^{-1} \tag{4.10}$$

$$\mathbf{v}_{0_2} = \begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ -4 \end{bmatrix} \times 10^{-3}\,\mathrm{m\,s^{-1}} \tag{4.11}$$

If these two femto-spacecraft are ejected at the same time, over one full Earth orbit for the deployer, their trajectories relative to the deployer would take the form as shown in Fig. 4.2.

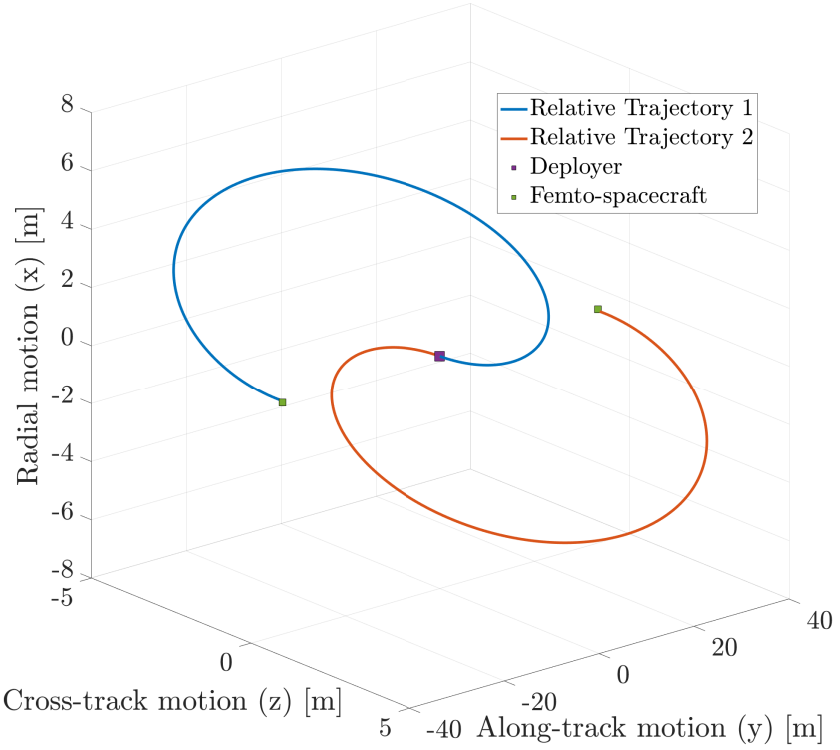


FIGURE 4.2: RELATIVE TRAJECTORIES OVER 1 ORBIT (SEQUENTIAL EJECTION)

This shows the trajectory of each femto-spacecraft over the course of the orbit, taking a helical form with radial, cross-track and along-track displacement from the deployer. This also shows the final position of each femto-spacecraft, which are at that time directly in front of and behind the deployer in the along-track sense. Note that radial (x-axis) motion is plotted vertically, inline with the CW reference frame shown in Fig. 4.1. By deploying the swarm sequentially and in even time steps over this orbit, the swarm members would be located at different phases along this same relative trajectory, as shown in Fig. 4.3. In this case, a swarm of 20 femto-spacecraft (10 on each relative trajectory) are deployed, which would then gradually drift farther from the deployer in following orbits, as shown in Fig. 4.4. With this type of deployment sequence, the swarm drifts away from its deployer, bounded only in the radial and cross-track directions, dispersing approximately ±50 m from the deployer after 1 orbit, to ±500 m from the deployer after 10 orbits.

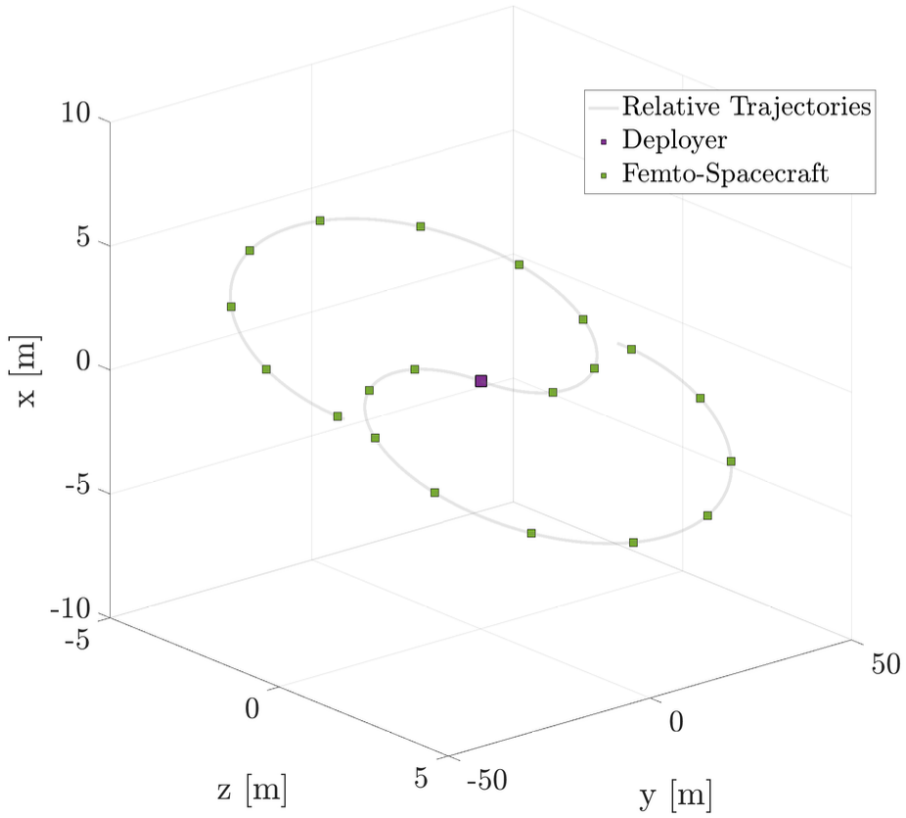


Figure 4.3: Swarm state at $t = T$ (sequential ejection)

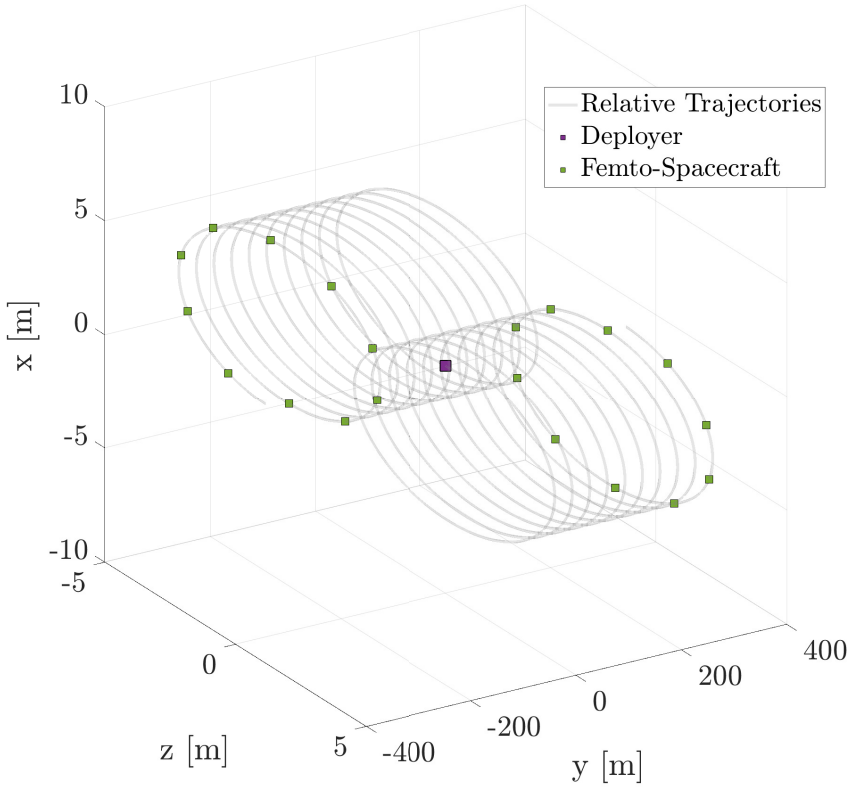


Figure 4.4: Swarm state at $t = 10T$ (sequential ejection)

4.2.2 Instantaneous swarm ejection strategy

Rather than a controlled sequential ejection of the swarm to drift away from the carrier, the swarm could be ejected instantaneously in random directions around the deployer at the same speed, with the initial state for the entire swarm given by:

$$\mathbf{r}_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{m} \qquad |\mathbf{v}_0| = 1 \times 10^{-3}\,\text{m}\,\text{s}^{-1} \tag{4.12}$$

This ejection is modelled with the velocity vectors pointing in uniformly distributed random directions around the deployer, each with the same magnitude. In practice, the available deployment directions would be limited by what the deployer spacecraft structure can permit due to both its geometry and attitude, but all directions are modelled here for generality. This approach is taken to demonstrate how the swarm would evolve in such a scenario. Figures 4.5-4.7 display a swarm of 20 femto-spacecraft deployed randomly around their deployer, shown at different stages throughout the first orbit.

After each relative orbit, as shown for the first orbit after ejection in Fig. 4.7, the swarm returns to a state where each femto-spacecraft is located predominantly either in front of or behind the deployer, drifting further in the along-track sense over the course of several orbits. Note that the discussion is limited to 20 femto-spacecraft here and in the relative navigation simulations for clarity in the figures presented. This helps highlight how the navigation system could work, but the approach is valid with larger numbers of femto-spacecraft as considered in Chapter 3.

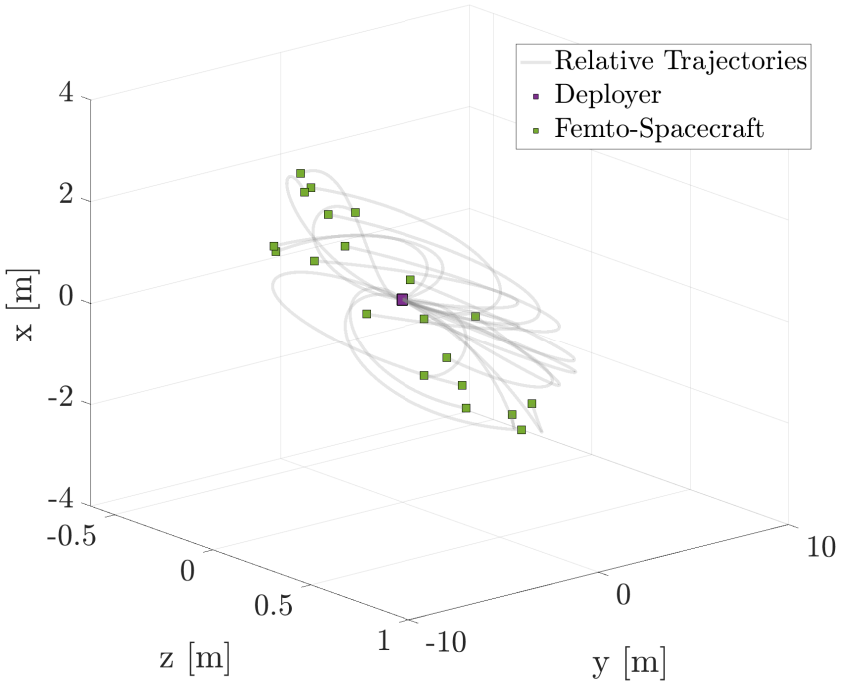


Figure 4.5: Swarm state at $t = \frac{1}{2}T$ (random ejection)

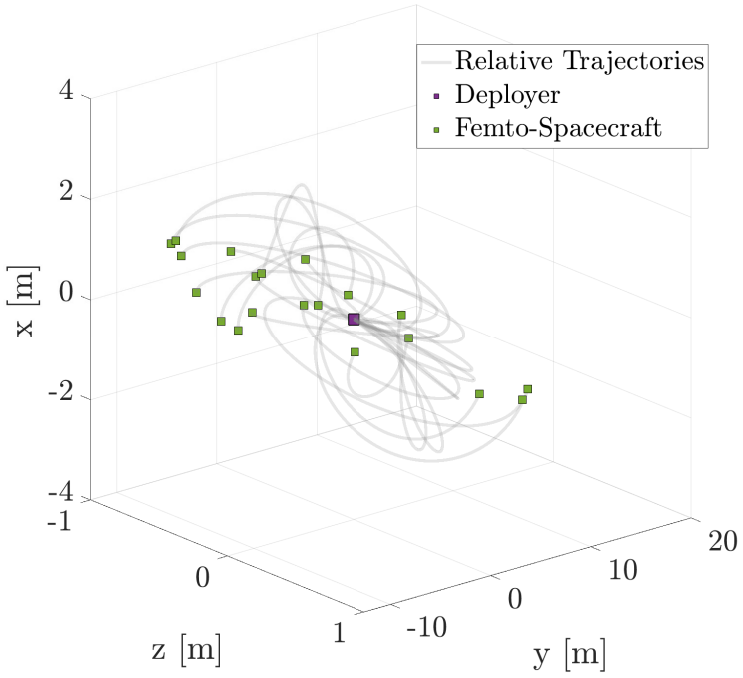


Figure 4.6: Swarm state at $t = \frac{3}{4}T$ (random ejection)

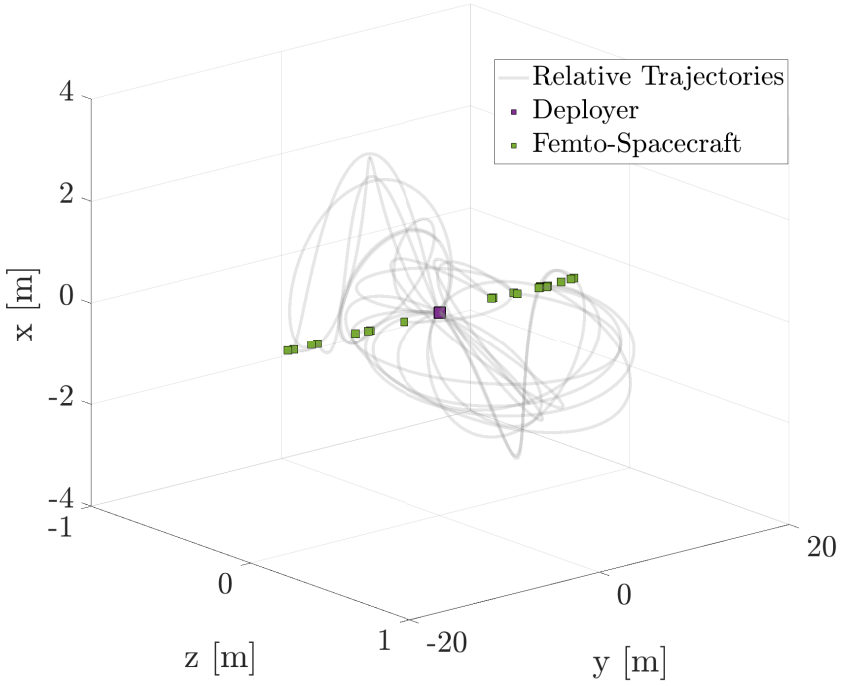


Figure 4.7: Swarm state at $t = T$ (random ejection)

For the random dispersal, it is seen that after each relative orbit the swarm returns to a state where each femto-spacecraft is generally located either in front of or behind the deployer. This expected behaviour could be incorporated into the relative navigation strategy as a calibrating step based on expected geometry and order of magnitude range estimates between nearby swarm members as a 'sense-check' for relative positioning performance. Primarily, this random

ejection technique highlights the available volume of space that the swarm could occupy relative to the deployer with time, as dynamically shifting 'clouds' for distributed sensing and other applications.

4.3 Relative Navigation Filtering

In this section, Kalman filter (KF) [106] techniques are used to implement a relative navigation system for the swarm. This is achieved by combining the algorithm outputs and the relative dynamics in the above scenarios to filter the relative state estimation over time. Two methods are presented for the above ejection strategy scenarios. Firstly, a traditional linear Kalman filter is used that treats the outputs of the relative positioning algorithm as a partial state observer. Secondly, a non-linear extended Kalman filter (EKF) is presented as an alternative approach which uses the range estimates directly as measurements after solving the initial relative orbit determination (IROD) problem using the relative positioning algorithms to initialise the filter. In all of the following scenarios, the case of a centralised filter operating on the deployer spacecraft using the SDP algorithm is considered.

4.3.1 Kalman filter

The discrete time model used for the relative navigation system is:

$$\mathbf{x}_k = \Phi \mathbf{x}_{k-1} + \mathbf{w}_{k-1} \tag{4.13}$$

$$\mathbf{z}_k = \mathbf{x}_k + \boldsymbol{\nu}_k \tag{4.14}$$

where $\mathbf{x}_k$ is the femto-spacecraft state at discrete time step k, Φ is the state transition matrix from the CW equations, $\mathbf{w}$ is the process noise vector, $\mathbf{z}$ is the state measurement and $\boldsymbol{\nu}$ is the observation noise vector. Process and observation noise is modelled as zero-mean additive white Gaussian noise (AWGN). Process noise accounts for model approximations and integration errors, while observation noise accounts for measurement errors.

As the femto-spacecraft are modelled as passive units, there is no control input in this model, but this could be added to Eq. (4.13) to model translational control inputs. In this model, the state measurement $\mathbf{z}$, representing the relative position estimate of a femto-spacecraft, is provided directly from sampling the relative positioning algorithm, the output of which provides the swarm position estimates at each time step. Without velocity measurements, this is therefore a partial linear observer of the femto-spacecraft state in the measurement model. In this implementation, the algorithm is sampled at every discrete time step, with the swarm relative position output by the SDP algorithm providing the input for the filter's measurement update at each step.

The filter iteratively works through two stages. This begins with an initial state estimate $\hat{\mathbf{x}}_{k-1}^{-}$ and state covariance estimate P_{k-1}^{-} from the initial conditions. Firstly, the prediction (or time

update) stage, using the relative dynamics model:

$$\hat{\mathbf{x}}_{\mathbf{k}}^{-} = \Phi \hat{\mathbf{x}}_{\mathbf{k-1}}^{+} \tag{4.15}$$

$$P_k^- = \Phi P_{k-1}^+ \Phi^T + Q_{k-1} \tag{4.16}$$

where the superscripts denote the predicted (-) and measurement updated (+) estimates,[2] and Q is the process noise covariance matrix. Secondly, the measurement update stage:

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{4.17}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K_k(\mathbf{z}_k - H\hat{\mathbf{x}}_k^-) \tag{4.18}$$

$$P_k^+ = (I - K_k H) P_k^- \tag{4.19}$$

where K is the Kalman gain, H is the observation matrix, and R is the measurement noise covariance matrix. Note that in this case H is an identity matrix. The filter then iterates through $k-1 \leftarrow k$, using the output of the measurement update stage as part of the calculation in the next time update stage. As described in Chapter 3, it can be expected that there are time steps where individual femto-spacecraft fail to localise and the algorithm produces no estimate. In these cases, the filter uses the predicted estimate instead of the measurement updated one. In this implementation, for simplicity, Q is modelled as a diagonal matrix with the following elements for the CW equations [107]:

$$Q = \begin{bmatrix} Q_r & 0 \\ 0 & Q_v \end{bmatrix} \tag{4.20}$$

where:

$$Q_r = \begin{bmatrix} 2^2 & 0 & 0 \\ 0 & 2^2 & 0 \\ 0 & 0 & 2^2 \end{bmatrix} \times (10^{-2}\,\mathrm{m})^2 \qquad Q_v = \begin{bmatrix} 2^2 & 0 & 0 \\ 0 & 2^2 & 0 \\ 0 & 0 & 2^2 \end{bmatrix} \times (10^{-3}\,\mathrm{m\,s^{-1}})^2 \tag{4.21}$$

where the values used here are intended to compensate for the additional process noise due to the linearisation. Additionally, the measurement noise R is modelled with the following elements:

$$R = \begin{bmatrix} R_r & 0 \\ 0 & R_v \end{bmatrix} \tag{4.22}$$

where:

$$R_r = \begin{bmatrix} 15^2 & 0 & 0 \\ 0 & 20^2 & 0 \\ 0 & 0 & 20^2 \end{bmatrix} (\mathrm{m})^2 \qquad R_v = \begin{bmatrix} 1.5^2 & 0 & 0 \\ 0 & 0.5^2 & 0 \\ 0 & 0 & 0.5^2 \end{bmatrix} \times (10^{-2}\,\mathrm{m\,s^{-1}})^2 \tag{4.23}$$

[2] in literature, these are often referred to as *a priori* and *a posteriori* estimates

The relative navigation of the swarm would begin after release from the deployer. Ejection from the deployer would provide the initial state estimate of each femto-spacecraft within the swarm. The relative velocity at the point of ejection would then introduce uncertainty in practice. This system is now applied to both swarm ejection scenarios to demonstrate relative positioning performance with the SDP algorithm. In this simulation, an AWGN level of $\sigma_r = 0.02\bar{r}$ is set, with a sampling interval of 1 minute. Figure 4.8 displays filtering results from the sequential ejection of 20 femto-spacecraft over the course of an orbit (where those deployed last are re-tracing the relative trajectories of those deployed first), with initial conditions from Eqs. (4.9) - (4.11). In the deployment sequence, 10 femto-spacecraft are ejected in front of and 10 are ejected behind of the deployer.

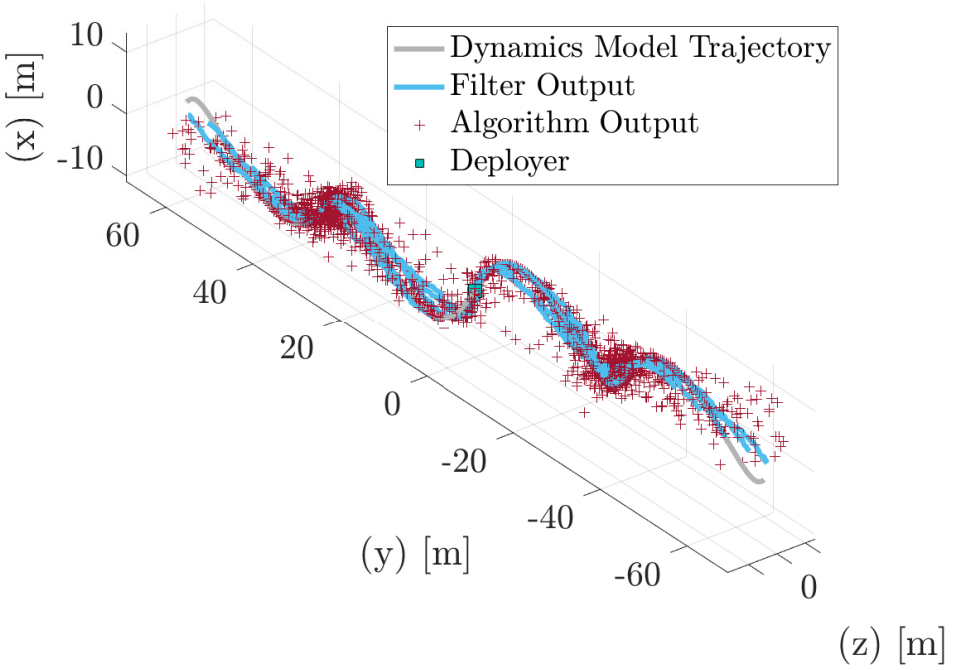
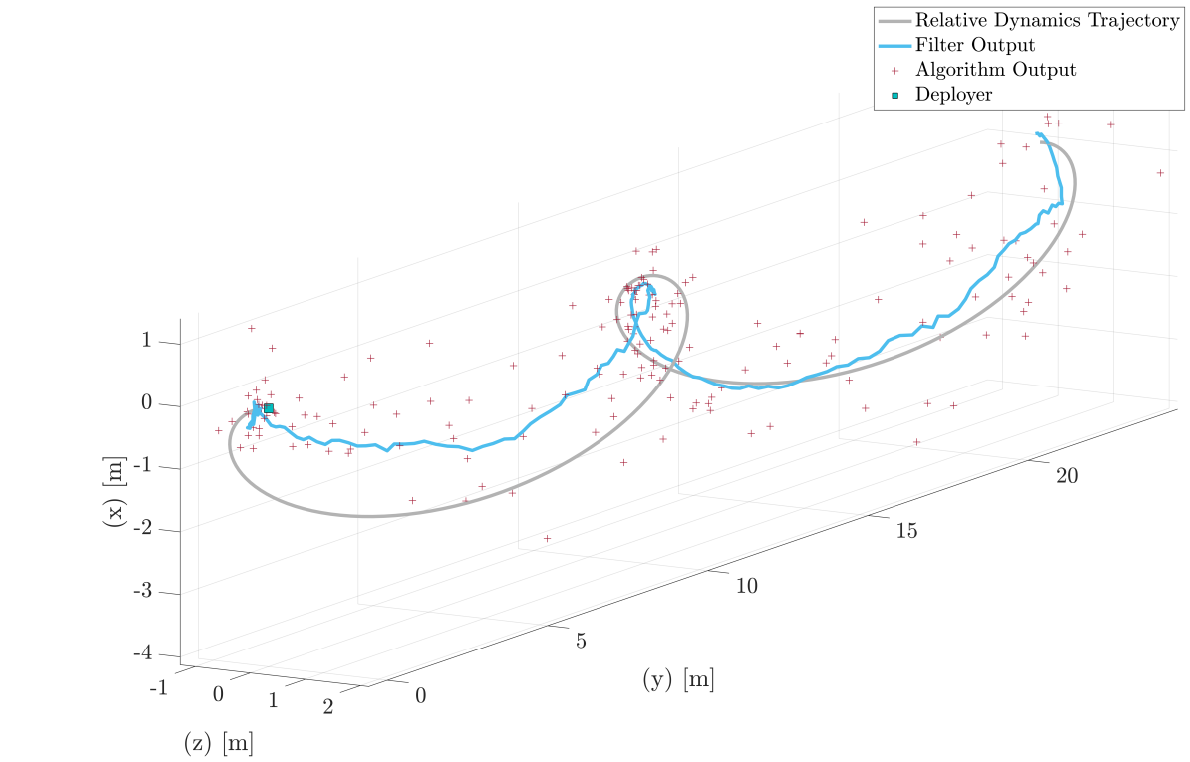
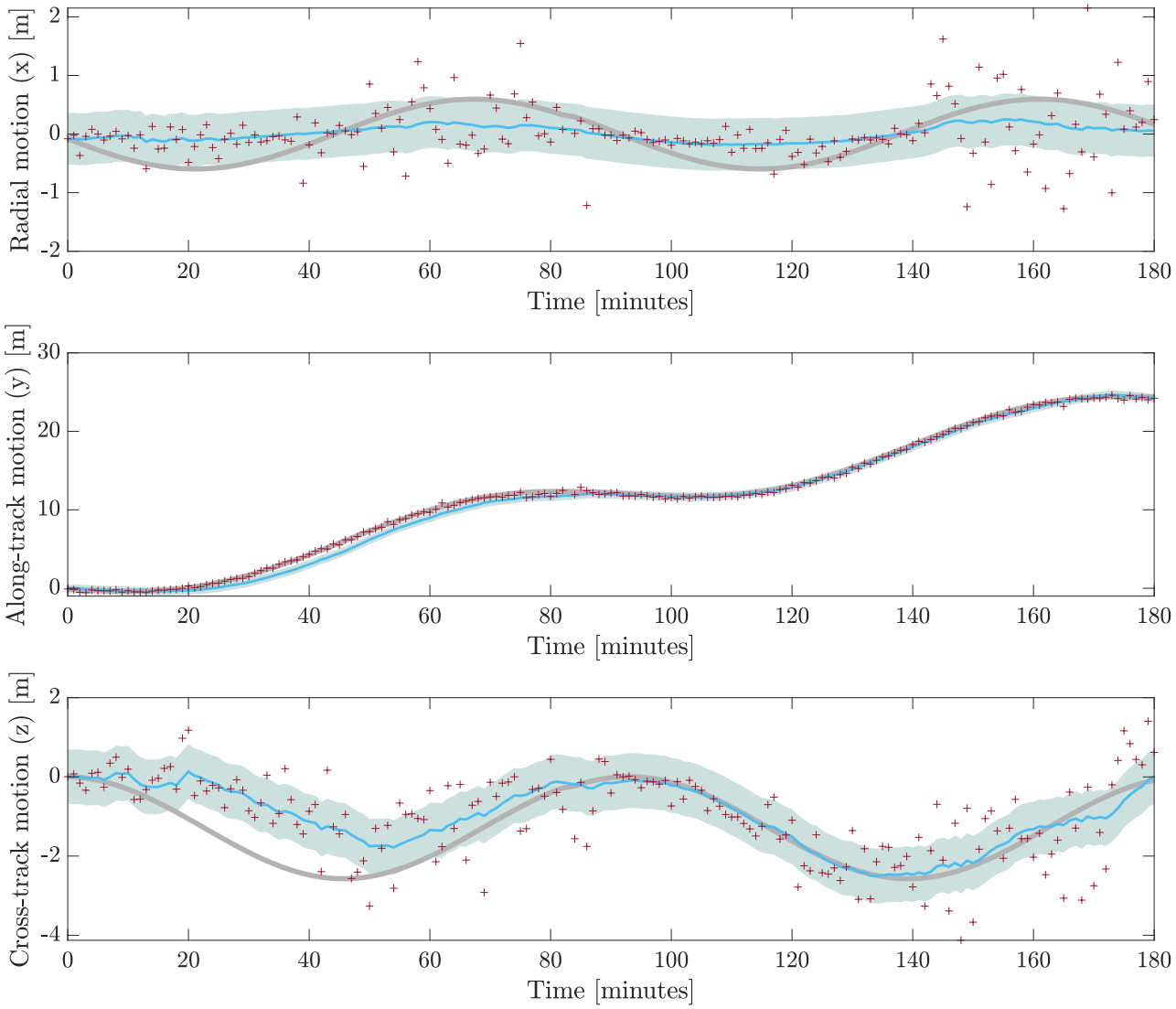


Figure 4.8: Swarm relative navigation (sequential ejection)

The filtering in Fig. 4.8 occurs between the first and second orbit after all femto-spacecraft have been deployed over the course of one orbit. The algorithm output samples are marked by the red crosses, while the blue lines indicate the Kalman filter output and the grey lines indicate the dynamics model of the trajectories for each femto-spacecraft from the deployer, which is located at the origin. This shows the ability of the filter to smooth out discrete algorithm samples of position for all the femto-spacecraft and track the swarm's position over this orbit. This also shows similar performance for all 20 femto-spacecraft. However, the figure is busy due to the overlap of the trajectories. To analyse the filter performance in detail, and for clarity, the filtering of just one femto-spacecraft within the swarm is isolated over the course of 2 orbits (approximately 190 minutes), as shown in Figs. 4.9a and 4.9b.



(a) 3D view



(b) Individual axes view

Figure 4.9: Relative navigation of a femto-spacecraft (sequential ejection)

The algorithm output samples are again marked by the red crosses, indicating the sampled algorithm outputs at each discrete time step for this particular femto-spacecraft's position, while the blue line indicates the Kalman filtering of these samples. As can be seen in the plots of both the three-dimensional relative motion and the individual axes, the femto-spacecraft's relative position state estimate with time is tracked well by the algorithm and smoothed by the filter, with the filter improving the position estimate based on previous measurements of position and knowledge of the dynamics of the system.

Figure 4.10 displays the filtering of 20 femto-spacecraft deployed randomly and instantaneously from a deployer (for clarity, this is only shown up to $t = \frac{T}{5}$).

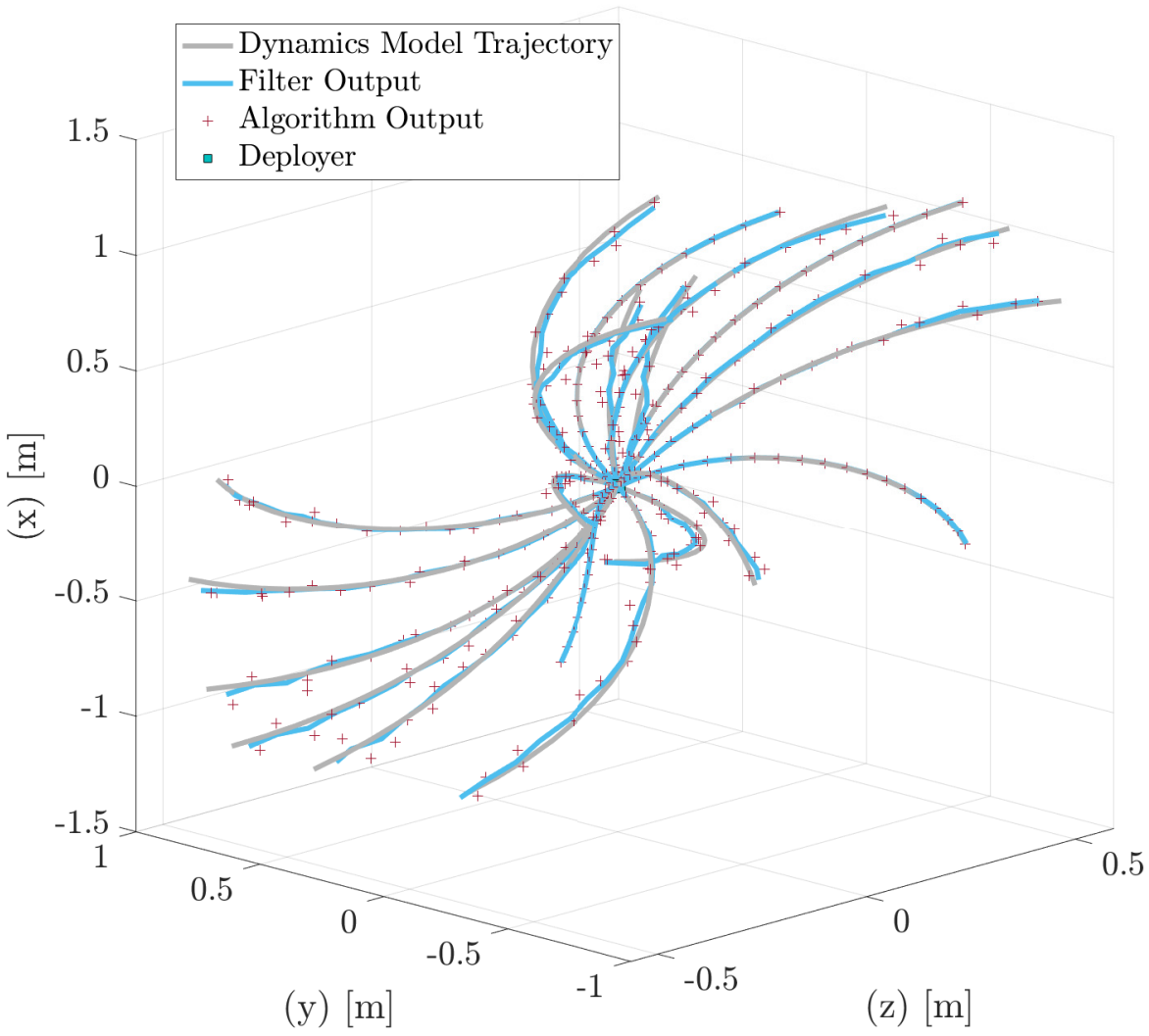
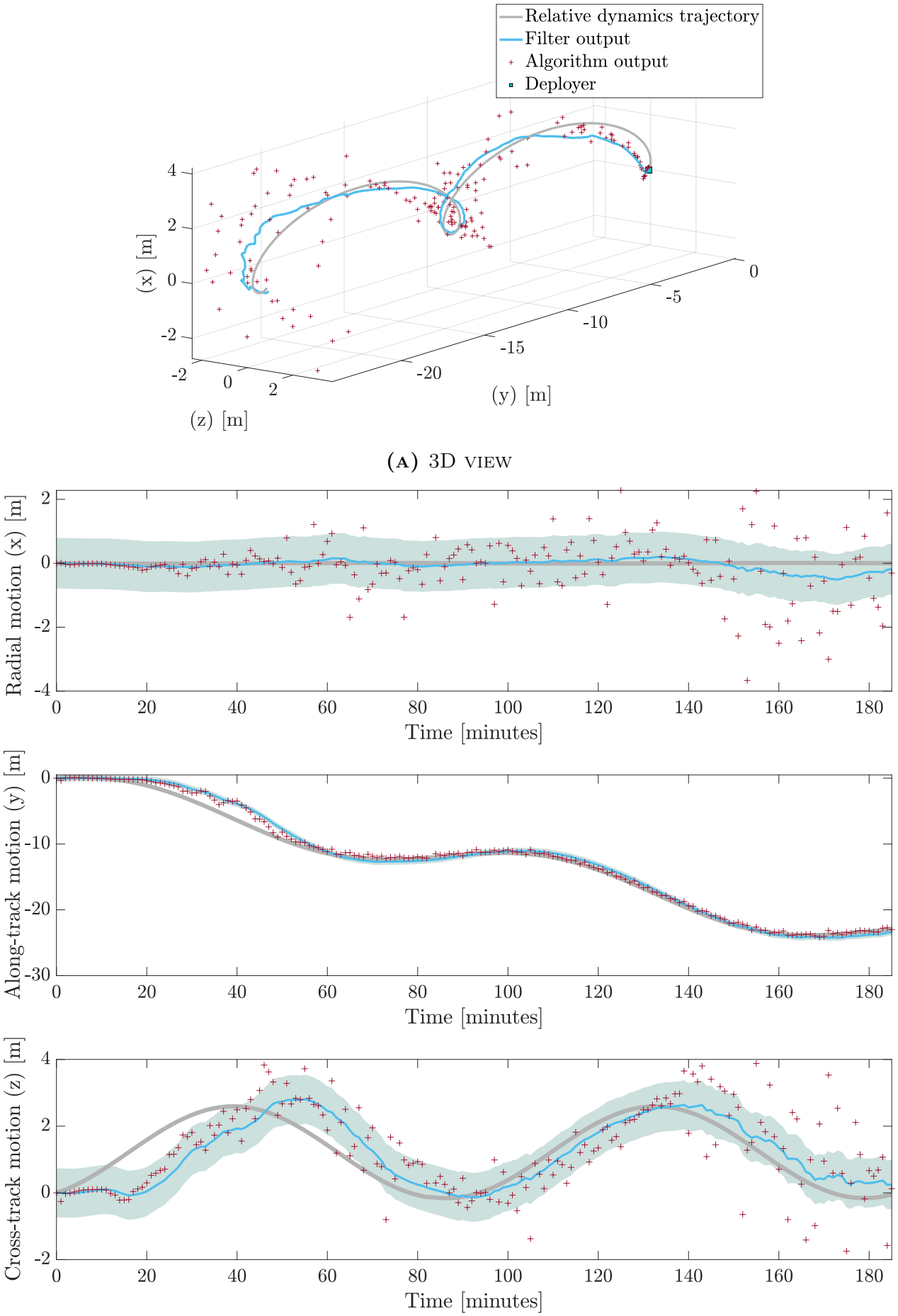


Figure 4.10: Swarm relative navigation (random ejection)

This case demonstrates similar filter performance as the sequential ejection. Again, isolating one femto-spacecraft within the swarm over the course of 2 orbits, the performance can be analysed as shown in Figs. 4.11a and 4.11b.



(a) 3D view

(b) Individual axes view

Figure 4.11: Relative navigation of a femto-spacecraft (random ejection)

Both the three-dimensional relative motion and the individual axes plots again demonstrate that the femto-spacecraft's relative position state estimate with time is improved using the filter over relative positioning outputs alone at discrete sample times. Note that the x-axis displacement is much smaller than the other axes in this case.

4.3.2 Extended Kalman filter

As an alternative approach, the outputs of the positioning algorithms can be used for initial relative orbit determination (IROD). With this method, the relative positioning algorithm is used only to initialise the filter state, providing a starting swarm position state estimate for an extended Kalman filter (EKF) that uses range estimates directly to further update the swarm state estimates with time. This method requires an EKF only because such a range-based measurement model is now non-linear.

This approach is presented here as an alternative to the standard linear KF, but the suitability of either method would be an open design option for practical implementation. For example, this approach could be used in larger networks to be less computationally intensive for the deployer spacecraft for most of the time in operation. An algorithm's relative position estimates could be used periodically for calibration or re-initialisation.

To implement this approach, the state measurement model from Eq. (4.14) can be adapted for the EKF to form:

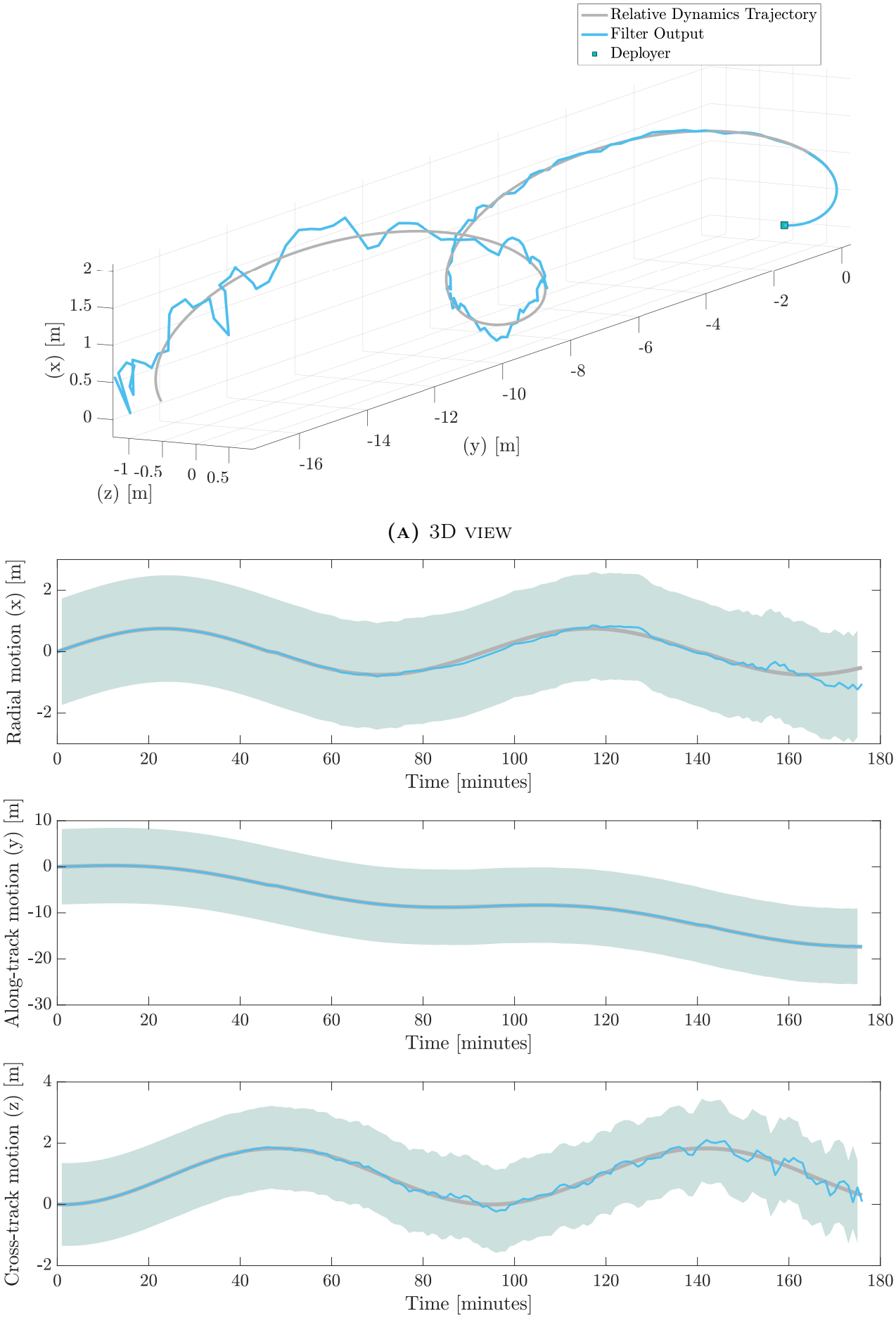
$$\mathbf{z_k} = h(\mathbf{x}_k) + \boldsymbol{\nu}_k \tag{4.24}$$

where h is an observation function of the state, and the observation matrix H is now defined by the following Jacobian:

$$H_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\hat{x}_k^-} \tag{4.25}$$

The SDP algorithm is used to initialise the swarm state for the filter, with the range estimates then being used directly via the observation matrix H, which unlike in the linear KF is no longer an identity matrix in this case. Now, in a centralised implementation with full communication links between the femto-spacecraft, H contains all the partial derivatives for the Jacobian. The state measurement $\mathbf{z}$ is no longer position outputs from the algorithm, but now contains all the range estimates provided from networking. The femto-spacecraft state is updated using these new parameters, just as with Eq. (4.18), to determine the measurement residual. Apart from these modifications, the EKF is implemented in simulation in the same way as the KF with the stages in Eqs. (4.15) - (4.19).

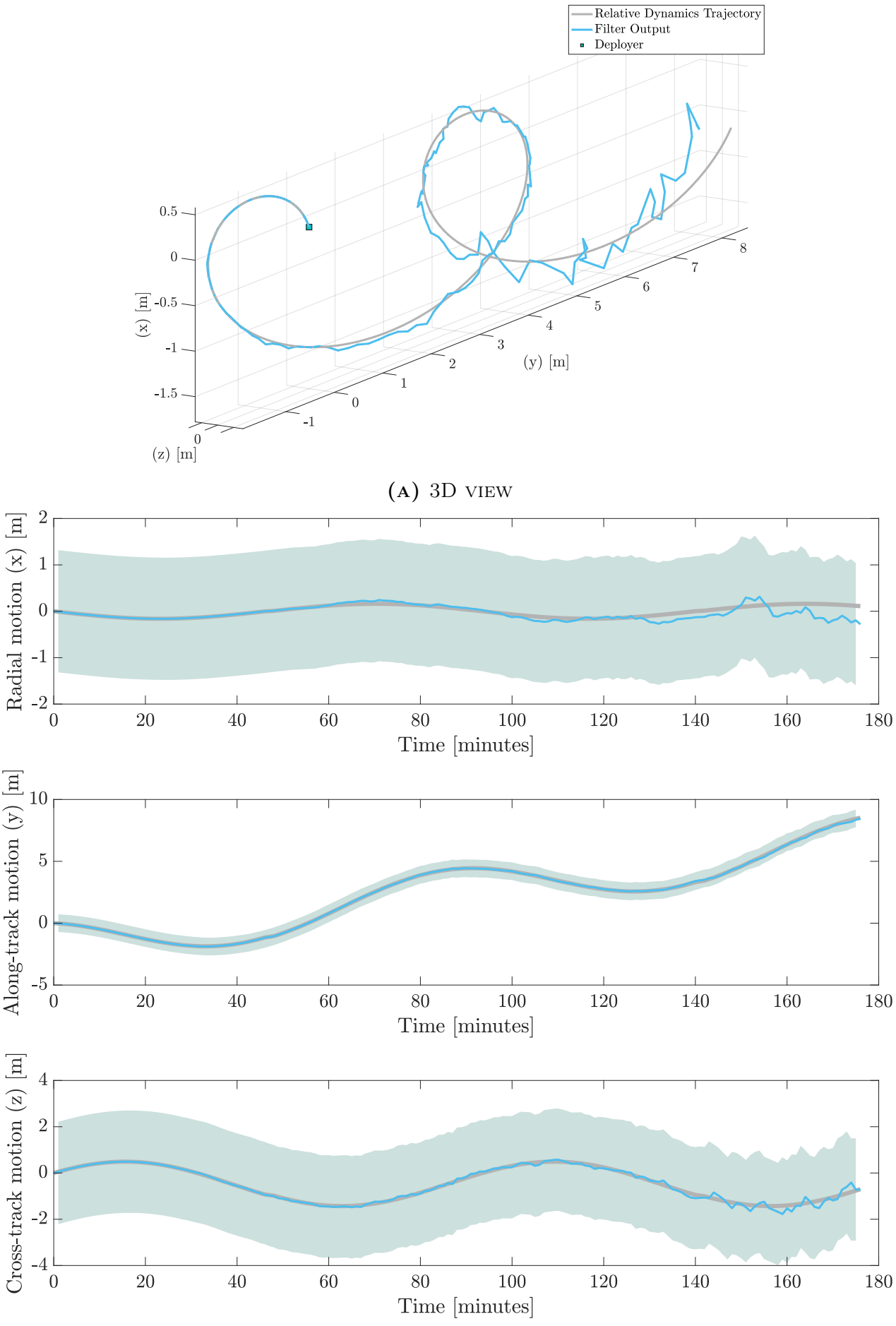
The outputs of this filtering approach for both ejection strategies are displayed in Figs. 4.12a and 4.13b, using the same simulation parameters. This highlights the filtering of one of the femto-spacecraft within the swarm over 2 orbits for clarity.



(a) 3D view

(b) Individual axes view

Figure 4.12: EKF Relative navigation of a femto-spacecraft (sequential ejection)



(a) 3D view

(b) Individual axes view

Figure 4.13: EKF Relative navigation of a femto-spacecraft (random ejection)

The results of both the KF and EKF systems implemented in this section demonstrate the improved relative navigation that Kalman filtering provides over relative positioning algorithm outputs alone. Combining confidence in the algorithm sampling and the relative dynamics provides improved relative positioning when compared to discrete algorithm samples at each discrete time step. In the EKF case, where the algorithm is used for initialisation, the results show that using a range-based measurement model is also viable. This could provide an alternative approach that requires less computation at each discrete time step. Different sampling rates, particular approaches, and different ejection velocity strategies are left as a point of consideration for future implementation for specific mission scenarios and applications. As a femto-spacecraft swarm drifts further from its deployer, the uncertainties in the state estimation would grow with time, so this filtering is important in bounding errors and providing improved relative positioning for a swarm. The results demonstrate that either of these approaches could be implemented for a centralised navigation filter for swarm relative positioning. Depending on operational constraints, such as available processing power, one method or a combination of both KF and EKF methods could be implemented in practice.

4.4 Chapter Summary

This chapter has presented a relative dynamics model for femto-spacecraft swarm motion in LEO, considered ejection strategies and has presented relative navigation systems using Kalman filtering techniques. These methods have been demonstrated with the ejection strategies presented. The relative dynamics model used shows how a swarm, initially contained within its deployer, could be ejected in a controlled and sequential or instantaneous manner into orbits neighbouring the deployer. This motion could help inform sensing applications and use cases for swarms of femto-spacecraft in orbit. Both strategies presented highlight the principal ways this could be achieved in LEO.

The navigation filters presented offer improvements over relative positioning algorithm samples at discrete time steps. This is only possible when combining the relative motion of femto-spacecraft from the dynamics model using known ejection velocities. In real-time relative navigation, this helps to smooth out errors and reduces the discrete effect of measurement noise and inaccuracies in position estimates.

Chapter 5

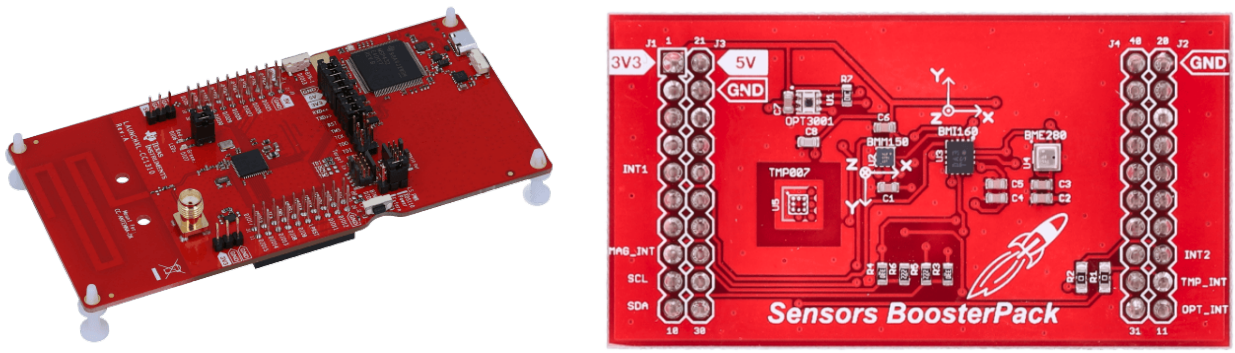
Network Implementation

This chapter bridges the gap between simulation and experimental demonstration of relative navigation. It presents the development of a fully interconnected network protocol designed and tested to operate on many femto-spacecraft. The network protocol is implemented through embedded software development in C, using small development kits as proxies for femto-spacecraft. Firstly, the experimental technology used is detailed in terms of the hardware and software. Then, the implementation of the network protocol is detailed, presenting an address filtering and sequential transmission technique, tested extensively for reliable operations. This is described in the context of accumulating network RSSI data for use in the relative positioning algorithms presented in Chapter 3. The conversion of this data into range estimates necessary for inputs to the algorithms is discussed in Chapter 6. The approach outlined here can be applied in general for sharing other data (such as other range measurements) throughout the network. Finally, modifications to this protocol to support alternative network communications topologies and the integration of the sensor and inertial measurement unit (IMU) data that would be available on femto-spacecraft are also detailed.

5.1 Experimental Technology

In this section, the experimental technology used to develop and implement the network protocol is outlined, in terms of both the hardware and software utilised. The following descriptions explain the development kits used as femto-spacecraft proxies and the software development tools implemented.

Texas Instruments (TI) SimpleLink™ sub-1 GHz CC1310 wireless microcontroller (MCU) LaunchPad™ development kits were used as proxies for femto-spacecraft hardware for experimental testing (Fig. 5.1a) [108]. This development kit was chosen because its MCU is representative of femto-spacecraft technology in terms of computing power available at the femto-spacecraft mass and centimetre length-scale, and it has an integrated radio module for wireless communications. The CC1310 MCU is also currently implemented in a printed circuit board (PCB) femto-spacecraft design in development within the Space and Exploration



(a) TI CC1310 development kit (b) Sensors Boosterpack

Figure 5.1: Experimental hardware used as a femto-spacecraft proxy (image credits: TI)

Technology (SET) group at the University of Glasgow. This means that the developments in this Chapter will be compatible with femto-spacecraft platforms to support ongoing research. The development kit has 128 KB of in-system programmable flash memory and 8 KB of static random-access memory (SRAM). Additionally, this MCU specifically targets low-power and long-range radio communications for wireless applications. The development kit has an integrated PCB trace antenna, but also a sub-miniature version A (SMA) port for connecting external antennas. For experimental demonstration, V-dipole antennas were attached to this port to provide a more omni-directional radiation pattern than the integrated PCB trace antenna. This will be discussed later in Chapter 6.

The CC1310 radio module supports operation at a radio frequency of 868 MHz, which is a license-free industrial, scientific and medical (ISM) band, at a maximum transmitting power of 14 decibel-milliWatts (dBm). This radio module supports half-duplex communications, meaning that development kits can both receive and transmit radio packets, but not simultaneously. This necessitates the development of a specific network protocol that enables full interconnection in a network of multiple femto-spacecraft in any direction as required, without simultaneous transmissions. In this work, full interconnection is considered to be the case where any femto-spacecraft can both receive from and transmit to any other femto-spacecraft in the network. Decentralisation is a straightforward extension to this approach and is discussed in Section 5.2. Distributed network protocol development is left as a topic of future work, owing to the scale of the testing and implementation in this work, which was limited to under 25 development kits.

Additionally, the TI BOOSTXL-SENSORS BoosterPack™ plug-in module [109], shown in Fig. 5.1b, was attached to each development kit to provide 9-axis IMU (3-axis accelerometer, gyroscope and magnetometer) support (the Bosch BMI 160) and a variety of sensors to measure pressure, ambient temperature, humidity, and ambient light. These booster packs attach directly to the plug-in connector pins by stacking on top of the development kit. This MEMS sensor suite compliments the development kits in using them as a femto-spacecraft test bed, and could be used for further testing outside the scope of this work, such as inertial attitude determination and experimental demonstration of distributed sensing applications [9], as dis-

cussed in Chapter 7. Modifications to the overall network protocol to support the sharing of IMU and sensor data are detailed in Section 5.2.3. The datasheets for the development kit [110] and the booster pack [111] detail the complete technical specifications.

Embedded software development was programmed using TI Code Composer Studio (CCS). CCS is an integrated development environment (IDE) used to develop and test embedded software applications onto TI MCUs. All software was developed in C using the real-time operating system (RTOS) TI-RTOS [112], making use of its supported functionality for this application, which focuses on real-time network communications. The TI SimpleLink CC13x0 software development kit (SDK) [113] is used for specific MCU functionality interfacing. Within this SDK, the TI EasyLink™ radio frequency (RF) application programming interface (API) was used to help develop the custom network protocol. This provides a layer of abstraction in the programming above commanding the RF driver itself. In particular, this means that radio commands (e.g. transmit, receive, etc.) can be programmed using inbuilt functions within the main network embedded program, along with common parameter definitions such as setting the frequency and transmit power, and a radio packet structure that can be shared by every development kit on the network. Several modifications are made to the default Easylink™ API parameter definitions to enable this custom protocol implementation, as detailed in Section 5.2.

For developing IMU and sensing extensions onto the network protocol, the Simplelink™ sensor and actuator interface layer (SAIL) [114] was used as a plug-in added to programs with this feature. SAIL provides software module support for interfacing all equipment on the sensor booster packs used. Details of the implementation of this data into the network protocol is given in Section 5.2.3. Compatibility issues were encountered using the latest stable releases of software development tools, specifically with the SDK and SAIL plug-in. As such, version control was found to be important in testing reliable operations of in-built functionality. Once a function was verified to operate reliably, versions were not modified. All software developed in this work was implemented using the following versions and specifications: CCS version 11.0 running on macOS, TI-RTOS 6, Simplelink™ cc13x0 SDK version 4.20.01.03, and Simplelink™ sensor and actuator plugin (SAIL) version 1.40. The modifications described in this chapter may be achieved using similar approaches with the latest versions of these packages. However, the SDK and SAIL versions listed here were only found to be compatible through trial and error of using recent (but not the latest) releases to support the functionality required, especially for in-built sensor functions with the sensor booster packs. All versions of the cc13x0 SDK are currently available online [115], and within CCS (using the online resource explorer) for the SAIL plug-in.

5.2 A Fully Interconnected Network Protocol

Firstly, a protocol for a fully interconnected network topology is developed specifically for accumulating RSSI data from and between all development kits on the network. In this section, the development and implementation of this protocol through embedded programming and

testing on hardware is detailed. The discussion is initially presented from the perspective of sharing RSSI data between femto-spacecraft as an operational process, and then for the development kits for implementation in software. The protocol supports extensions for other data types and so could be used with other intended range metrics and sensor data as described in Section 5.2.3.

In this work, the fully interconnected network protocol is designed to be used with centralised processing, that is, where a main spacecraft in the network handles the entire network data, and other nodes simply send and receive data with little computational loads. In a centralised configuration, network data is primarily accumulated at one central node, which is also intended as the node that does all central processing of the data. As discussed in Chapter 3, wireless networks can be broadly categorised into either centralised, decentralised, and distributed systems in terms of both their communication topology and the general flow of data for computational purposes. For instance, in the centralised relative positioning algorithm developed in Section 3.2, range data flows to one central node for computation, however a high degree of inter-spacecraft connectivity is assumed (and required) in order to make this work. For distributed networks, which are not tested experimentally in this work due to the small numbers of devices used, there would still be a need for a high degree of interconnectivity, and so modifications to the approach here could be made to enable this.

5.2.1 OPERATIONAL PROCESS

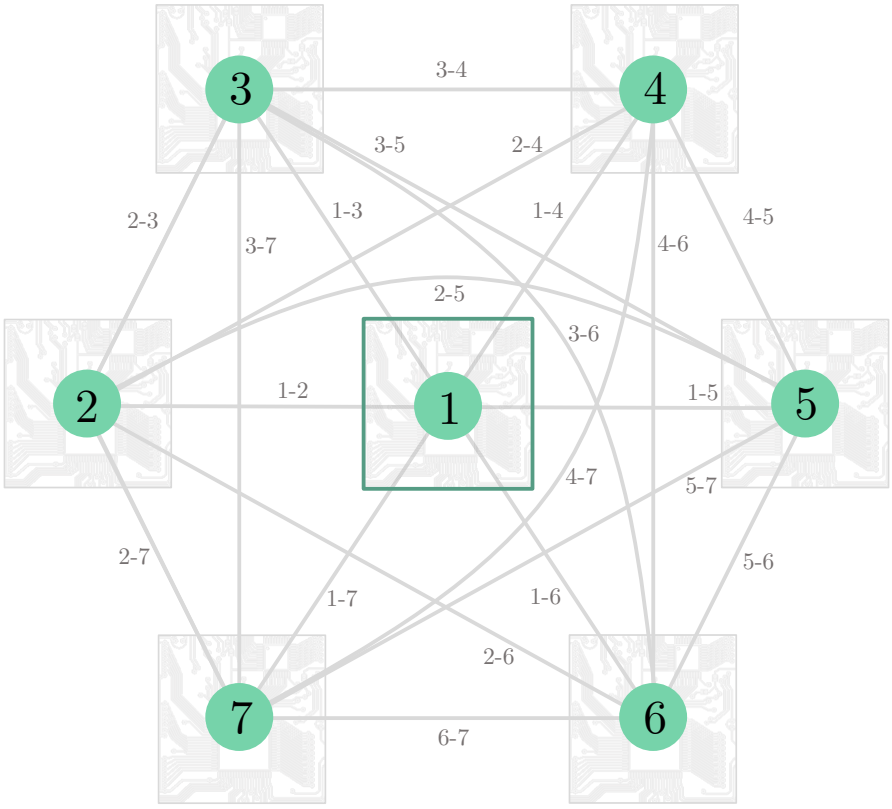


FIGURE 5.2: FULLY INTERCONNECTED NETWORK

For a fully interconnected network of n femto-spacecraft, as shown in Fig. 5.3, n successful

transmissions are sufficient to provide all femto-spacecraft with communicated data between all possible $\frac{n(n-1)}{2}$ undirected communication links. Alternatively, $(n-1)$ transmissions are sufficient to provide this data to one femto-spacecraft within the network as is typically used for testing purposes in this thesis.

This is possible if each femto-spacecraft sequentially transmits while the other $(n-1)$ femto-spacecraft receive and store incoming data. This is enabled by programming each femto-spacecraft to remain in a receive state until it is their turn to transmit. This is achieved using an address filtering technique, whereby each femto-spacecraft transmits using a different address, making it possible for each femto-spacecraft to be uniquely identifiable to others in the network. When a femto-spacecraft receives a radio packet, it therefore knows which femto-spacecraft sent it.

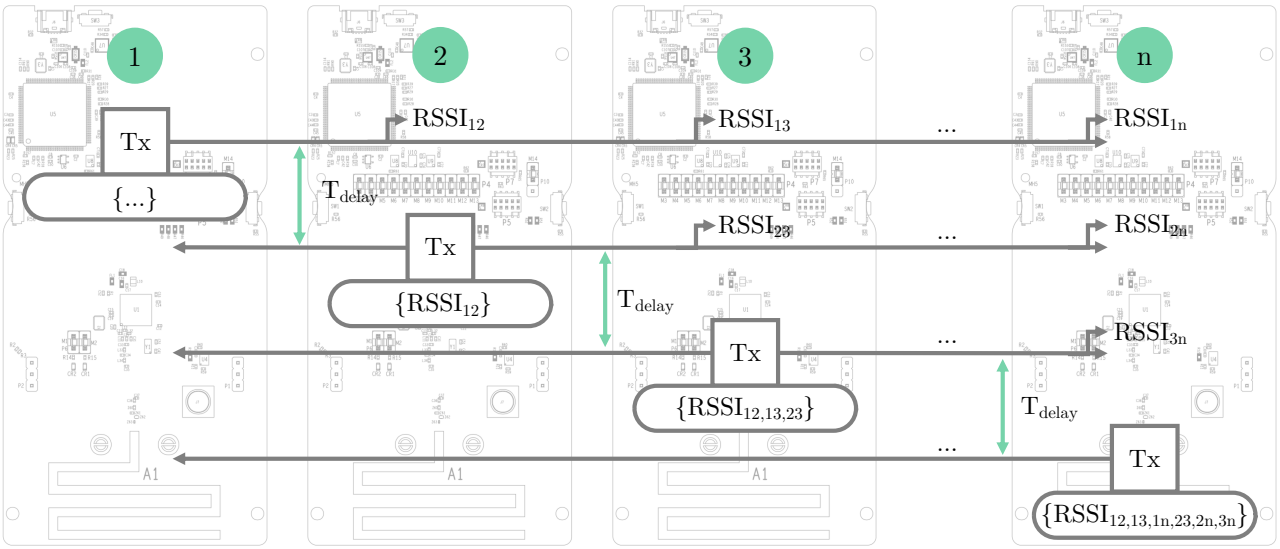


FIGURE 5.3: WIRELESS DATA FLOW IN THE RSSI GATHERING PROTOCOL

As shown in Fig. 5.3, femto-spacecraft 1 starts the communication sequence by transmitting a radio packet with an empty payload ($\{...\}$) to all other $(n-1)$ femto-spacecraft in the network. As this packet is received, it generates $RSSI_{12}$,$RSSI_{13}$,..., $RSSI_{1N}$ for those femto-spacecraft to store, respectively, where $RSSI_{ij}$ denotes the received signal strength indication received at femto-spacecraft j from femto-spacecraft i. Then, femto-spacecraft 2 is able to transmit a radio packet containing $RSSI_{12}$ over the network. Femto-spacecraft 3 then transmits $RSSI_{12,13,23}$, and so on, until femto-spacecraft n has received all $\frac{n(n-1)}{2}$ RSSI values between all n femto-spacecraft in the network. In this implementation, femto-spacecraft n then transmits all network RSSI data back to femto-spacecraft 1, which starts the process over again.

When femto-spacecraft 1 transmits, it sets a timer of nT_{delay} before transmitting again, where T_{delay} is a controllable delay period. When femto-spacecraft 2 receives a packet from femto-spacecraft 1, it waits T_{delay} before transmitting. Femto-spacecraft 3 repeats this process upon receiving from femto-spacecraft 2, and so on. This timing sequence can be modified by changing the delay period. If this sequence is interrupted by, for example, an unresponsive femto-

spacecraft, then no data will be received that cycle, and the process would restart after a time of nT_{delay}.

There are several points to note about this overall process. For simplicity, it is assumed that communication links are undirected (i.e. that $RSSI_{ij} \equiv RSSI_{ji}$). Otherwise, the number of communication links required would be doubled. In further testing (or operationally), directed communications could be used to check and reject new RSSI data for the positioning calculations, but this was not considered here, due to the high refresh rate set. This would be useful in cases where a recent $RSSI_{ij}$ measurement is significantly different to a recent $RSSI_{ji}$. Secondly, using RSSI data in this way, each femto-spacecraft must transmit at the same power (or at a minimum, transmit at a power known to every other femto-spacecraft), so that the fraction of that power received by each femto-spacecraft is properly correlated to range. In testing, all development kits transmit at a power of 14 dBm.

5.2.2 Implementation in software

The operational process flow was realised in software by building upon the TI 'rfEasyLinkEcho' project [116], which was modified to accommodate the network protocol with address filtering and pre-programmed delay periods. The original project demonstrates bi-directional communication between 2 development kits by having one transmit a packet, and another re-transmit (or 'echo') that same packet back. Substantial modifications were made to enable full interconnectivity between a network of n development kits within a given time interval. Specifically, each development kit was given a unique transmission address, and given an address checking procedure, enabling each development kit to know what data to store and when to transmit. This relies on each node being identifiable by its transmitting address, which is not a limitation, but for dynamically adding nodes to larger networks would likely require a process of identification and synchronisation throughout the network to add nodes that follow this protocol in an ad-hoc fashion. This could be programmed as an identification process.

Table 5.1: Address filtering structure for the network

Femto-spacecraft	Tx address	Rx address list
1	0x01	0xFF
2	0x02	0x01
3	0x03	0x{01, 02}
4	0x04	0x{01, 02, 03}
⋮	⋮	⋮
255	0xFF	0x{01, 02, 03, ...,FE}

Table 5.1 shows the transmit and receive address structure. Note that these addresses are given in hexadecimal. Each development kit has its own unique transmit address, and a list of addresses that it can receive from (by being pre-set to perform an action once hearing from that specific development kit). Also note that every development kit could be set to use received

data from any other development kit on the network. For the purposes of this testing and with the assumption of undirected communications, each development kit only utilises received radio packets from preceding development kit in the network to retrieve relevant RSSI data. Additionally, as testing is limited to 23 development kits, there is no need to go above an 8-bit address identifier, but this can be supported up to 64-bits to implement a network with more than 255 femto-spacecraft.

In each development kit's embedded program, it can therefore operate to either store incoming data in its local memory or transmit this data. The following pseudocode presents the straightforward implementation of address filtering, shown for a minimal example as programmed for development kit 3 to highlight the receive or transmit decision:

Algorithm 5.1 Address Filtering

Inputs: Rx Packet

Enter receive packet callback function:
if *packet received* **then**
    check Rx packet address and save as $Rx\_address$
    if $Rx\_address$ = 0x01 **then**
        store $RSSI_{13}$ in transmit packet payload element 1
        remain in receive state
    end if
    if $Rx\_address$ = 0x02 **then**
        store $RSSI_{23}$ in transmit packet payload element 3
        switch to transmit state
        save Rx packet reception time
        transmit packet
    end if
end if
post semaphore signalling to other tasks that the callback task is complete

This is placed within a callback function which is entered after receiving a radio packet. A callback function is one that is called from another function and can use the calling function as a parameter when triggered by an event, in this case, when a packet is received. The address of the received packet is checked and used to identify what element the development kit's transmitting packet payload is written to. This determines whether the development kit remains in a receive state when it receives a packet from development kit 1, or to transmit when it receives a packet from development kit 2. Whenever a development kit receives a new packet from a development kit it has already received from, the according payload elements are overwritten. A semaphore function is then used to signal that the development kit's task is complete with the received packet, be it storing or transmitting its data. By default, all development kits except for development kit 1 initialise and default to a receive state. This is only changed upon reception of a packet from a specific development kit. This ensures sequential noiseless transmission. For example, if development kit 3 receives from development kit 1, it will store $RSSI_{13}$ and remain in a receive state. If development kits 3 receives from

development kit 2, it will store $RSSI_{23}$ and transmit $RSSI_{13}$ and $RSSI_{23}$.

All development kits transmit with a common radio packet payload structure, with each development kit only ever writing to appropriate payload elements. This is used for two main purposes. Firstly, for allocating incoming RSSI data sequentially, such that each development kit can write to the appropriate payload elements. Secondly, for transmitting and re-transmitting data to other development kits. As implemented, only the n^{th} development kit in an n node network stores all incoming payload elements to re-transmit values back to a central unit (in this case, the first development kit). However, in practical uses, the task of central data passing could be shared by any number of devices in many configurations. Another limitation is that reliability against failures of one development kit in the loop has not been added. These changes would add robustness in larger networks and for error checking and would be straightforward to add to the implementation described here.

Table 5.2 shows the radio packet payload structure. This shows the payload element array index, the payload contained in that element, and the development kit that writes to and transmits this data. For brevity, the structure for a 4-node network is shown. The number of payload elements is the same as the number of communication links, $\frac{n(n-1)}{2}$, which in this case is 6. The radio packet is initially empty, but the appropriate element is filled as each femto-spacecraft transmits in sequence. Note that this structure logically fills the payload with all elements pertaining to each development kit sequentially, which is a design choice.[1]

TABLE 5.2: RADIO PACKET PAYLOAD STRUCTURE

Payload element	Payload	Written by
1	$RSSI_{12}$	2
2	$RSSI_{13}$	3
3	$RSSI_{14}$	4
4	$RSSI_{23}$	3
5	$RSSI_{24}$	4
6	$RSSI_{34}$	4

The full radio packet format is shown in Fig. 5.4. The preamble is a short set of data at

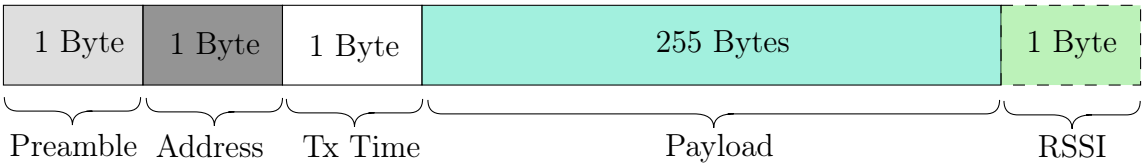


FIGURE 5.4: NETWORK RADIO PACKET FORMAT

the start of a packet used by the radio modules to signal the beginning of a packet. This is followed by the transmit address, specific to each development kit as shown in Table 5.1. Next the transmit time is appended, followed by the payload, which in this case contains

[1] pseudocode is written to show elements for a 4 node network (using zero-based numbering in C)

indexed RSSI values from packets previously sent to other development kits. The new RSSI data from this transmitted packet becomes available at the receiving end, appended onto the received packet structure. All RSSI data is available at a resolution of 1 dBm due to its 8-bit representation [117]. The fourth development kit in a 4-node network would use address filtering and this payload packet structure to store incoming data from all development kits and transmit according to the following pseudocode:

Algorithm 5.2 Full network RSSI transmit packet process

Inputs: Rx Packet

Enter receive packet callback function:
if *packet received* **then**
    check Rx packet address and save as $Rx\_address$
    if $Rx\_address$ = 0x01 **then**
        store $RSSI_{14}$ in transmit packet payload element 2
        remain in receive state
    end if
    if $Rx\_address$ = 0x02 **then**
        store received packet payload element 0, $RSSI_{12}$, as transmit payload element 0
        store $RSSI_{24}$ in transmit packet payload element 4
        remain in receive state
    end if
    if $Rx\_address$ = 0x03 **then**
        store received packet payload element 1, $RSSI_{13}$, as transmit payload element 1
        store received packet payload element 3, $RSSI_{23}$, as transmit payload element 3
        store $RSSI_{34}$ in transmit payload element 5
        switch to transmit state
        save Rx packet reception time
        transmit packet
    end if
end if
post semaphore signalling callback task complete

In this case, the development kit utilises the additional received packet payload elements and writes them to its transmitting packet in addition to the new RSSI values. As implemented, this is sent to development kit 1, connected to a laptop which can store this data.

All further modifications to the TI 'rfEasyLinkEcho' project are now outlined in order to realise the network protocol as described above. In the original project showing bi-directional communication between 2 development kits, one development kit runs the 'rfEasyLinkEchoTx' program while the other runs the 'rfEasyLinkEchoRx' program. The former initialises in a transmit state, transmitting a radio packet every second, switches to a receive state, awaits reception of the echoed radio packet, and if successful, repeats this process continuously. The latter initialises in a receive state, switches to a transmit state upon reception of a radio packet, and after a 500 ms delay, echoes back the received radio packet. For the fully interconnected network, development kit 1 runs a modified 'rfEasyLinkEchoTx' program while all other $(n-1)$ development kits run different modified 'rfEasyLinkEchoRx' programs.

The radio packet payload is represented by an array of signed integers structured to store RSSI data. This was modified from the default unsigned integer format in the Easylink documentation. Both the transmit and receive packet payload formats were changed to signed integers from the default unsigned integers, to preserve the RSSI values (and also any sensor and IMU data) shared over the network. RSSI values are stored in the development kit's internal memory as hexadecimal values, that when converted to decimal, form a signed 2's complement format corresponding to RSSI in decibel milli-Watts (dBm). The packet checking task on the 'rfEasyLinkEchoTx' program is also removed. In the original project, any received packet must be the same (echoed back). This is no longer the case as the payload is filled by all the development kits in the network. The default packet payload length was increased from the default 30 Bytes to the maximum natively supported by the Easylink API, which is 255 Bytes. This was done to accommodate all RSSI data for testing, which is made full use of for a network of up to 23 development kits as discussed in Chapter 6.

The delay period (T_{delay}) and transmit wait times are modified to accommodate more than 2 development kits. This is configurable for the desired refresh rate for a given network size. The basis for this modification is that development kit 1 must repeat transmission at a rate that provides all other development kits with an opportunity to transmit before development kit 1 transmits again. While a lower limit on refresh rate has not been tested, testing has successfully shown that a 100 ms delay period between each development kit transmitting works with at least 23 development kits. As described above, development kit 1 would wait nT_{delay} (2.3 s for a 23-node network) before restarting the data gathering sequence. As such, the wait time increases as the number of development kits on the networks increases. This delay period of 100 ms was lowered from the default 500 ms to maximise data return with time for outdoor testing. As all other development kits only transmit after receiving from a specific address, no timer modifications need to be made to these programs. Only on the first development kit is it crucial that this delay is programmed to allow enough time to elapse before the transmit cycle begins again. Within the main program loop,[2] a semaphore pending function is added to wait for the suitable amount of time to elapse before development kit 1 is able to transmit again.

Incoming network data sent to development kit 1 is processed and stored in real-time over a serial port link. This is added to the program using inbuilt TI-RTOS display functions that can show incoming data live on a terminal. A MATLAB program was developed to read in this serial data live from the embedded program running on development kit 1 and process it in real time for relative positioning testing. In the experimental testing described in Chapter 6, development kit 1 is connected via USB to a laptop, storing and processing incoming data transmitted from development kit n. Development kits $(2 - n)$ operate wirelessly and are powered by USB battery packs. In this setup, the laptop can be thought of as the central carrier spacecraft for the other femto-spacecraft, which in space could be a CubeSat that deploys the femto-spacecraft swarm. In testing, development kits were programmed to display a green LED

[2] all embedded systems run on infinite loops until reset or turned off, a 'while(1)' in C

when in receive mode and a red LED when in transmit mode for debugging purposes. For initial testing and development purposes, displaying the incoming data live on a terminal provided a useful way to check that communications throughout the network were working reliably before moving onto experimental testing outdoors. Additional display function calls were used on other development kits while testing to check that packets were being received reliably and in the correct order. Supplementary figures from testing are provided in Appendix A.3.

The embedded program operations are summarised in the following flowcharts. Figure 5.5 shows the operations for development kit 1, while Fig. 5.6 shows the operations for all other development kits on the network. Note that Fig. 5.6 has highlighted optional modifications as described in Section 5.2.3. Inputs are shown with dashed lines to indicate that these are conditional.

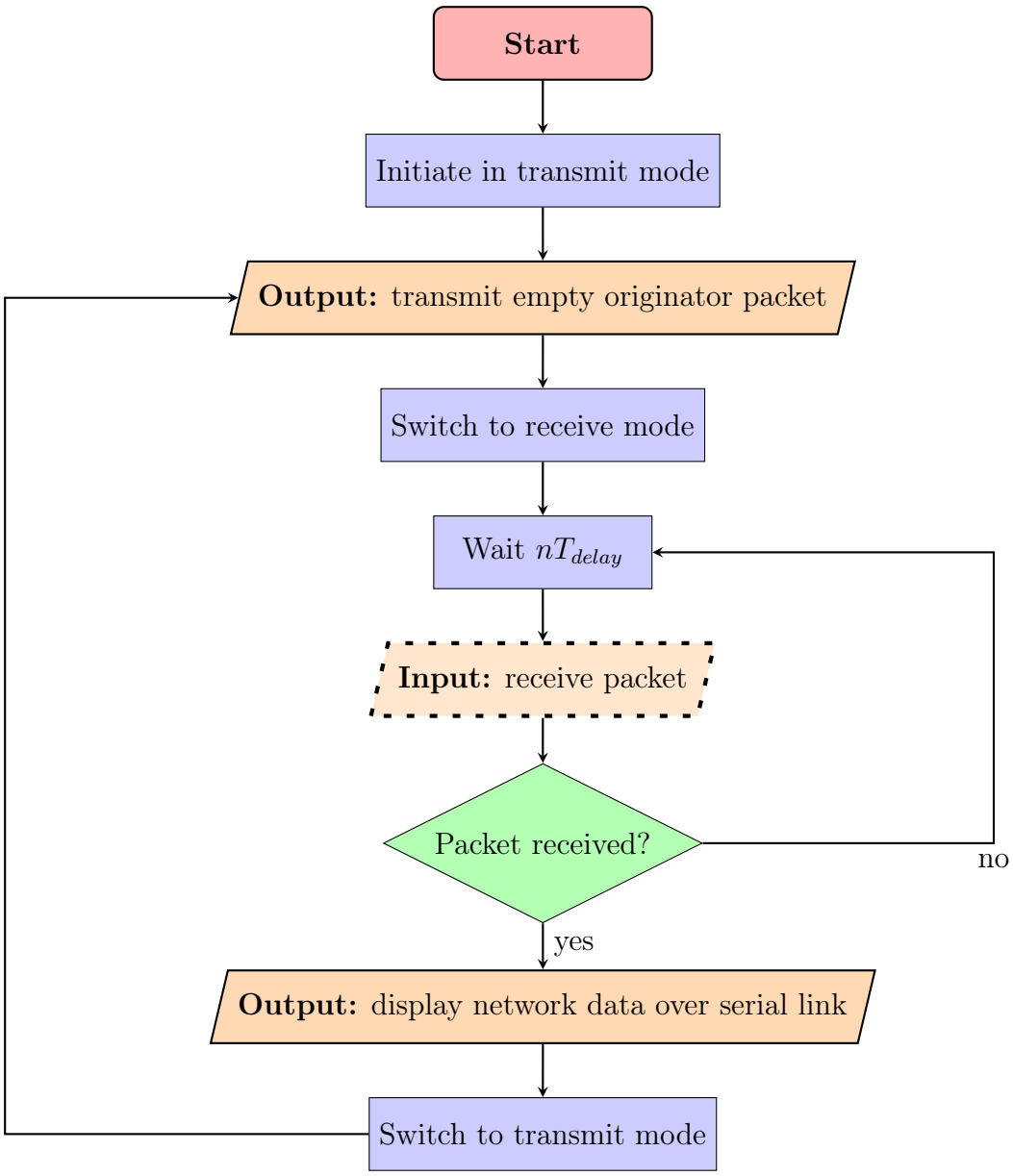


FIGURE 5.5: DEVELOPMENT KIT 1 NETWORK FLOWCHART

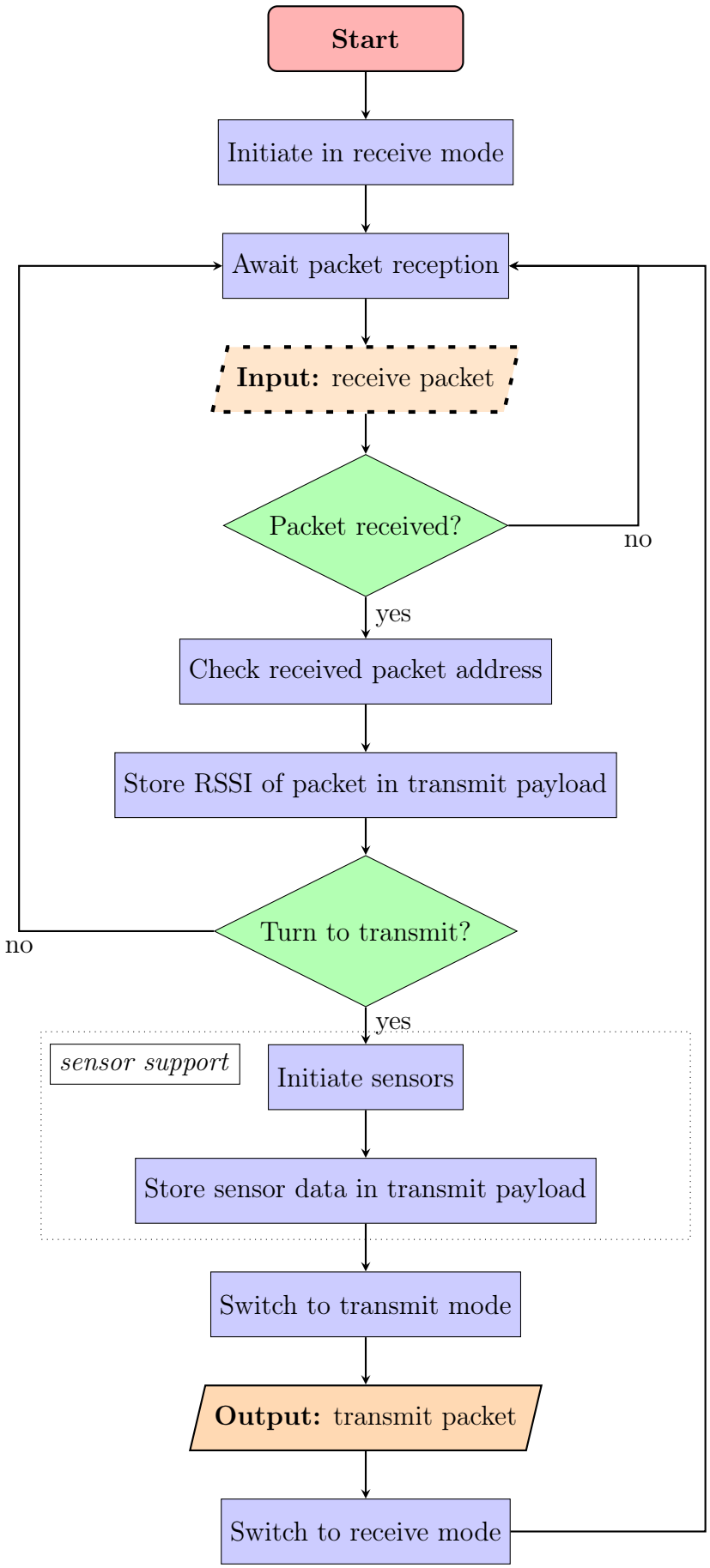


Figure 5.6: Development kits $(2-n)$ network flowchart

The approach detailed here would be compatible with a substantially larger number of development kits. However, as computational and storage limits would be reached as the number of devices within the network increases, it would be necessary to develop decentralised and distributed network structures and protocols for larger networks. This approach could be straightforwardly decentralised by having the central femto-spacecraft of one cluster share data with the central femto-spacecraft of another cluster, as shown in Fig. 5.7.

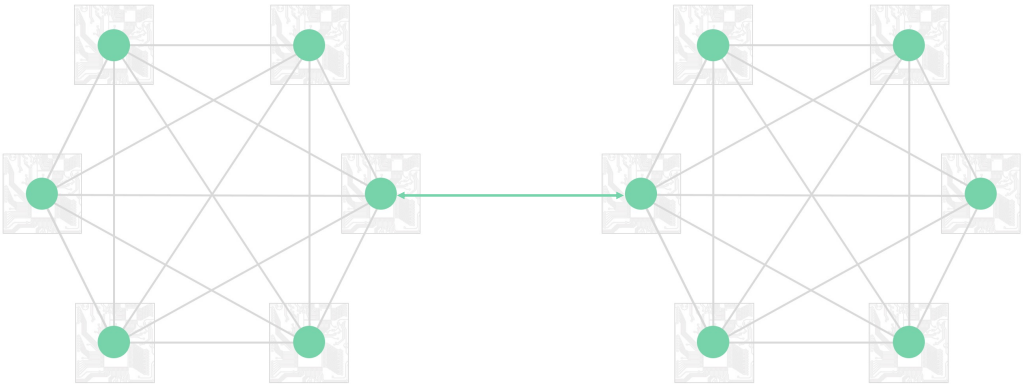


FIGURE 5.7: DECENTRALISED NETWORK COMMUNICATIONS

Decentralisation could be enabled by the nomination of cluster heads to accumulate individual cluster data, share between other cluster heads, and (but not by necessity) send this to the central deployer spacecraft. This modification would require synchronisation between the cluster heads, and could be added to the the fully interconnected network process by implementing further semaphore based functionality and delay period timing on programs running similar to development kit 1.

5.2.3 MODIFICATIONS TO SUPPORT SENSOR DATA

In addition to the network facilitating the sharing of RSSI data, in orbit, a femto-spacecraft would need to be able to communicate data from its sensor suite to other members of the swarm. In this section, the integration of IMU and sensor data collected on TI BoosterPacks on each development kit across the network is detailed as an extension to the above approaches. For IMU data, this would enable attitude estimation or the consideration of attitude-dependent effects for range estimation in extended 3D testing, or for sensor data, experimental demonstration of distributed sensing methods. This is implemented by porting sensor functionality into the existing embedded programs developed for the network protocol. This was achieved by adding the SAIL plug-in, which provides the sensor functions and driver support required [118]. Shared dependencies, directories, required linker files, headers and support files are added to the network protocol programs to enable compatibility with the booster packs.

The main modification is that communications and sensing are now treated as two separate tasks. Both tasks are based on semaphores, with the communications task given a higher priority than the sensing task. In this way, operational networking is prioritised over being able to sense data but not communicate it. This program structure avoids the use of threading

(i.e. having the networking operate as a main thread and the sensor tasks as a separate thread simultaneously), and means that the sensor tasks run only when required, saving power operationally.

Within the communications task, once the device has gone into receive mode, a callback can unblock the sensor task. The sensor task initialises the IMU and sensors, reads the data and stores the values in memory. This data is then added onto the transmit packet payload structure to send over the network when it is the development kit's turn to transmit, according to the network protocol. As implemented in testing, the callback on each development kit is made upon receiving from the address that would trigger a packet to be transmitted. Therefore, the sensors are sampled immediately before transmitting a packet, providing the most up to date measurements. In the pseudocode presented in Algorithm 4.2, the callback would be made between lines 18 and 19. This is shown in Fig. 5.6 as an additional sensor support task.

5.3 Chapter Summary

In this chapter, the implementation of a fully interconnected protocol developed for femto-spacecraft networking has been presented. This enables the experimental testing of relative positioning algorithms in Chapter 6. Embedded software was developed in C for development kits using an MCU and MEMS sensor technology representative of what is currently possible at the femto-spacecraft scale. This implementation could be straightforwardly ported onto femto-spacecraft PCB test-beds.

The main network protocol was outlined first as an operational process, and then as a series of software modifications to TI bi-directional communications demonstrations. Additionally, modifications to integrate sensor and IMU data passing over this protocol were outlined, made possible using separate task and callback mechanisms with semaphores in C.

The protocol is utilised extensively in Chapter 6 for path loss modelling and range-based relative navigation demonstrations. Throughout testing, this address filtering technique has been found to operate reliably with up to 23 development kits. However, it is anticipated that this process would work with a far higher number of femto-spacecraft in practice. This approach could also be straightforwardly decentralised using the methods outlined in this chapter.

Chapter 6

Experimental Demonstration

Following the development and testing of algorithms and filters in simulation and the network protocol tested on hardware throughout Chapters 3-5, this chapter presents the results of an experimental test campaign using development kits as femto-spacecraft proxies in an outdoor testing environment. The first step in this process is the development of a path loss model (PLM) to use RSSI data as a range metric. This involves both antenna characterisation and an empirical derivation of a PLM for a given testing environment by taking RSSI measurements between two development kits placed at known distances apart from one another. Finally, the results of a small-scale relative positioning and navigation demonstration are presented, using up to 23 development kits placed in known locations around the testing environment. To do this, networked RSSI data is filtered through the PLM function, which is then provided to the algorithms developed in Chapter 3 as range estimates between the development kits. The algorithm outputs are then compared against the known locations to assess experimental performance in the demonstration.

6.1 Path Loss Modelling

In this section, the development of a PLM used to convert raw RSSI data into a range estimate is presented. This comprises the theoretical basis of the model, antenna characterisation, and finally the experimental derivation of the PLM for the testing environment used.

6.1.1 The testing environment

Terrestrially, environmental factors such as the atmosphere, weather, terrain, topography, electromagnetic interference and obstacles such as trees and buildings have a significant impact on the nature of radio propagation. Consequently, the effect of these factors in addition to the theoretically expected attenuation of radio waves can be accounted for in path loss modelling. However, it is desirable to test within a clear and wide open space with flat terrain to minimise random effects. This also means that the effect of RSSI on range estimates can be modelled between all development kits predictably and reliably within a given environment. As the de-

velopment kit PCBs are not insulated, and to reduce the effect of the impact of weather, testing was only conducted on dry days.

As a result of these requirements, a 120 m × 60 m sports pitch at the University of Glasgow's Garscube Sports Complex was used for outdoor testing. This environment is ideal because the area is wide, open, and flat, and the length-scale is representative of a swarm of femto-spacecraft in orbit when initially deployed in LEO with a small, representative deployment impulse. Development kits were placed approximately one metre off the ground on top of tripods made of bamboo to minimise radio interference. Each development kit is powered using a small battery pack attached to one of the legs of the tripod. The testing environment is shown in Fig. 6.1.

(a) Sports pitch (b) Development kit

Figure 6.1: The outdoor testing environment

All results in this chapter for the path loss model development and relative positioning and navigation demonstrations were obtained in this testing environment via a series of tests conducted between July 2022 and February 2023.

6.1.2 Model development from principles of radio propagation

In theory, an isotropic transmitter will propagate radio waves equally in all directions of 3D space. As this spherical wavefront extends outwards from its source, a perfect radiator antenna, the wavefront surface area is ever-increasing, causing attenuation of the signal's strength in free space according to the inverse-square law, as shown in Fig. 6.2. As the energy spreads out over a larger area, the signal strength weakens proportional to the square of the distance travelled from the source [119]. Theoretically, the fraction of the power P_{tx} transmitted at the source that can be received a distance r from it by a perfect receiver comes simply from the surface area of the sphere:

$$P_{rx} = \frac{P_{tx}}{4\pi r^2} \tag{6.1}$$

Therefore, if the received signal strength has a value of P_{rx} at a distance r from the source, then the strength will weaken as $\frac{P_{rx}}{4}$ at $2r$, $\frac{P_{rx}}{9}$ at $3r$, and so on.

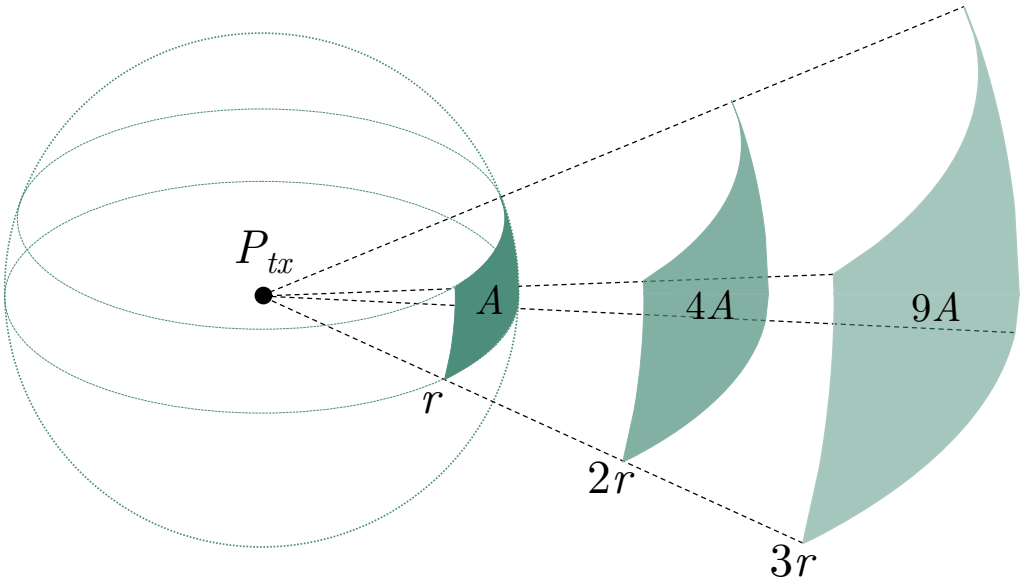


FIGURE 6.2: ISOTROPIC RADIATION ACCORDING TO THE INVERSE-SQUARE LAW

In reality, no physical antenna is fully isotropic, and even undirected (or 'omni-directional') antennas exhibit some degree of directed losses due to nulls in the antenna's radiation pattern. Antenna directivity, $D(\theta, \phi)$, defined as a function of angular displacement in a spherical coordinate system,[1] is a measure of how directed an antenna's signal is with respect to a theoretical isotropic antenna. A directivity of 1 would imply isotropic radiation.

An antenna's radiation efficiency, η_r, is the ratio of radiated power to the input power supplied to its terminals (or, when receiving a radio signal, the ratio of power received to that delivered as electrical output). Then, the antenna gain $G(\theta, \phi)$ is the product of directivity and radiation efficiency:

$$G(\theta, \phi) = \eta_r D(\theta, \phi) \tag{6.2}$$

The Friis transmission formula [120], expressed in terms of the gains of the receiving and transmitting antennas, outlines the free space path loss (FSPL) relationship of a radio signal with distance:

$$\frac{P_{rx}}{P_{tx}} = G_{rx} G_{tx} \left(\frac{\lambda}{4\pi r} \right)^2 \tag{6.3}$$

where G_{rx} and G_{tx} are the receiver and transmitter antenna gains, λ is the signal wavelength and r is the distance between the receiver and transmitter. As signal strength is inversely proportional to the square of the distance, it is convenient to use the logarithmic decibel (dB) scale. Antenna gain can therefore be expressed in decibels relative to an isotropic radiator (dBi). Signal power for small antennas and radios using industrial, scientific, and medical (ISM) bands can therefore be conveniently expressed in decibel-milliwatts (dBm). This unit

[1] where θ is the polar angle and ϕ is the azimuth angle

expresses a power ratio in dB with respect to 1 mW:

$$P^{[dBm]} = 10 \log_{10} P^{[mW]} \tag{6.4}$$

The Friis transmission formula can therefore be expressed logarithmically as:

$$P_{rx}^{[dBm]}(r) = P_{tx}^{[dBm]} + G_{rx}^{[dBi]} + G_{tx}^{[dBi]} + 20 \log_{10} \left(\frac{\lambda}{4\pi r} \right) \tag{6.5}$$

As the transmitting power and the antenna gains are constants for a given antenna orientation, the received power as a function of distance from the transmitter in free space typically depends only upon the final term in Eq. (6.5). However, free space is the absence of any obstructions or environmental factors causing reflection, refraction and diffraction of radio waves and resulting in multi-path effects. Even in clear testing environments, where FSPL will be the dominant effect, it is necessary to account for additional factors that alter radio propagation behaviour.

In addition to FSPL, signal attenuation in outdoor testing will be impacted by two predominant factors that are necessary to model. Firstly, by environmental factors causing multi-path errors through reflection and diffraction of radio waves. These effects are mitigated by using a clear, flat, and open space to test within, as discussed in Section 6.1.1. Secondly, anisotropy in radio wave propagation (due to the antenna's directivity and orientation) will also affect the received signal strength. As the experimental testing in this work is static and limited to 2D, it is found from the modelling and antenna characterisation in Section 6.1.3 that this effect can be minimised with careful arrangement of the antennas used. This is because the effect of orientation on the theoretical radiation pattern in 2D is in principle negligible. The path loss model developed ultimately takes the form of a modified version of the Friis transmission formula that incorporates these additional effects.

Defining the path loss PL as:

$$PL^{[dBm]} = P_{tx}^{[dBm]} - P_{rx}^{[dBm]} \tag{6.6}$$

A logarithmic model can be developed that takes the form:

$$PL(r) = PL(r_0) + 10\gamma log_{10} \frac{r}{r_0} + \chi \tag{6.7}$$

where $PL(r_0)$ is the path loss at a reference distance r_0, found from Eq. (6.3), γ is the path loss exponent, r is the path length between the receiver and the transmitter, and χ is a zero mean normal random variable that represents the effect of attenuation due to environmental factors. The path loss exponent is of course 2 for free space but must be experimentally determined for a given environment.

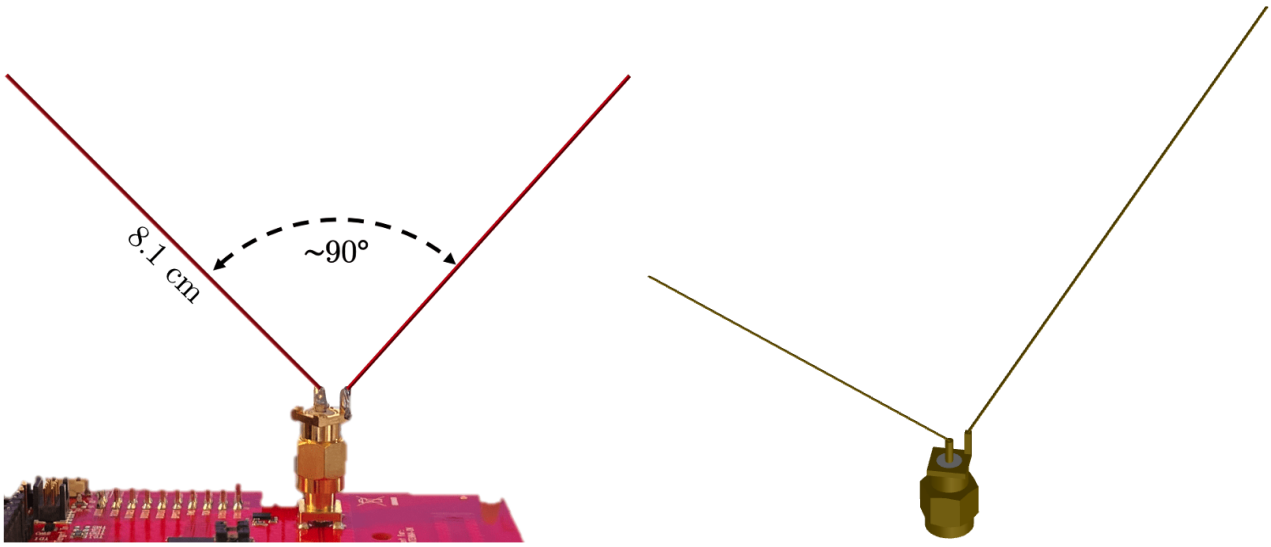
The received signal strength indication (RSSI) is a measurement of the received power P_{rx},

that is, the RF power input to the radio transceiver [121]. RSSI is typically used as an indicator of signal quality, how robust a network is to losses, and as an indicator for maximum operating distances. As discussed in Chapter 5, provided all femto-spacecraft transmit with the same power, network RSSI data will indicate a correlated relative received power that can be associated with range without calibration. Therefore, an RSSI-based version of a PLM function can now be developed experimentally to characterise signal strength decay with range in a given testing environment, as described in Section 6.1.4. However, it is firstly necessary to characterise the real antennas used in testing.

6.1.3 Antenna modelling and characterisation

The antennas used in testing were characterised using a combination of simulation-based modelling and experimental techniques using an RF anechoic chamber facility at the University of Glasgow. A 3D model of the antenna was made in Solidworks and imported to Ansys HFSS to model its radiation pattern. This was then compared with experimental results to verify the antennas that were used operate as intended.

Half-wavelength 868 MHz V-dipole antennas were fabricated with 0.5 mm diameter enamelled copper wire soldered onto an SMA connector and attached to the development kits as shown in Fig. 6.3.



(a) Placement on development kit (b) 3D Model

Figure 6.3: Half-wavelength 868 MHz v-dipole antenna

V-dipole antennas were chosen due to their undirected radiation patterns, which for testing minimises the impact of antenna orientation on the relationship between range and RSSI. This is a significant benefit over using the integrated PCB trace antenna on the development kits. A small modification to the development kit PCB was required to enable this antenna change, namely the 0 Ω resistor that links the trace on the PCB to the integrated antenna was moved to close this link and open the link to the external antenna port [108]. The angle between the

antenna wires was kept approximately at 90° during testing. There is literature to suggest this angle should optimally be kept closer to 120° [122], but 90° was chosen to ensure consistency when testing with many development kits. As the antenna arms are flexible, it is much easier to quickly determine an approximate right angle when attaching to the development kits and in use outdoors for consistency.

V-dipole antennas have toroidal radiation patterns. This corresponds in theory to radiation equally in all directions in the plane perpendicular to the antenna wire, reducing to null along the antenna wire. For the V-dipole antenna with the properties used here, this results in the radiation pattern shown in Fig. 6.4. Note that the orientation of this pattern reflects the antenna being oriented as shown in Fig. 6.3. In testing, the antennas were oriented perpendicular to this, as shown in Fig. 6.1.

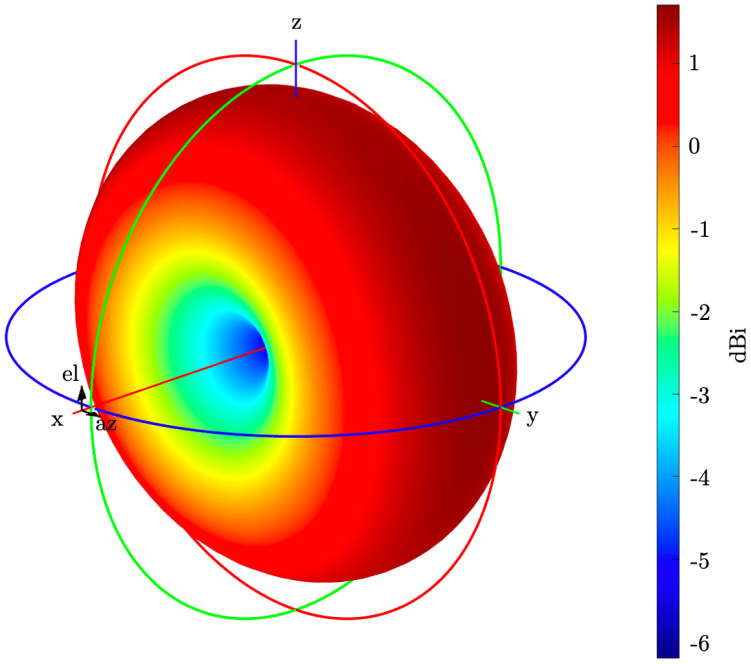


Figure 6.4: Radiation pattern of a perfect 868 MHz v-dipole antenna

If this antenna was aligned vertically with respect to a plane shared by the receiver (in this context the ground), this maximises the signal received. In contrast, if the two antennas were aligned horizontally along the same plane facing the receiver with the direction of the antenna wire, this would minimise the signal received. This is because the principle of reciprocity applies to antennas, meaning the radiation pattern for transmitting a signal reflects its ability to receive a signal in those same directions. The antenna is a half-wave dipole, requiring that the length of each antenna arm is approximately a quarter-wavelength long ($\frac{\lambda}{4} = 8.64$ cm). In practice, the exact length required is marginally smaller than a quarter-wavelength due to not operating in a vacuum and the real effects of the waveform being carried through a wire (i.e. the velocity factor of the copper wire used). Real antennas therefore need to be correctly characterised to test their properties at the desired frequency of operation. This involves verifying that the antenna resonates at the intended frequency and characterising its gain and radiation pattern [123].

Firstly, a vector network analyser (VNA) was used to perform tuning measurements on fabricated antennas. An antenna's S_{11} parameter describes its reflection coefficient, also known as return loss. This is a measure of the reflected power that the radio is delivering to the antenna. A maximal $S_{11} = 0$ dB implies all power delivered to the antenna at that frequency is reflected and no power is radiated. As it is desirable to maximise the power radiated by the antenna, it is desirable to minimise this value at the frequency of operation. The S_{11} parameter data obtained from the VNA over a range of frequencies is shown in Fig. 6.5.

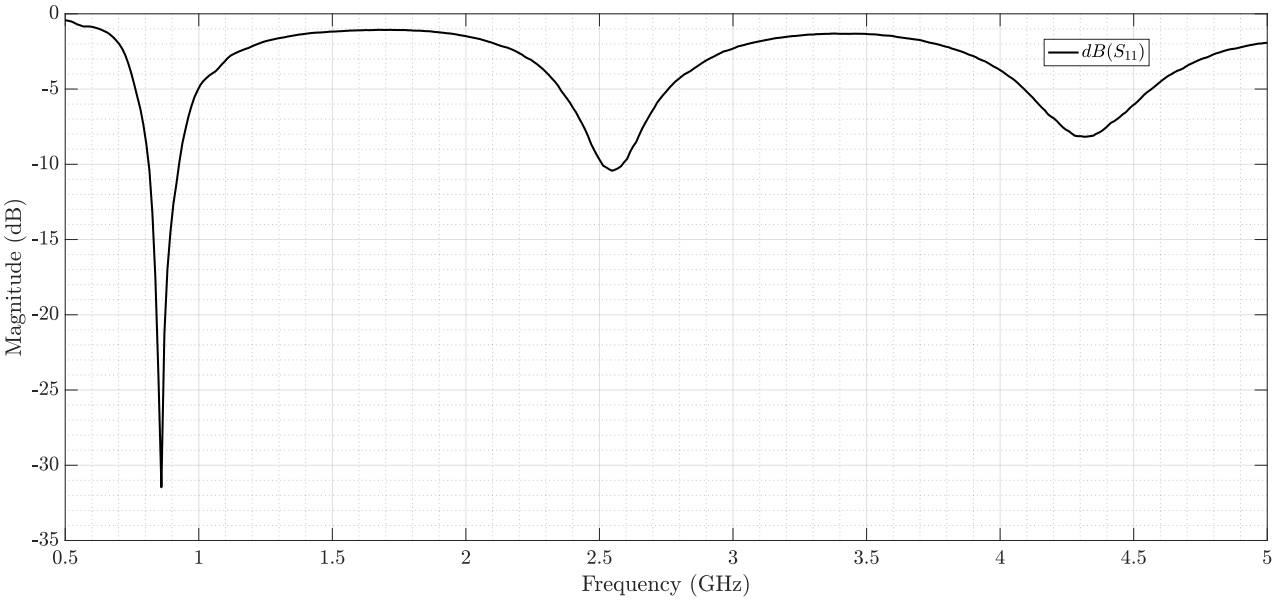
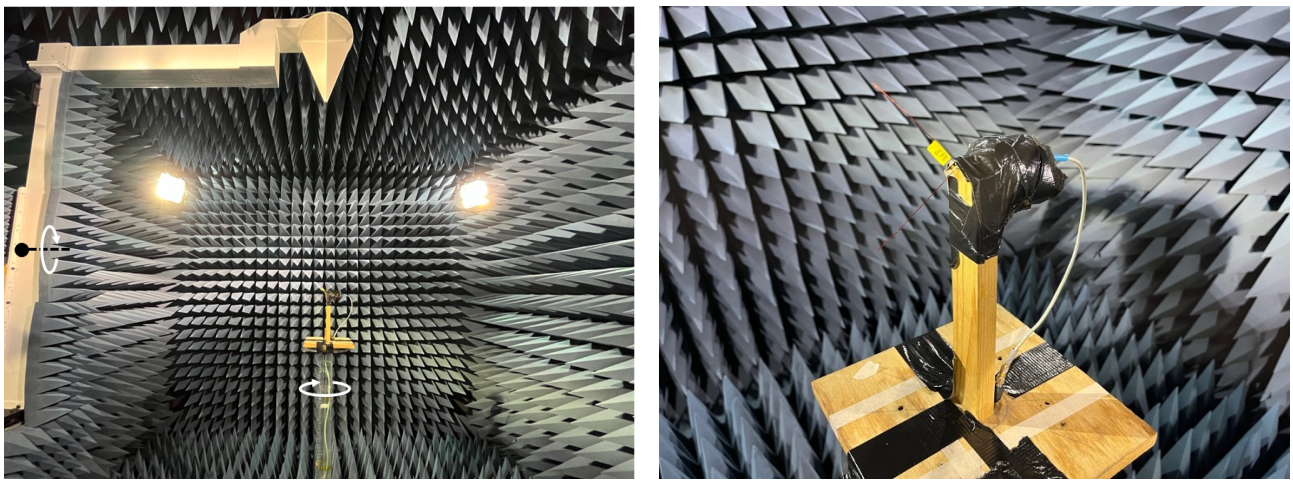


FIGURE 6.5: S_{11} PARAMETER PLOT FOR THE 868 MHZ V-DIPOLE ANTENNA

This shows the highest reflection coefficient at 868 MHz for the half-wavelength V-dipole antenna tested, where it was found that an arm length of approximately 8.1 cm provides the desired antenna behaviour when accounting for the effect of the SMA connector and the spacing between the antenna arms. This means that the antenna will operate best at that frequency. This also illustrates that the antenna has a narrow bandwidth. The reflection coefficient dips observed at approximately 2.6 and 4.3 GHz are due to these frequencies being multiples of the operating frequency (causing some resonant behaviour).

Next, to characterise the antenna's gain and radiation pattern, the antenna was placed in an RF anechoic chamber as shown in Fig. 6.6. This facility provides a clean environment to test antennas by covering the ceiling, walls, and floor of the room in foam pyramids that absorb radio waves. The antenna under test is placed in the centre of the room, on a stage that can rotate from side to side about a vertical axis. The large moveable arm carries a calibrated directional antenna driven by an RF carrier at the frequency of interest, pointed towards the antenna under test. The arm pivots around the stage, enabling a full sweep of the antenna to characterise its far-field radiation pattern in 3D. The far-field radiation pattern of an antenna expresses its radiated power transmitted as a function of angle. This pattern is normalized by dividing the absolute values obtained in testing by the maximum transmitted power used.



(a) Test equipment (b) Antenna under test

Figure 6.6: RF anechoic chamber

This makes the pattern valid in general at the operating power used (which in this case is 14 dBm). The radiation pattern of an antenna provides the normalized magnitude (from polar coordinates) in dB of the signal strength at any angle. The experimentally obtained radiation pattern can be compared with the model, as shown in Fig. 6.7.

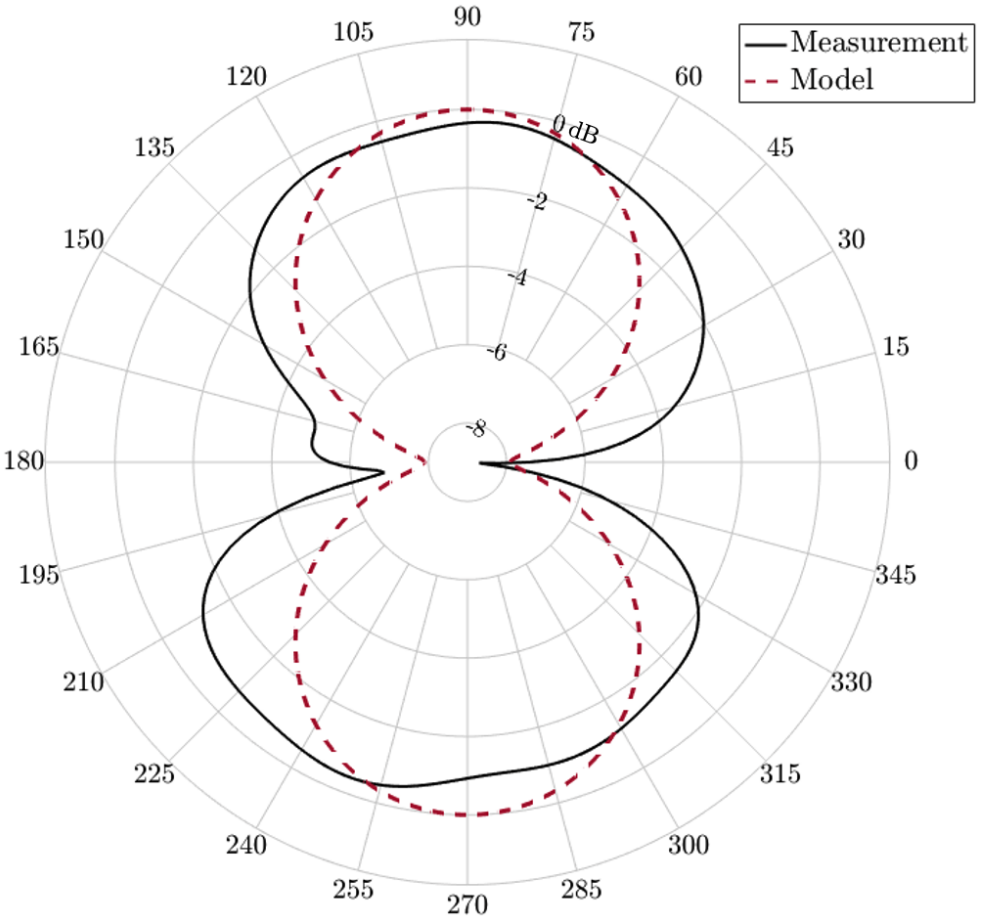


Figure 6.7: 868 MHz v-dipole radiation pattern

This shows that across the full sweep of azimuth angles, the antenna's gain is largely as expected from the model. Discrepancies at certain discrete measurement angles can be due to small unwanted reflections of radiation due to the metal in the testing equipment, and imperfections in the antenna geometry and materials. Overall however, this shows excellent undirected antenna gain characteristics required for testing in 2D, with close to 0 dB gain in the plane required. With the antenna tested and characterised, a number of antennas were made according to these specifications and used with the development kits.

6.1.4 Experimental derivation

The path loss model function from Eq. (6.7) can now be derived experimentally by logging RSSI values obtained from one-to-one communications between two development kits located at a series of known ranges apart from one another within the outdoor testing environment. This is shown in Fig. 6.8.

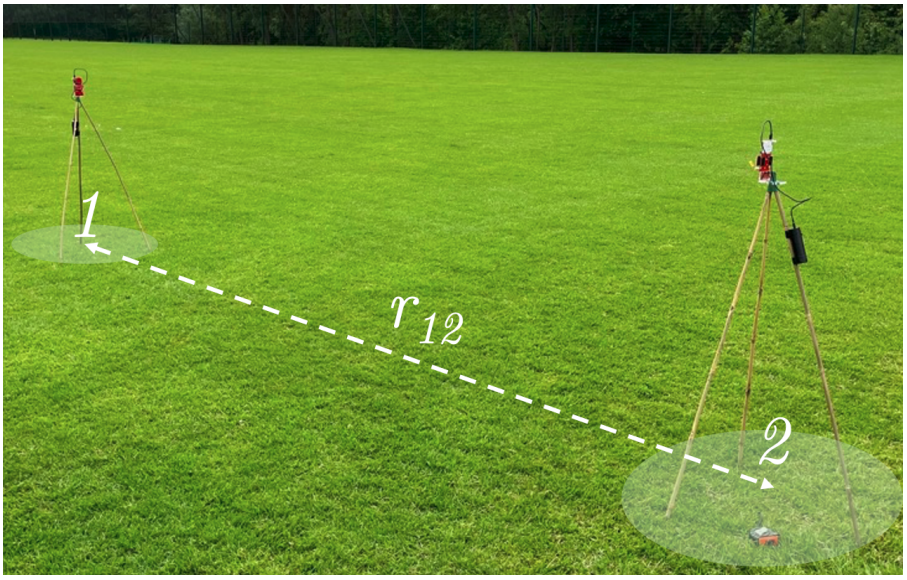


Figure 6.8: Path loss modelling procedure

Two development kits were placed between 1-150 m apart with the antennas placed facing towards each other. This is the most favourable alignment to maximise signal strength reception if any effect of PCB material interference is to be minimised. Test data was gathered periodically across many test days so as to not bias the model for the particular ambient weather conditions of the test day, and to verify the applicability of the model for given test days.

Further to this, the PLM function was enhanced by taking all RSSI samples gathered at all known (and measured) ranges between development kits in the outdoor testing environment during the relative positioning tests described in Section 6.2. The RSSI data used to develop the function is therefore a combination of PLM-specific data and experimental positioning data. The distinction is that the specific PLM data was obtained entirely for the purposes of developing an ideal PLM in the testing environment. The experimental positioning data however comes from the many configurations of multiple tests used for demonstrating relative

positioning throughout the testing programme. This provides test data at different orientations in 2D. This is important because it can be expected that the PCB (which houses thin layers of conductive copper) interferes with the antenna's radiation pattern to an extent, meaning that the propagation behind the antenna as implemented on the development kits is reduced, making this an additional effect to consider in 2D space. Developing a path loss model that incorporates these effects makes the model more generally applicable. All testing was performed with a transmit power of 14 dBm.

From the raw RSSI data, the PLM was developed in MATLAB using the curve fitting toolbox. Logarithmic space was used for range to develop a simple linear model using linear regression techniques, following the log-range model from Eq. (6.7). This results in a linear model with two coefficients that can be tuned as new modelling data is gathered or for different test environments. This takes the form:

$$RSSI(r) = c_1 \log_{10}(r) - c_2 \tag{6.8}$$

where the coefficients c_1 and c_2 were derived as (with 95% confidence bounds given in brackets):

$$c_1 = -22.91 \quad (-23.21, -22.61) \tag{6.9}$$

$$c_2 = -20.64 \quad (-21.17, -20.12) \tag{6.10}$$

For these values, RSSI is measured in dBm and the range r is in metres. Inverting this function, a range estimate as a function of RSSI is obtained:

$$r(RSSI) = 10^{\left(\frac{RSSI - c_2}{c_1}\right)} \tag{6.11}$$

This can then be used as an input for the relative positioning algorithms to generate range estimates between development kits in testing. The path loss model is shown in Fig. 6.9, showing all RSSI data used to generate the function, along with the 95% confidence bounds. Note that the experimental data is obtained at even distances apart, but the spacing of the samples appears closer together at larger distances due to the logarithmic axis used.

This demonstrates the clear trend between RSSI and range in this logarithmic model, but also the considerable inaccuracy of raw RSSI values as range estimates. This is seen particularly at larger ranges, where the increment of individual RSSI values can represent several metres. It is also observed that the distribution of RSSI values increases with range, showing considerably larger variation outwith the model confidence bounds for some sampled values at the largest ranges. Mitigating against these large inaccuracies in the range estimates available from this approach is a central part of the algorithmic development presented in Chapter 3 and is also addressed for sensor accuracy in Chapter 7. Ultimately, as the confidence bounds in the model are accounted for with the algorithmic implementation, and as the number of femto-spacecraft in the swarm increases, the expected probabilistic effect of fused data is to compensate consid-

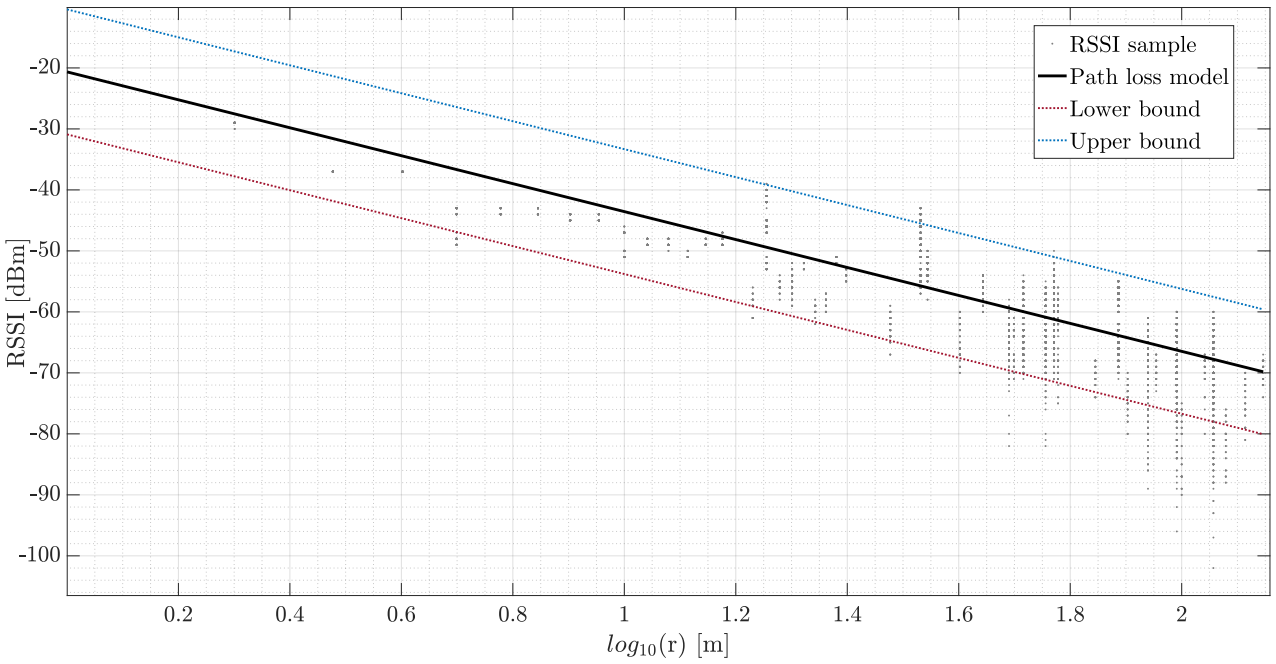


Figure 6.9: Experimentally derived path loss model

erably for this inaccuracy. However, it is important to acknowledge that this indicates that raw RSSI data is a very coarse proximity metric for estimating the range between two development kits in this environment.

As discussed, the PCB may cause interference with the antenna's radiation pattern. This means that the propagation behind the antenna as implemented on the development kits is reduced, making this an additional effect to consider in 2D space. This effect has however been neglected for experimental testing of relative positioning. The effect that orientation has on signal strength at different ranges was tested, but the reduction in signal strength for all orientations in 2D (including the least favourable scenario of both antennas facing directly away from one another) was found to well within the approximately 20 dBm confidence bounds of the PLM derived for all 2D orientations (at most 5 dBm). To develop a ranging metric in 3D space, further antenna characterisation and modelling, along with relative attitude data provided by an IMU, may improve the range estimates by scaling the PLM function according to the relative attitude between two femto-spacecraft. This would be possible by explicitly considering the directivity of the antenna and relating this to the attitude of the femto-spacecraft, potentially as an update step after an initial relative positioning calculation without considering attitude.

6.2 Relative Positioning Demonstration

In this section, the algorithmic implementation of the ranging metric derived from the PLM for experimental demonstration of range-based relative positioning and the testing environment used is detailed. Experimentally, the RSSI data is filtered through the PLM, then input to the relative positioning algorithms. As this is a small demonstration using up to 23 development kits as femto-spacecraft proxies, only the centralised SDP and NPDF algorithms from Sections

3.2 and 3.4 are tested experimentally. A series of relative positioning tests were performed by using some development kits as anchors, with their positions already known to the relative positioning algorithms. The algorithms are only provided with the RSSI data between anchors and unknown nodes to be localised, along with RSSI data shared between unknown nodes.

Development kits were placed deliberately in measured known locations. Then, the outputs of the relative positioning algorithms were compared with the true location to assess performance and accuracy. All development kits were placed on the bamboo tripods approximately 1 m above the ground, with the orientation of the antenna with respect to the ground kept constant.

All RSSI data was collected using the fully interconnected network protocol developed in Chapter 5. In all experiments, the central development kit was connected to a laptop for real-time data logging and relative positioning calculations. This was enabled by developing a MATLAB program which reads live serial data from the development kit connected. All development kits transmit with a power of 14 dBm at a frequency of 868 MHz. All operators, the laptop and any other testing materials were kept outside of the perimeter enclosed by the development kits during testing to minimise interference.

The results of the relative positioning tests are now presented. Development kits were placed on the sports pitch in the configuration as shown in Fig. 6.10.

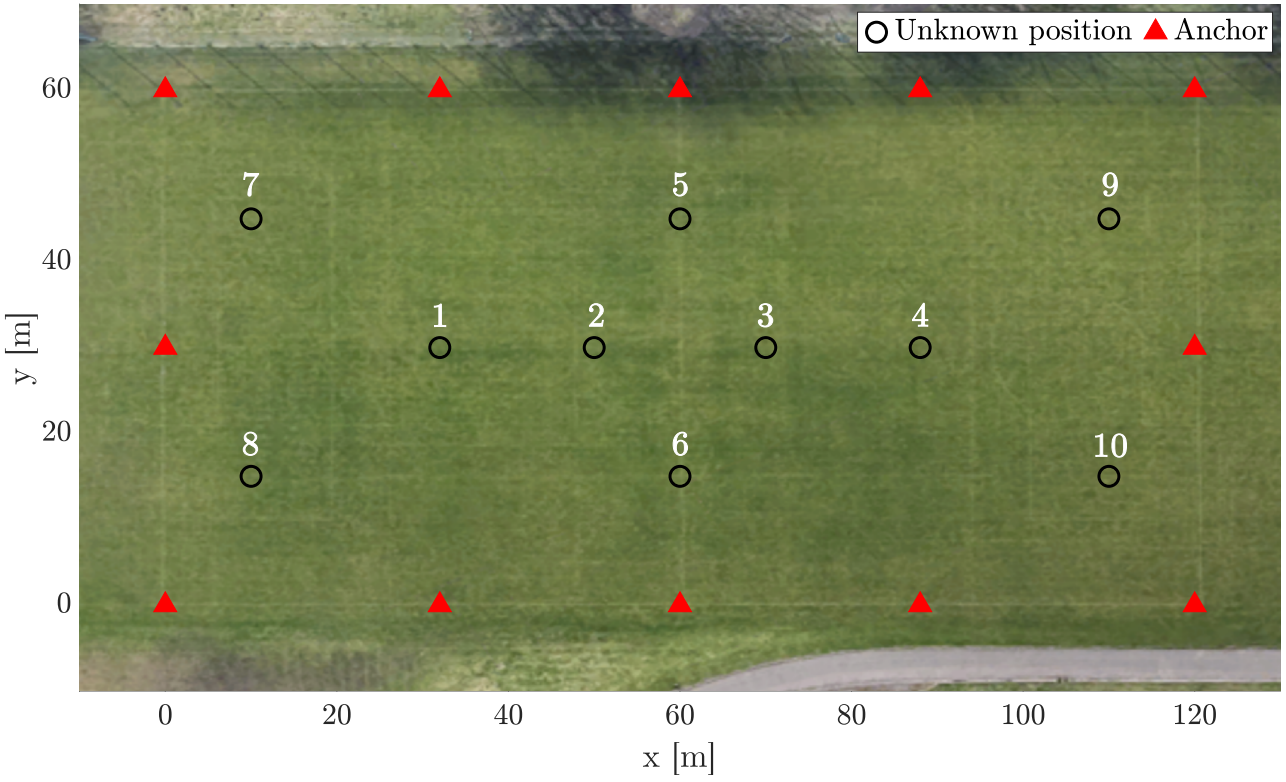


FIGURE 6.10: DEVELOPMENT KIT PLACEMENT[2]

With 12 anchor development kits (marked by the red triangles) located around the perimeter of the test environment, the objective was to estimate the positions of the 10 development kits (marked by the black rings) within the pitch. The particular placement is chosen by making use

of the rugby league pitch markings painted onto the grass, which makes the precise placement of development kits within the pitch easier to measure and verify, while distributing the kits fairly evenly throughout the test area. To analyse each algorithm's performance in detail, each development kit was localised using the RSSI constraints of the development kit of unknown position to the 12 anchors ($\hat{r}_{ik}$, as described in Chapter 3). This is equivalent to locating all 10 development kits of unknown position simultaneously while neglecting the presence of RSSI values shared between kits of unknown position.

The results are now presented throughout Figs. 6.11-6.20, where the NPDF (Section 3.4) and SDP (Section 3.2) algorithm estimates of position are marked by the burgundy and cyan crosses respectively. The heat map overlaid onto the imagery of the pitch is a visual representation of probability in 2D from the NPDF algorithm, which takes the highest value of this as its 2D position estimate. The NPDF outputs are generated using a mesh-grid that provides positions estimates to the nearest half metre. The result of each development kit's localisation is presented individually in order to discuss each algorithm's general performance in this test scenario. Figure 6.11 shows the results from both algorithms for the unknown position of development kit 1.

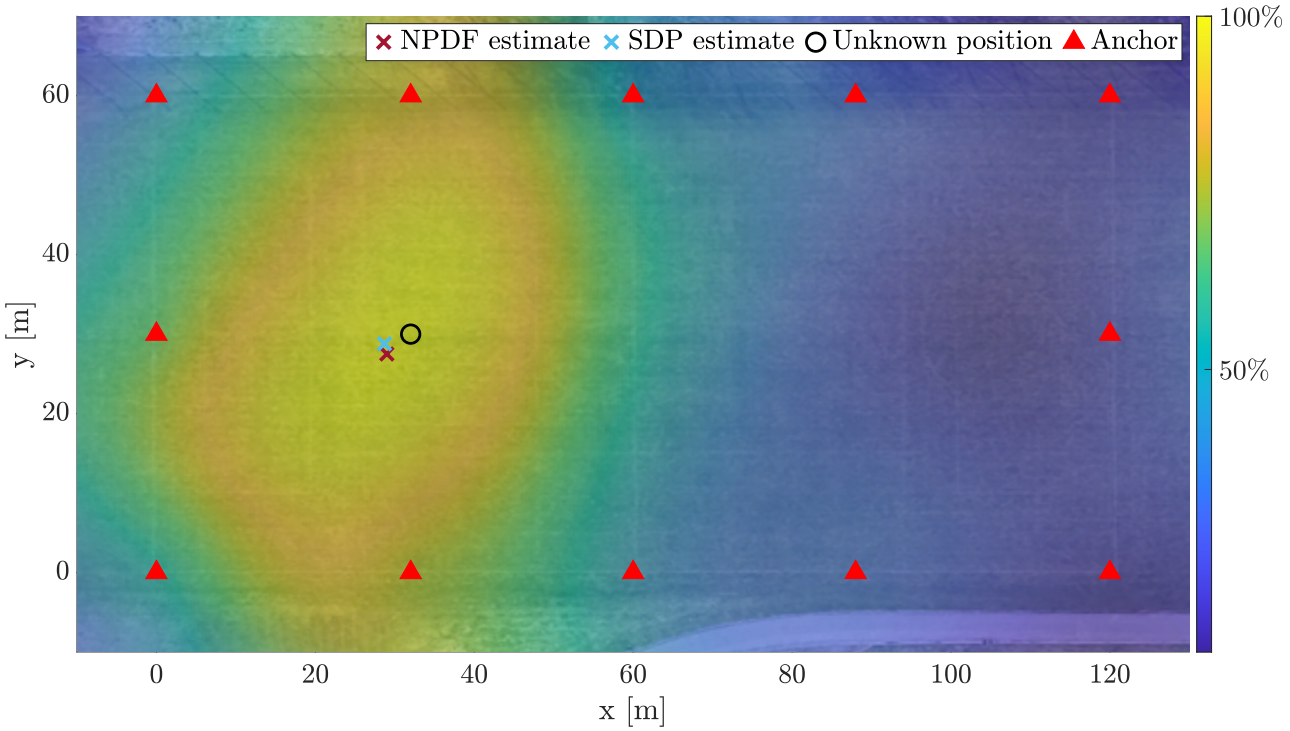


FIGURE 6.11: RELATIVE POSITIONING OF DEVELOPMENT KIT 1

The SDP position error is 3.5 m and the NPDF position error is 3.9 m. As with the simulation-based results in Chapter 3, the position error is taken as the range discrepancy from the true and estimated position. In this case, both algorithms perform comparably to localise the development kit of unknown position.

Figure 6.12 shows the results for the unknown position of development kit 2. In this case,

[2]Image and map data credits: Google Maps 2023

the SDP position error is 6.6 m and the NPDF position error is 7.4 m. The region of highest confidence for the NPDF estimate is again well defined and takes a similar form to the result for development kit 1, but the overall region shown by the sections shaded green where the probability of location is above 50% is much larger in this case. This highlights both algorithms' ability to deal with considerable inaccuracy in range approximations, and the importance of using as many range constraints as possible to prevent inaccuracy in estimates.

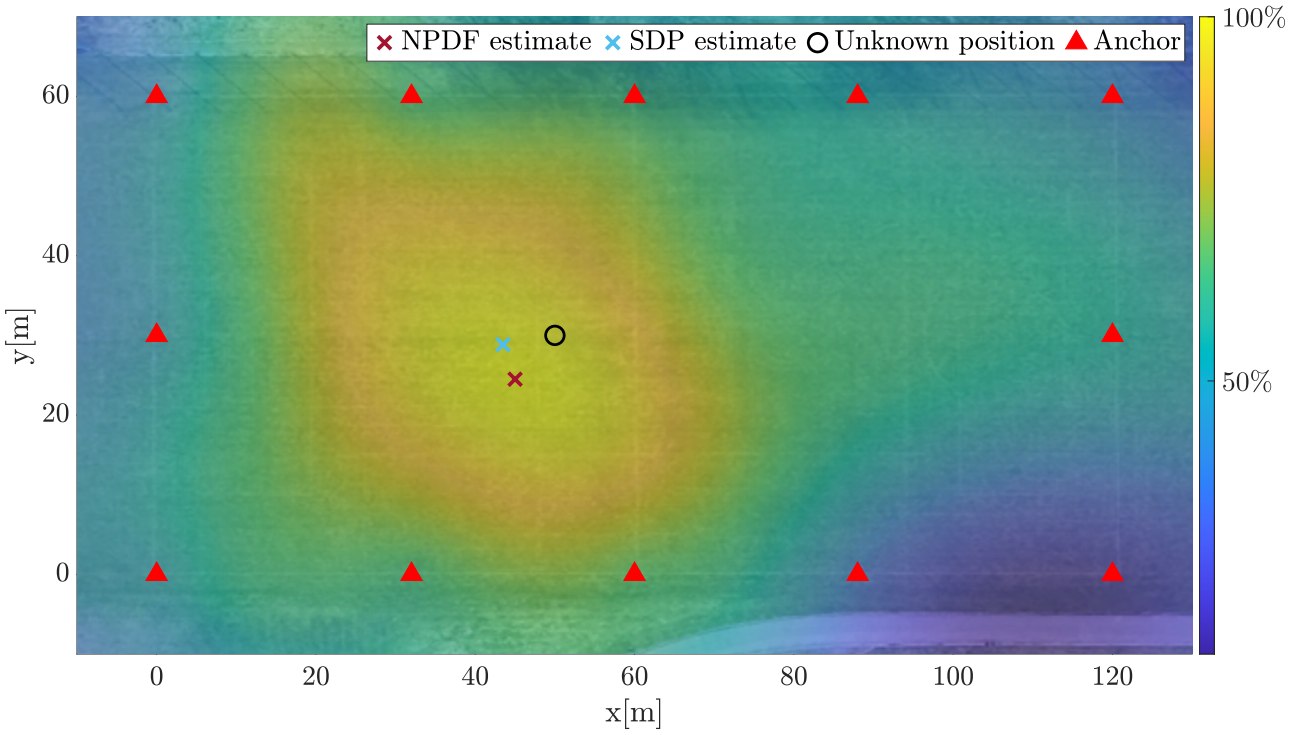


FIGURE 6.12: RELATIVE POSITIONING OF DEVELOPMENT KIT 2

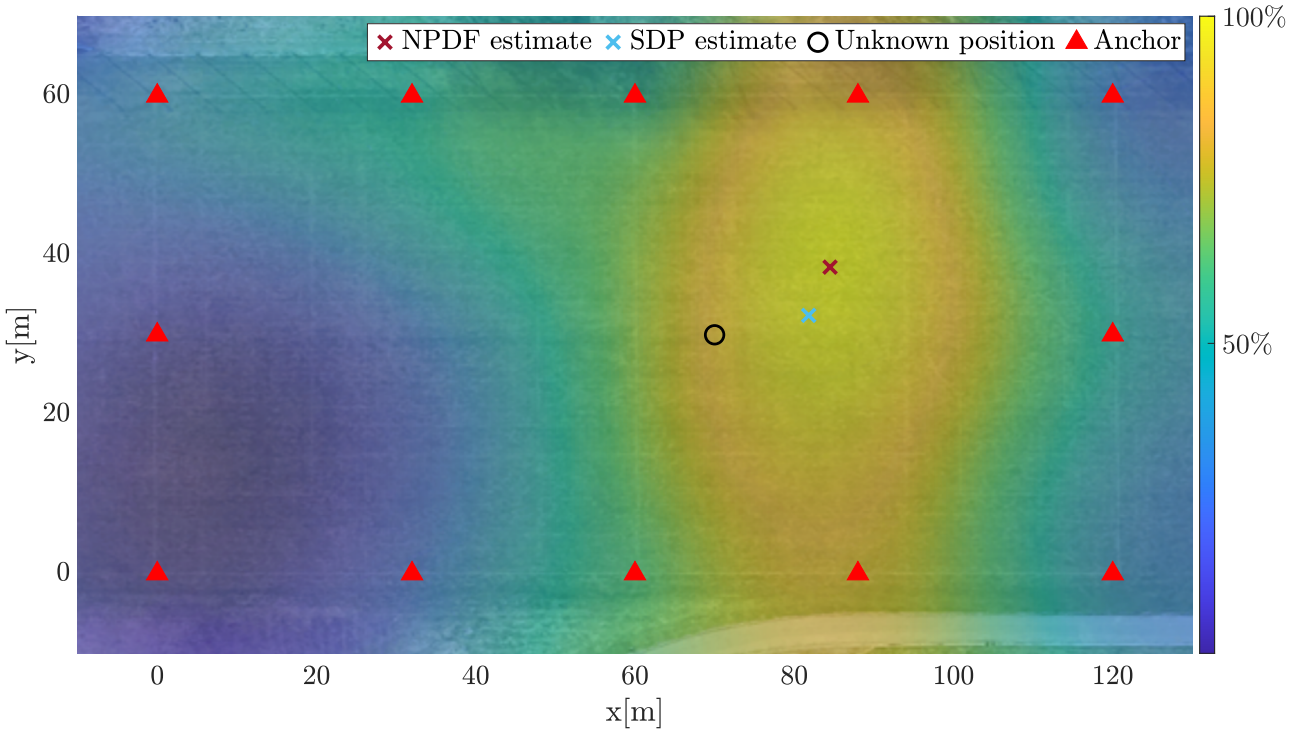


FIGURE 6.13: RELATIVE POSITIONING OF DEVELOPMENT KIT 3

Figure 6.13 shows the results for the unknown position of development kit 3. Here, the SDP position error is 12.1 m and the NPDF position error is 16.8 m. These are higher errors than observed for the first two development kits, showing a clear bias towards to the upper right of the testing area. This can be caused by underestimations (i.e. range estimates less than the true range) from anchors closest to the unknown node, or likewise overestimations (i.e. range estimates greater than the true range) from those anchors farthest away. It is also observed that the anchors towards the bottom left of the pitch have operated well, successfully pushing the estimate away from them. Yet, the shape of the shaded green region implies general underestimation in this sample. Furthermore, the SDP estimate is noticeably better in this test case, implying that the SDP algorithm can handle such biases better than the probability-based method from its feasible region approximation.

Figure 6.14 shows the results for the unknown position of development kit 4. Here the SDP position error is 2.5 m and the NPDF position error is 8.5 m. Again, this result shows a slight bias from the NPDFs towards the top of the testing region, while the SDP algorithm performs excellently.

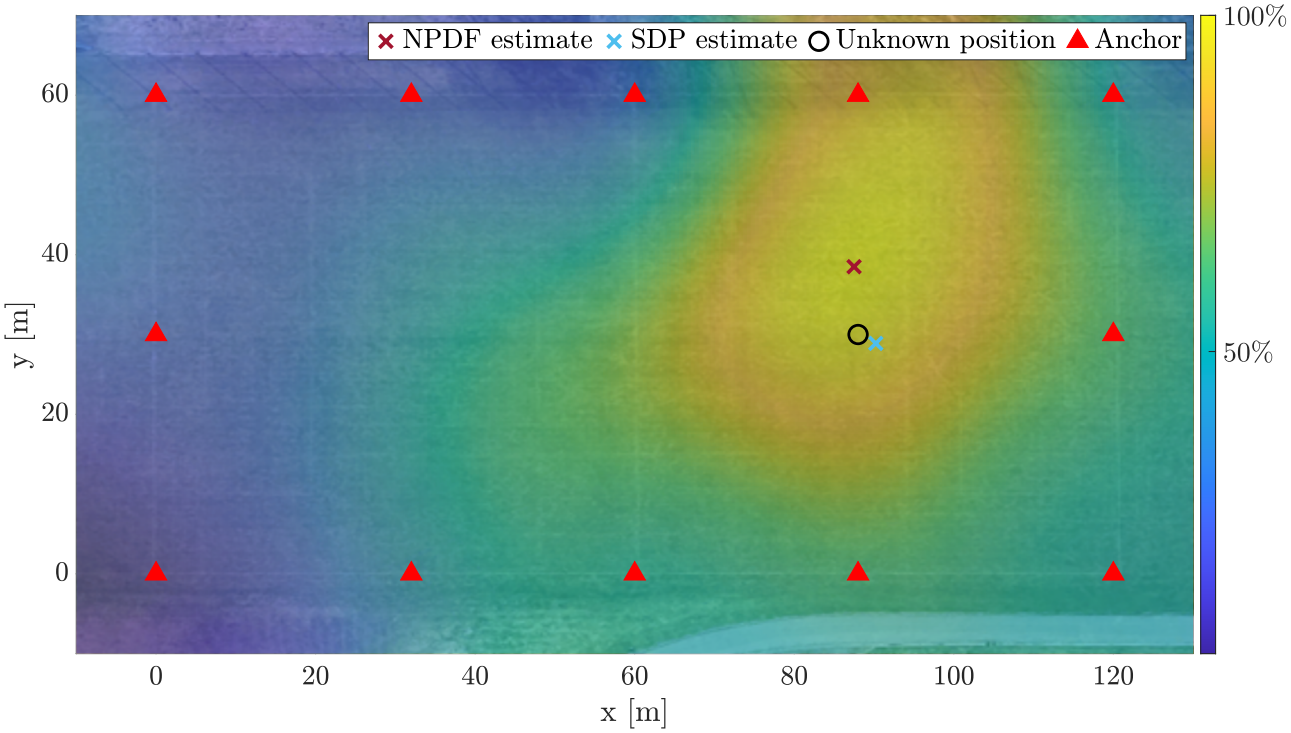


Figure 6.14: Relative positioning of development kit 4

The results for the unknown position of development kit 5 are shown in Figure 6.15. Here the SDP position error is 5.9 m and the NPDF position error is 5.5 m. While both algorithms perform well in this scenario, the region of highest confidence for NPDF is noticeably more distributed in general but with a steeper peak around the estimate. This implies most anchors estimates are accurate but with some outliers. In particular, the range from anchor located at the coordinates $(0, 88)$ appears to be completely anomalous, creating a small confidence region around itself.

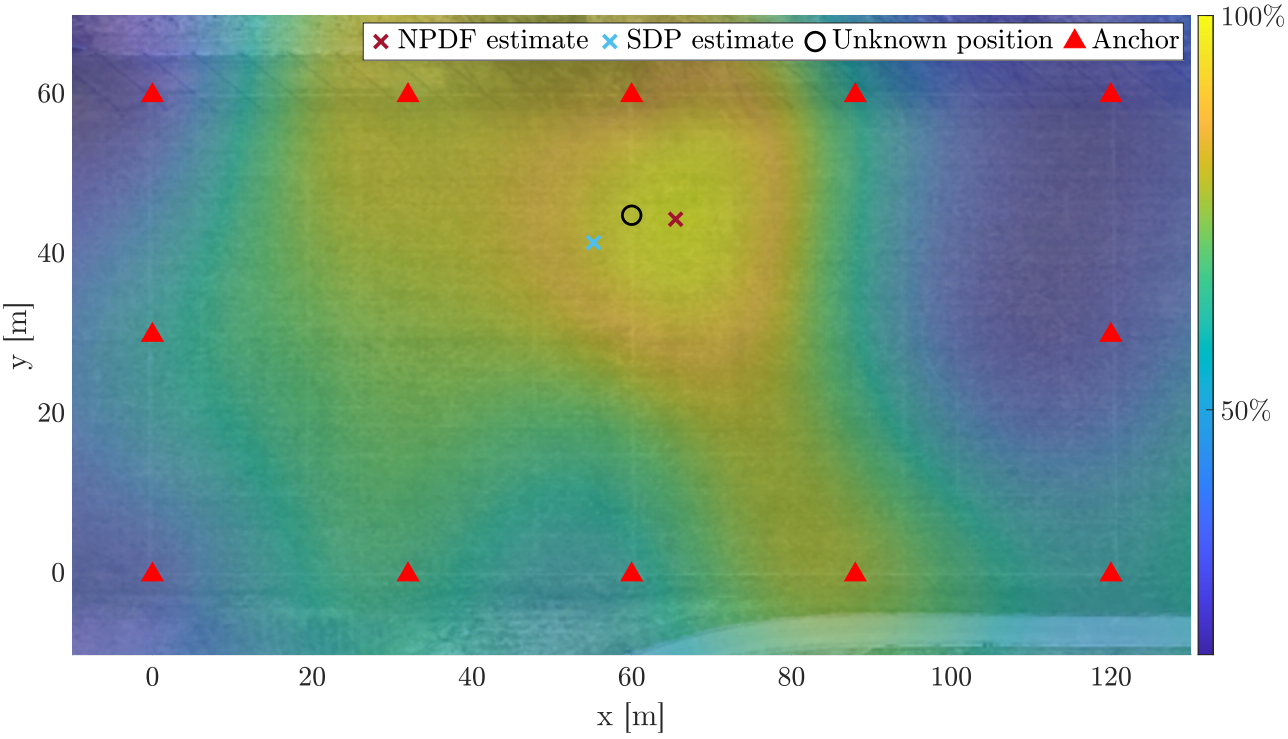


Figure 6.15: Relative positioning of development kit 5

Figure 6.16 shows the results for the unknown position of development kit 6. In this case, the SDP position error is 8.5 m and the NPDF position error is 5.3 m. Both algorithms have worked well to estimate the unknown position in this case, and the region of highest confidence for NPDF is well defined and bounded without any obvious outliers.

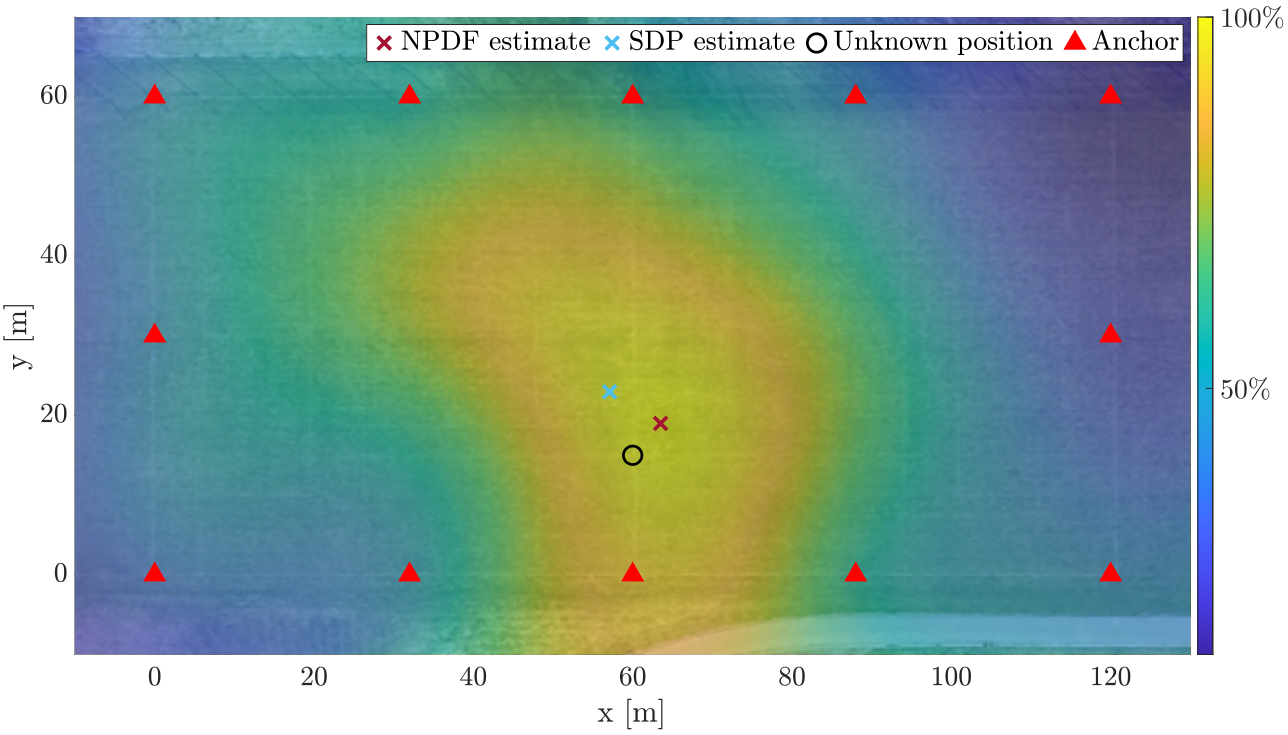


Figure 6.16: Relative positioning of development kit 6

Figure 6.17 shows the results for the unknown position of development kit 7. In this case, the

SDP position error is 25.0 m and the NPDF position error is 19.8 m. This is noticeably the least accurate result out of all ten test cases in both algorithms (with the exception of the SDP anomaly for development kit 10, Fig. 6.20). While the NPDF algorithm has successfully tended the estimate towards the top left of the test area, there is a noticeable bias to the right of the true position. As both algorithms have produced similar estimates, and as the NPDF highest confidence region is well defined, this is most likely the result of inaccuracies across many anchors' range estimates for this test case.

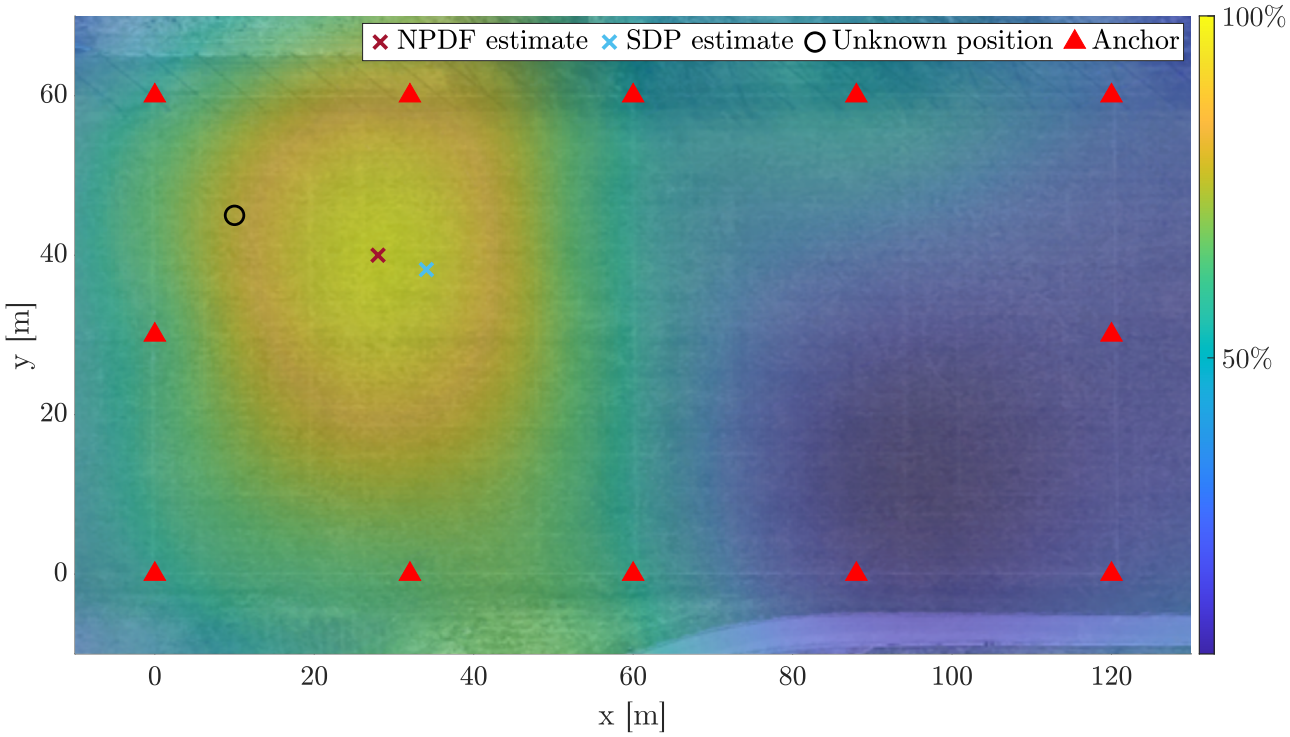


Figure 6.17: Relative positioning of development kit 7

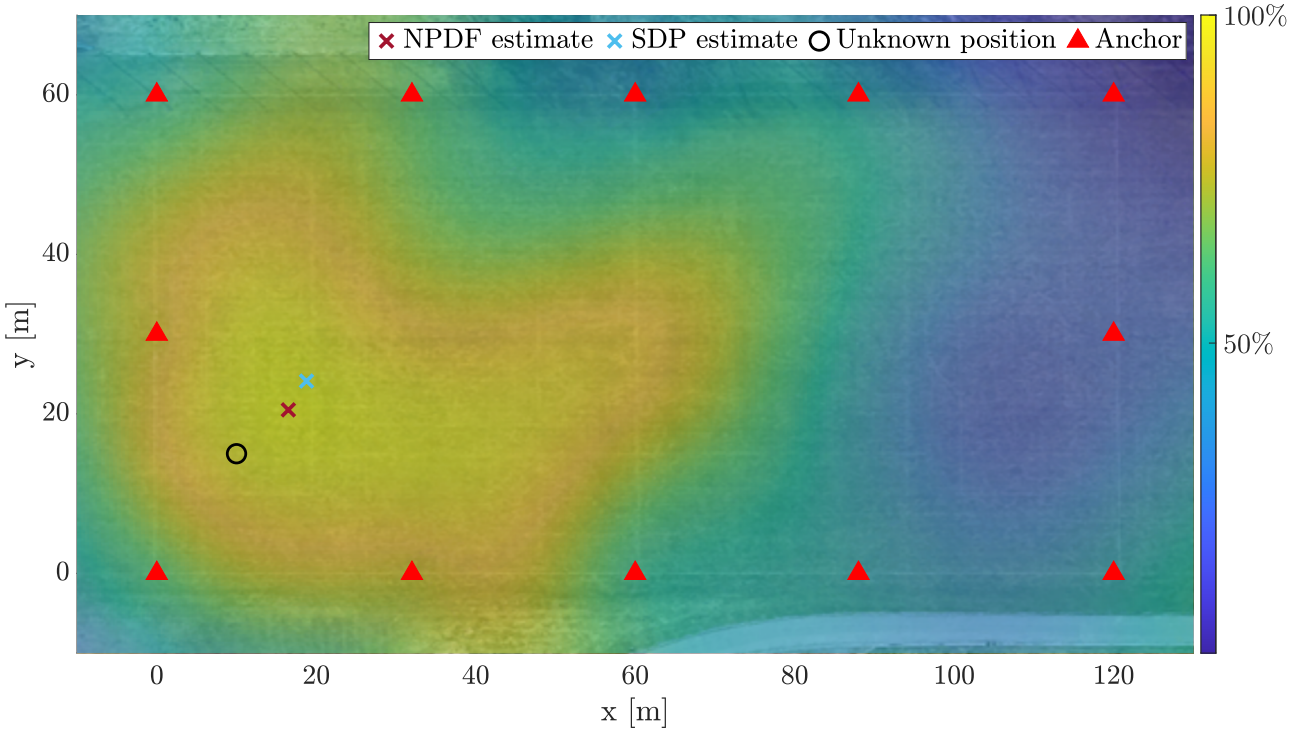


Figure 6.18: Relative positioning of development kit 8

Figure 6.18 shows the results for the unknown position of development kit 8. For this development kit, the SDP position error is 12.7 m and the NPDF position error is 8.5 m. Next, Fig. 6.19 shows the results for the unknown position of development kit 9. Here the SDP position error is 12.9 m and the NPDF position error is 6.0 m. In this case, while the estimates are typical there is noticeably two distinct regions of high NPDF confidence that have emerged. While the NPDF algorithm requires enough regions of intersection emerging to form a unique estimate, it is still noticeable that without additional anchors (or with worse range estimates) this estimate may have been significantly erroneous.

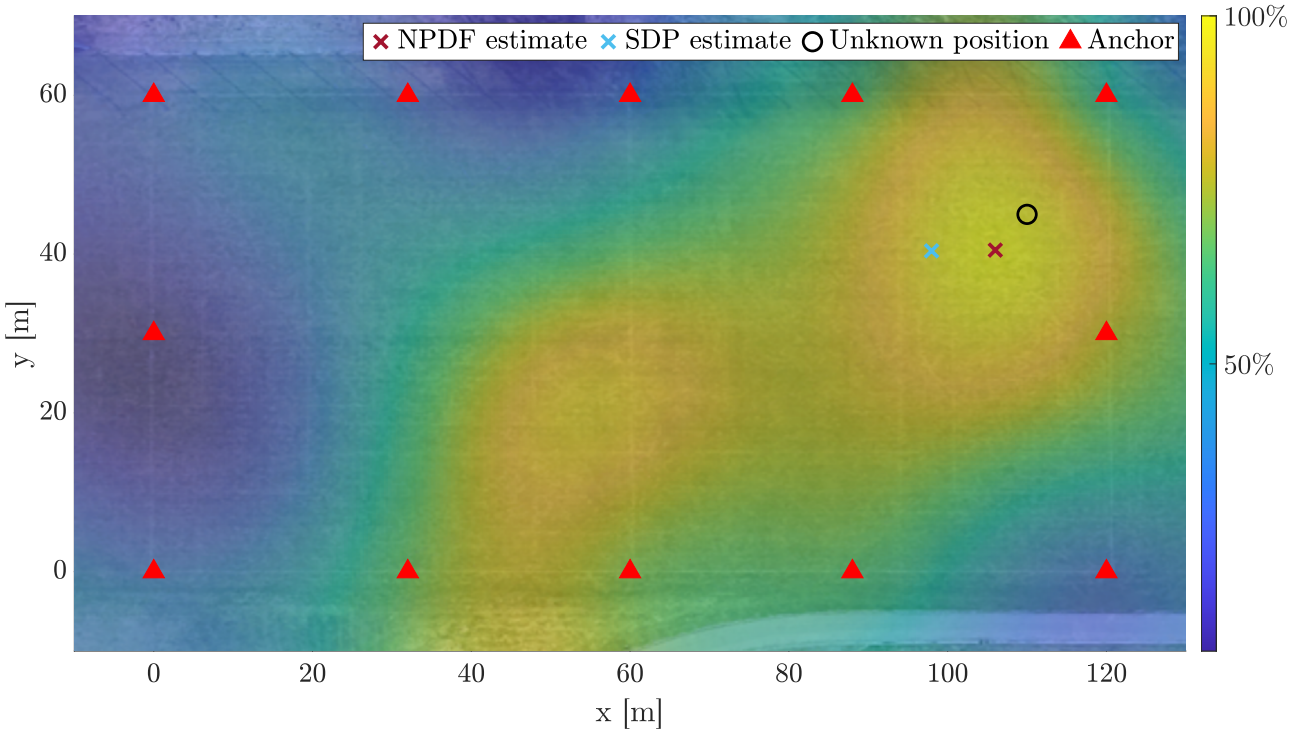


FIGURE 6.19: RELATIVE POSITIONING OF DEVELOPMENT KIT 9

Finally, Fig. 6.20 (overleaf) shows the results for the unknown position of development kit 10. Here the SDP position error is 32.8 m and the NPDF position error is 7.6 m. This is the only instance of the SDP and NPDF results being substantially different. As can be seen from the NPDF distribution, the SDP estimate appears to lie within this region of high probability but has localised far from the true location when optimising for all constraints. The SDP result here can be considered an outlier.

This set of results demonstrates the ability of both algorithmic approaches to localise an unknown development kit relative to many others. The estimates found here are broadly comparable for both algorithms, with the same order of magnitude in positioning error. The SDP algorithm had both the lowest and highest positioning error out of all the test cases. The magnitude of the positioning errors for this test are indicative of what is achievable with this many development kits over this length-scale. While the results for development kit 7 are worse than the average performance across all ten cases, only the SDP estimate from development kit 10 can be considered to be completely anomalous out of all the results obtained. This test

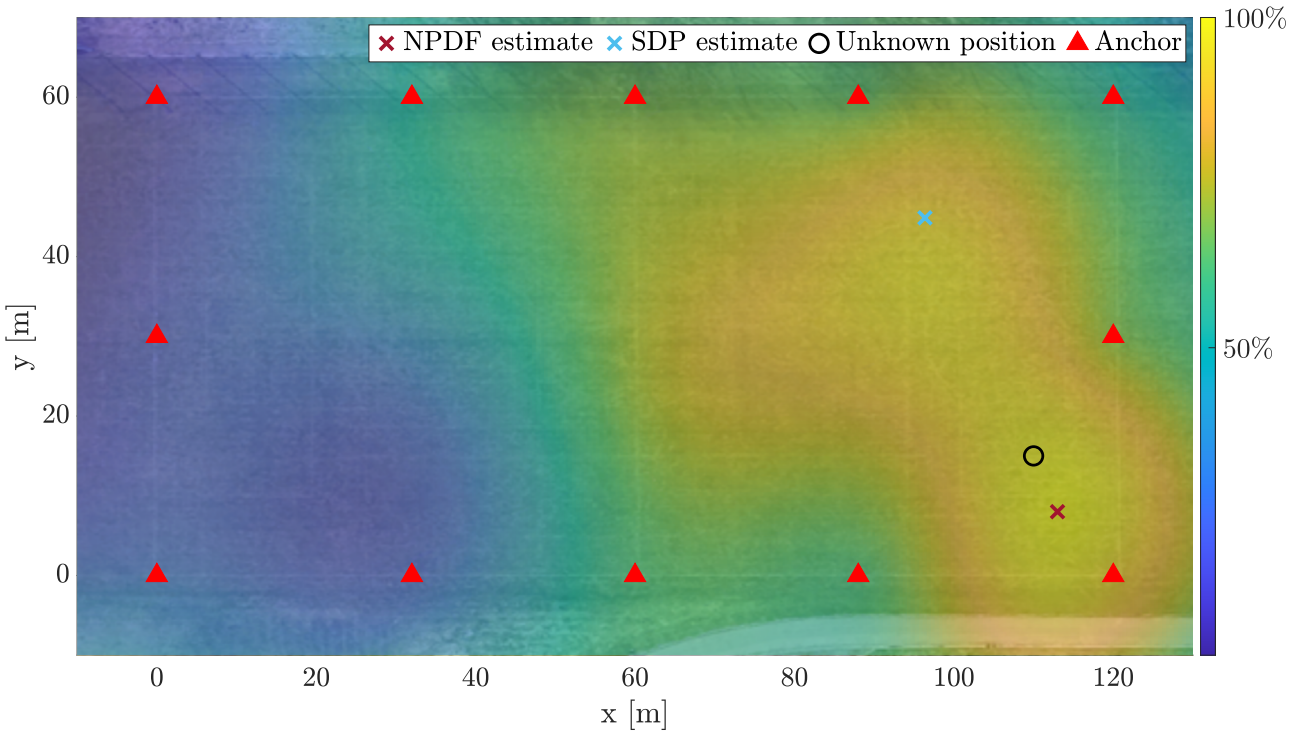


Figure 6.20: Relative positioning of development kit 10

case results show generally high reliability within a few metres of the true locations for most development kits, demonstrating the viability of using RSSI with an experimentally derived PLM in this way.

It is important to note from experimentation that the accuracy of the placement of development kits within the test environment is likely to be slightly less precise than those measured at marker points along the perimeter, due to the placement technique used. Development kits were placed within the testing area using tape measures from known markers measured along the edge and within the pitch. It is estimated that the true locations provided for estimates of positioning error are accurate to within ± 1 m based on this placement technique.

RSSI data is being shared rapidly throughout the network. In this case, full network RSSI data for the 23 development kits is obtained within 3 seconds. As this is a static demonstration, it would be possible to take an average of RSSI values obtained over a period of time, or to identify outlier RSSI values caused by environmental noise or random operational errors by comparing these values with those obtained in the time steps around it. However, as discussed in Chapter 4, the approach in orbit for the purposes of navigation or locating measurement points would be to filter the results with time, which would substantially alleviate the effect of random anomalous results in practice. As a proposed method, dynamic femto-spacecraft with a sufficient sampling rate (which could involve RSSI averaging or error detection methods such as identification of outlier RSSI data) would be robust against outlier estimates from algorithms, and in general improve the relative positioning performance.

A key strength of the SDP algorithm is its ability, in principle, to make use of additional range

constraints between femto-spacecraft of unknown position, helping to improve the position estimates as has been demonstrated in simulation. However, experimentally, the addition of these constraints was not found to improve positioning accuracy in this test case. This is most likely due to testing with a relatively small number of devices with a high range measurement error, whereby anomalous range estimates have a more significant impact statistically on the positioning performance. As the effect of errors is mitigated against by having more devices, it can be expected that testing with a larger number of devices would enable this improvement to be experimentally demonstrated.

For completeness, the full single run test case of the SDP algorithm localising all 10 development kits simultaneously, along with the individual NPDF algorithm test case computational runs, are shown in Fig. 6.21. Full numerical results are listed in Table 6.1. The dashed lines are

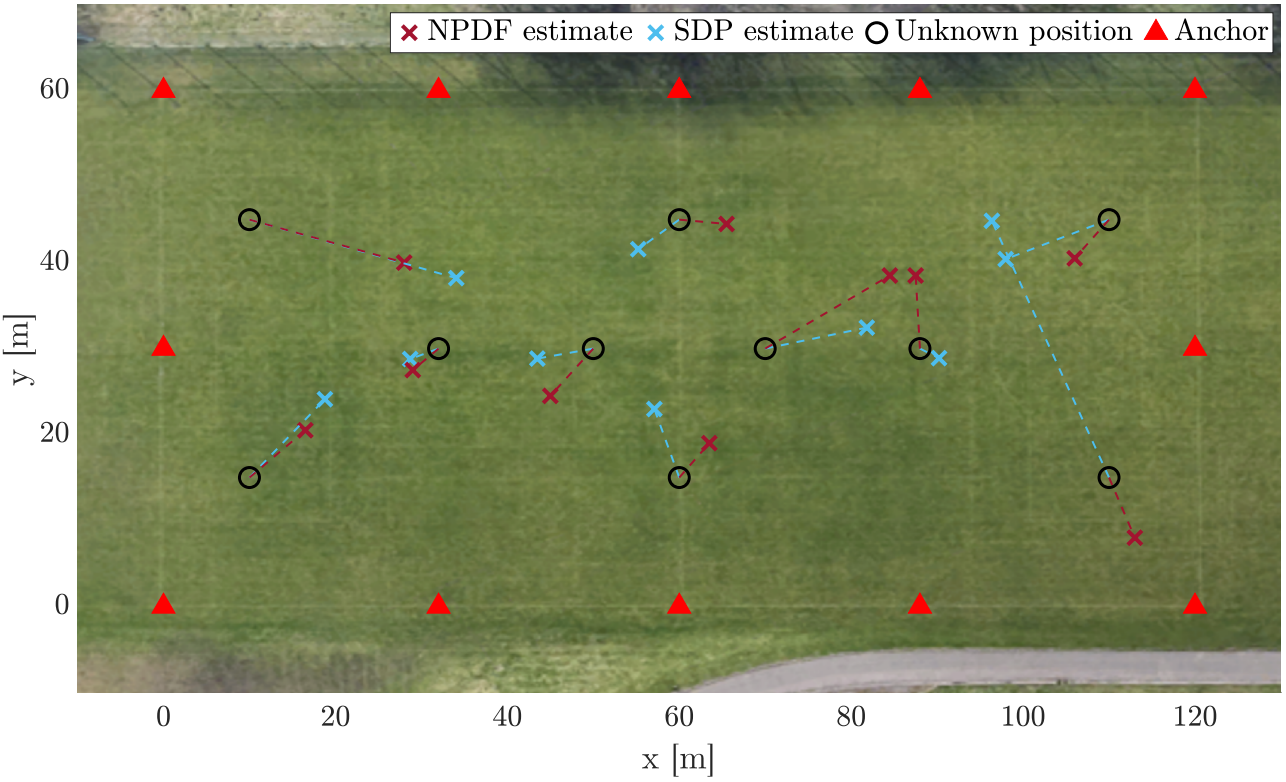


FIGURE 6.21: RELATIVE POSITIONING OF 10 DEVELOPMENT KITS

added to clarify the estimate corresponding to each development kit. From the experimental results obtained for relative positioning outlined in this section, and from the simulated test case results in Chapter 3, it can be anticipated that the accuracy of relative positioning will improve with larger scale tests with a greater number of devices than is possible in this demonstration, offering further insight into the expected performance levels in-orbit for a swarm of femto-spacecraft.

It is noticeable that the four development kits nearest to the edges or perimeter of the test area appear to have worse accuracy than those six development kits nearest the middle. This is an effect that was observed in simulation-based results with the 100 femto-spacecraft test case, where the more central femto-spacecraft had better positioning accuracy. This is understood

Table 6.1: Relative positioning of 10 development kits

Coordinates [m]	SDP estimate [m]	SDP error [m]	NPDF estimate [m]	NPDF error [m]
(32, 30)	(28.7, 28.8)	3.5	(29.0, 27.5)	3.9
(50, 30)	(43.5, 28.8)	6.6	(45.0, 24.5)	7.4
(70, 30)	(81.8, 32.4)	12.0	(84.5, 38.5)	16.8
(88, 30)	(90.2, 28.9)	2.5	(87.5, 38.5)	8.5
(60, 45)	(55.2, 41.6)	5.9	(65.5, 44.5)	5.5
(60, 15)	(57.1, 23.0)	8.5	(63.5, 19.0)	5.3
(10, 45)	(34.1, 38.2)	25.0	(28.0, 40.0)	19.8
(10, 15)	(18.8, 24.1)	12.7	(16.5, 20.5)	8.5
(110, 45)	(98.0, 40.4)	12.9	(106.0, 40.5)	6.0
(110, 15)	(96.4, 44.9)	32.8	(113.0, 8.0)	7.6

to be due to the convex constraints resulting in generally improved performance towards the middle of the swarm. Again, it can be expected that in larger networks, the femto-spacecraft closer to the perimeter will have higher inaccuracies in position estimates than those nearer the centre.

These results are a clear indication in the viability of using a highly inaccurate range metric, utilised effectively by the nature of the large number of noisy constraints within a highly (in this case fully) interconnected network. However, there are obvious limitations to this approach as seen with the metre-scale positioning errors over this testing area, and the effect that large outlier range estimates have on the localisation results. This would in principle be significantly mitigated against using more femto-spacecraft, corresponding to a quadratic increase in the number of available communication links and therefore range estimates and optimisation constraints. The extent to which increased numbers of femto-spacecraft would improve positioning performance, and at what scale this is no longer a significant factor, is an open question for future testing and implementation. However, it is clear from these results that this would at a minimum reduce the effect of outlier data.

It can therefore be theorised that with increased numbers of femto-spacecraft in practice, and with an improved range metric (either in resolution or measurement technique), that this positioning performance could be improved further beyond what is possible with this test approach and what has been experimentally demonstrated here using RSSI statically over this available area in 2D.

6.3 Relative Navigation Demonstration

In this section, the development kits are now placed in known locations on the sports pitch that are calculated to represent a static snapshot of a swarm's formation in LEO at a particular point in time, according to the relative dynamics of the trajectories of femto-spacecraft dispersing from a carrier spacecraft. These positions are determined using the same Clohessy-Wiltshire (CW) model of the swarm relative dynamics outlined in Chapter 4 and applying a

2D projection of the motion at realistic length-scales for the swarm as it initially disperses in orbit. The implications of the experimental results obtained are then discussed for future in-orbit applications.

6.3.1 SEQUENTIAL SWARM DEPLOYMENT DEMONSTRATION

While these tests are still static in nature and limited to 2D, equivalent to those in Section 6.2, this procedure furthers the demonstration with consideration of femto-spacecraft separation distances. Further testing in the future could make use of this approach with dynamic femto-spacecraft proxies. Using the closed form CW model, in this test the scenario of a swarm being ejected sequentially in a controlled manner from a single carrier spacecraft is considered. The other method considered in Chapter 4 of randomly and instantaneously dispersing the swarm from the carrier in all directions is not tested here. This random approach is more equivalent to the relative positioning demonstration in the previous section.

The swarm could be deployed in many directions to trail and lead the carrier along track with varying cross-track and radial components in the ejection velocity. In this test case, the swarm is deployed along two separate relative trajectories, one trailing and one leading the deployer, enabled by the following conditions. With a deployer in a circular LEO of altitude 400 km, and making maximum use of the testing area available for this demonstration, consider the case of two femto-spacecraft initially coincident in position with the deployer and with a constant ejection velocity of a realistic order of magnitude. The initial conditions of:

$$\mathbf{r}_{0_1} = \mathbf{r}_{0_2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{m} \tag{6.12}$$

$$\mathbf{v}_{0_1} = \begin{bmatrix} 0 \\ 2 \\ 30 \end{bmatrix} \times 10^{-3}\,\text{m}\,\text{s}^{-1} \tag{6.13}$$

$$\mathbf{v}_{0_2} = \begin{bmatrix} 0 \\ -2 \\ -30 \end{bmatrix} \times 10^{-3}\,\text{m}\,\text{s}^{-1} \tag{6.14}$$

for the CW motion permit taking a 2D projection of the motion of these two femto-spacecraft in the yz-plane, as shown in Fig 6.22, where it can be seen that the testing area accommodates the trajectories of these femto-spacecraft over 1.5 orbital periods with respect to the deployer. With 22 development kits (including one placed at the origin representing the deployer), by deploying a femto-spacecraft approximately every 12.5 minutes, the swarm at that point in time would take the form as shown by the development kit markers in Fig. 6.22. Development kits were placed in this configuration in the testing environment by measuring their locations from the vertices of sports pitch markings that had been marked out on the day of testing. This enables

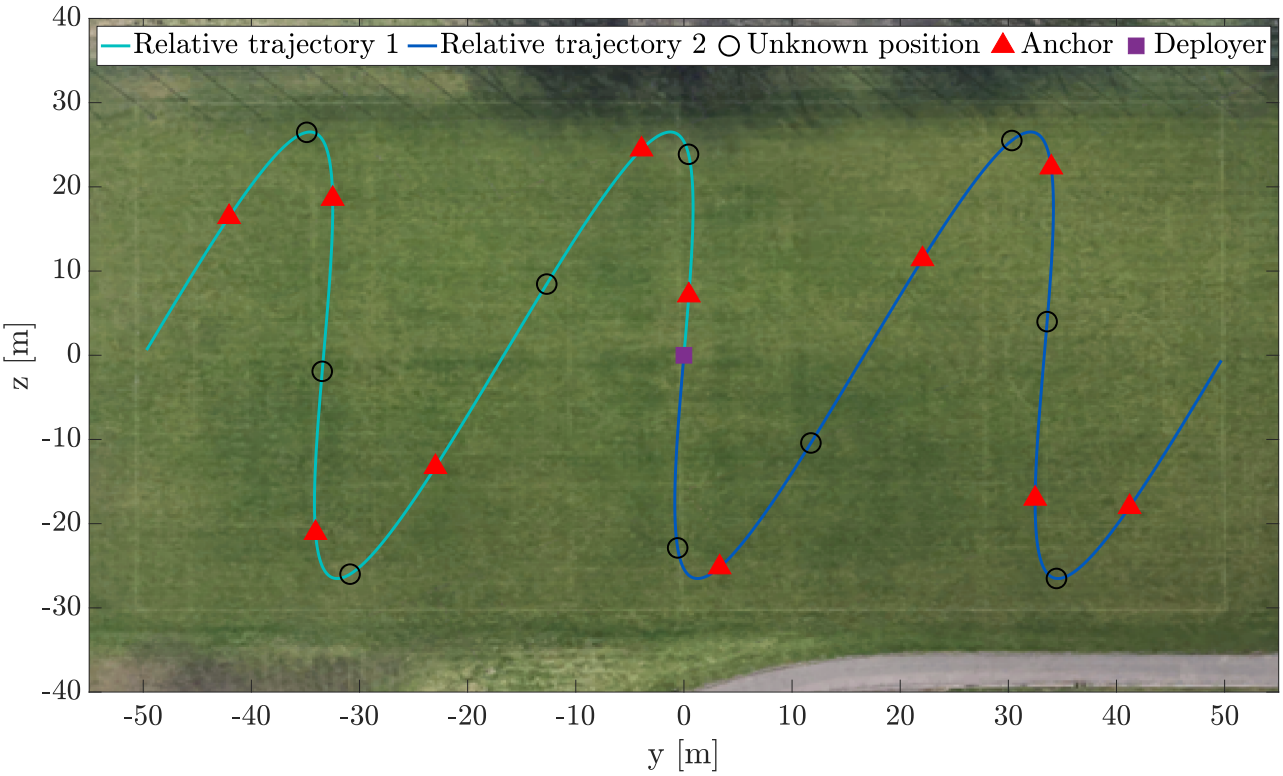


FIGURE 6.22: 2D PROJECTION OF THE YZ-PLANE, SHOWING EACH RELATIVE TRAJECTORY OVER 1.5 ORBITS

a static test of the entire swarm relative navigation at a particular time step, experimentally representative of how this strategy could work in orbit as a particular filter update measurement sample. Again, the unknown development kits are marked by black rings while known anchor positions are marked by red triangles. In this case, anchors are deployed alternately with femto-spacecraft of unknown position. Results of this test are shown in Fig. 6.23, again using both the SDP and NPDF algorithms to localise 10 of the unknown development kit positions with 12 anchors. Full numerical results are listed in Table 6.2.

TABLE 6.2: RELATIVE NAVIGATION OF 10 DEVELOPMENT KITS

Coordinates [m]	SDP estimate [m]	SDP error [m]	NPDF estimate [m]	NPDF error [m]
(-35, 26.5)	(-28.8, 23.6)	6.8	(-20.0, 23.5)	15.3
(-33.5, -2)	(-32.6, -1.5)	1.0	(-20.0, -8.0)	14.8
(-31, -26)	(-27.3, -17.5)	9.2	(-20.0, -20.0)	12.5
(-13, 8.5)	(-10.7, 4.1)	5.0	(-13.5, 4.5)	4.0
(0.5, 24)	(-5.4, 9.6)	15.5	(-4.0, 16.5)	8.7
(-0.5, -23)	(-10.2, -9.9)	16.3	(-9.5, -13.5)	13.1
(12, -10.5)	(14.0, -5.2)	5.7	(12.5, -3.5)	7.0
(30, 25.5)	(25.9, 15.0)	11.3	(33.0, 14.0)	11.9
(33.5, 4)	(25.8, 1.5)	8.1	(25.0, 2.5)	8.6
(34.5, -26.5)	(21.5, -14.5)	17.6	(23.5, -14.0)	16.7

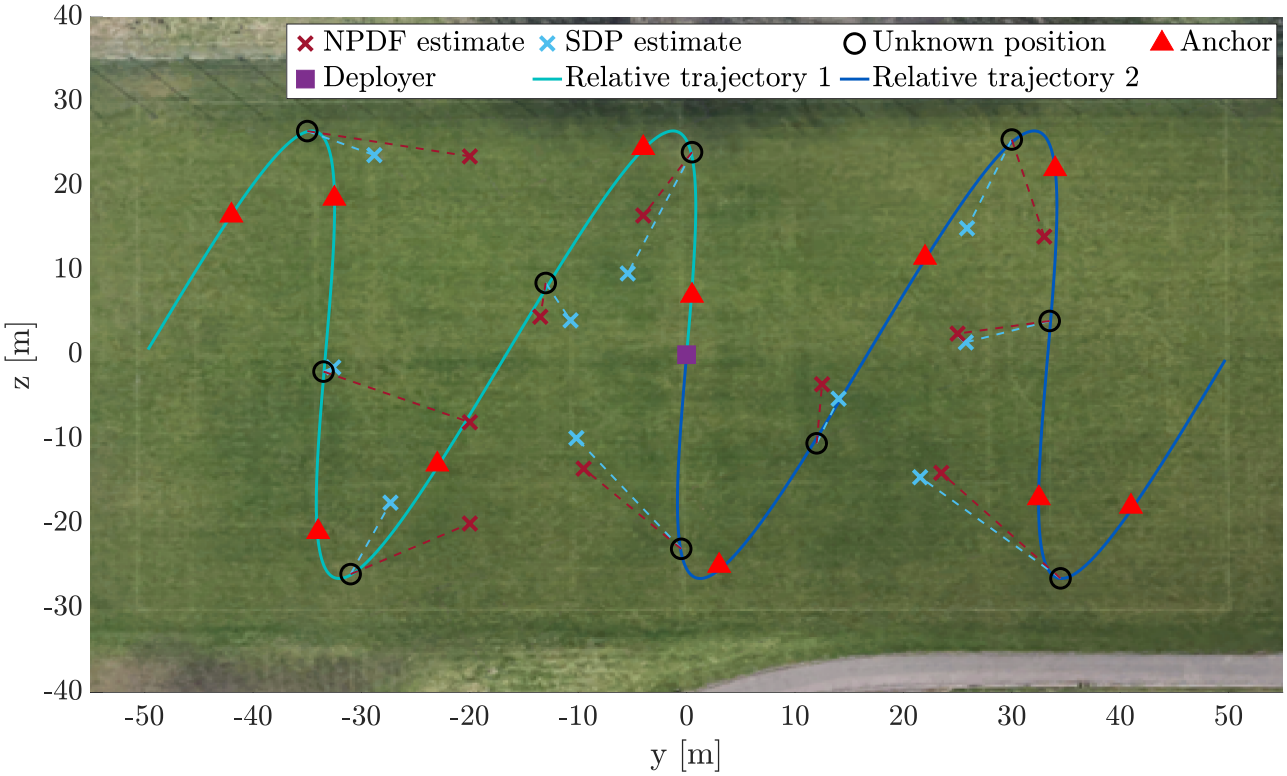


FIGURE 6.23: RELATIVE NAVIGATION ALONG TWO CW TRAJECTORIES

This result demonstrates the algorithms working for realistic spatial length-scales for a swarm being deployed in LEO. In the results here and in Section 6.2, the number of unknown positions and anchors has been kept the same for a comparative demonstration. In practice, with larger networks of more femto-spacecraft, the proportion of anchors in the swarm as a percentage of all spacecraft could be much lower than can be experimentally demonstrated with a smaller network. The positioning accuracy is of the same order of magnitude as with the relative positioning test case, with the SDP algorithm performing slightly better in most cases, with the best performance again witnessed in the centre of the swarm geometry.

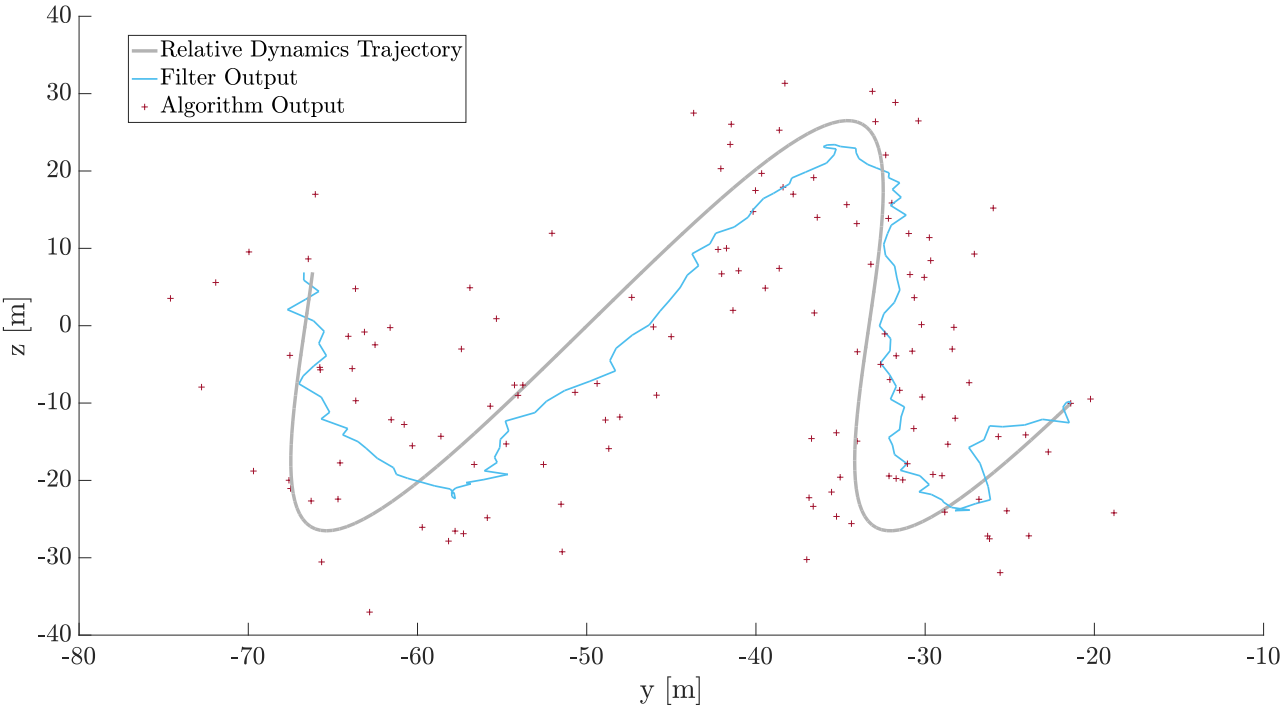
6.3.2 IN-ORBIT IMPLICATIONS

The position errors from the results of the test cases in Sections 6.2 and 6.3 vary considerably, from as low as 1 m to as much as 33 m. The mean error (σ_X) for the relative positioning test in Section 6.2 is 15.3 m for the SDP algorithm and 10.2 m for the NPDF algorithm. For the relative navigation test in Section 6.3, the mean error is 11 m for the SDP algorithm and 11.9 m for the NPDF algorithm. From the simulation-based results of the SDP algorithm in Chapter 3, experimental performance indicated here is comparable to a noise level ζ of between approximately 8-17.5%. It is therefore possible to extrapolate the expected in-orbit relative navigation performance by simulating noisy range estimates at this level for this size of swarm. As demonstrated in Chapter 4, the filtering of discrete estimates over time will in general improve relative navigation performance.

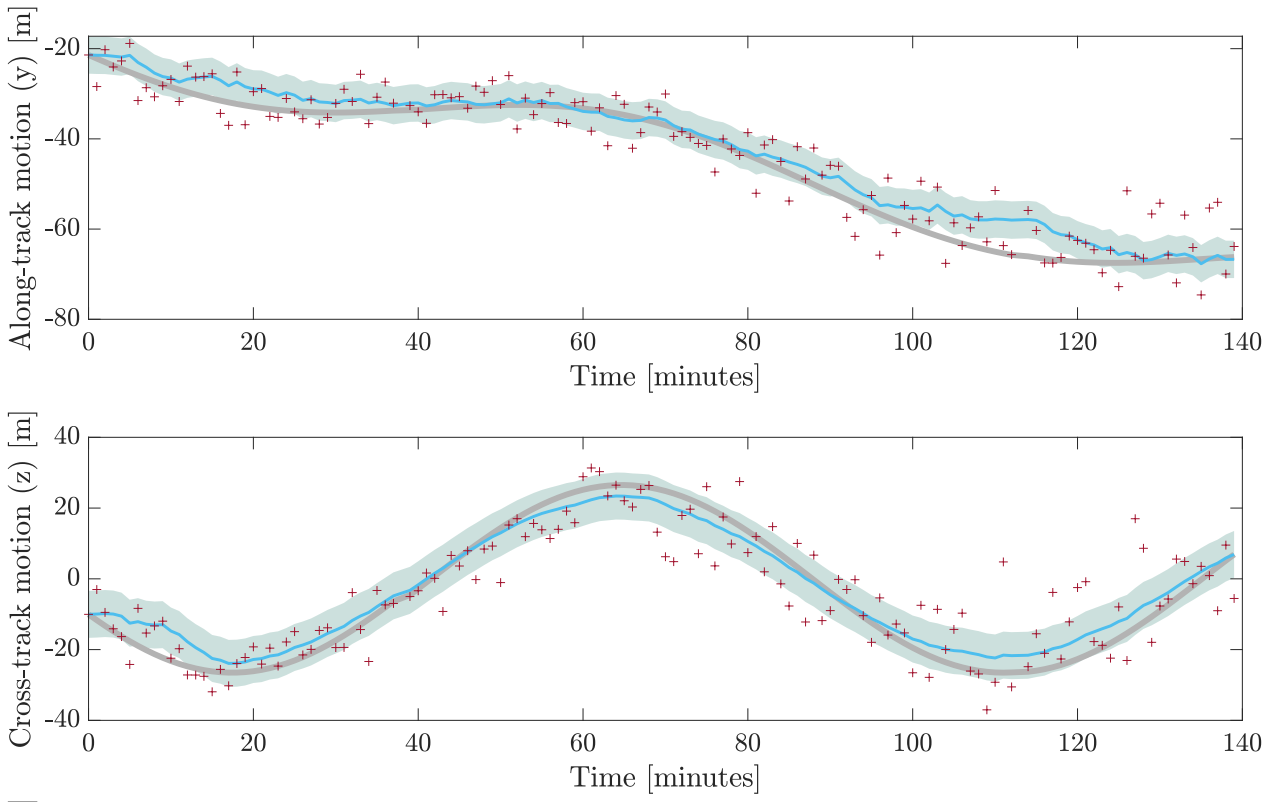
A limitation of this approach is that the PLM and range estimates used in the test cases clearly

indicate non-Gaussian behaviour in outlier RSSI data, partially due to greater variability at larger ranges. It would appear that large outliers can affect particular random estimates in a way that cannot be simulated with a particular additive white Gaussian noise level. It would of course be possible to artificially introduce non-Gaussian behaviour to the measurement model in simulation. However, the impact of outlier range estimates would be significantly weakened algorithmically as the swarm size increases, and it is also difficult to estimate the prevalence and magnitude of such variations in RSSI values. Instead, if the noise level in simulation is set to 20% to compensate for these effects, results can provide an indicative performance of relative navigation in-orbit for this particular scenario. Figure 6.24 (overleaf) shows the results of the Kalman filtering of a femto-spacecraft over 1.5 orbits, using the same simulation parameters and sample rate from Chapter 3 of position estimates every 60 s. Only the 2D projection of the yz-plane is shown for comparison with the experimental results.

The results of this simulation indicate the feasibility of in-orbit relative navigation for a swarm of this size deployed over this length-scale, using representative range-based measurement inaccuracies of what was obtained experimentally. Larger networks of spacecraft in practice may be able to improve relative positioning performance beyond what was experimentally demonstrated. While these results are indicative of in-orbit performance, the relationship between RSSI and range terrestrially would be expected to differ from the relationship in-orbit. As discussed in Future Work (Section 8.2), there are several ways the PLM could be adaptively tuned for in-orbit use. Additionally, analysis and simulation of the space-based communications environment may provide further insights.



(A) 2D VIEW



(B) INDIVIDUAL AXES VIEW

FIGURE 6.24: SIMULATION-BASED RELATIVE NAVIGATION OF A FEMTO-SPACECRAFT

6.4 Chapter Summary

This chapter has presented the experimental methods and results used to demonstrate relative navigation for a swarm of femto-spacecraft in this thesis. By combining the algorithms, insights from simulation-based results and network protocol development throughout Chapters 3-5, relative positioning has successfully been demonstrated.

The path loss model development in 2D using development kits with V-dipole antennas highlighted the large variability in raw RSSI data. This indicates the limitations of RSSI's utility as a range metric while providing a highly effective proximity metric as tested with the convex optimisation and normal probability density function fusion approaches.

The experimental results of relative positioning and navigation demonstrate the utility of RSSI data as a range metric in highly connected networks. Through utilisation of the path loss model within the testing environment and implementation of suitable algorithms for this testing, relative positioning has been experimentally demonstrated terrestrially for a small network of just over 20 development kits acting as femto-spacecraft proxies using RSSI data alone. The results demonstrate the viability of this method, through fusing coarse constraints to provide a relative positioning system for a swarm of femto-spacecraft entirely via networking. This is a principal advantage of this specific approach, in that any positioning information that can be estimated is found through normal swarm operations and without any additional sensor technology.

Chapter 7

Distributed In-situ Sensing using Femto-spacecraft

This chapter presents the results of an investigation into the use of femto-spacecraft swarms for distributed in-situ sensing in space. This implementation would in practice make use of the relative navigation strategies from previous chapters to demonstrate the utility of femto-spacecraft positioning when taking sensor measurements. Firstly, the background of how femto-spacecraft could be used for massively parallel distributed sensing is presented in terms of the applications and operations. Then, a methodology is presented for characterising phenomena using multiple femto-spacecraft with coarse positioning and sensor measurements. Operational trade-offs are then explored in the number of femto-spacecraft and the accuracy of their positional knowledge and sensor measurements. Finally, this is applied to a demonstration in simulation of scalar and vector field reconstruction, and of the Earth's magnetic field in LEO.

7.1 Background

Many physical phenomena in the near-Earth space environment are characterised by significant spatial and temporal dynamics and ambiguities. While satellite clusters can be utilised to sense such phenomena in a distributed manner, clouds of femto-spacecraft equipped with MEMS sensors could be used as a new sensing strategy approach. This architecture would deliver massively parallel simultaneous multi-point sensing, facilitate sampling at a range of spatial length-scales and could help deconvolve spatial and temporal effects. In this chapter, the qualitative and quantitative trade-offs in utilising thousands of femto-spacecraft dispersed from a carrier platform to improve the measurement of phenomena across the volume of space that the cloud forms is analysed.

This sensing strategy contrasts with traditional satellite cluster architectures, which use a relatively small number of large and expensive satellites equipped with customised precision instrumentation for sensing and relative navigation, although offer far fewer simultaneous measurements [124, 125]. Therefore, it is necessary to examine how the performance trade-offs scale as the number of sensor nodes increases with lower quality measurements in several use cases. As discussed in Chapter 1, the implementation of swarms could offer many operational benefits

over traditional distributed space systems. This includes robustness and redundancy against failure, as individual femto-spacecraft are not essential to the swarm's operation; the potential for sensor fusion; the capacity for modular and dis-aggregated operations; a substantial increase in spatial resolution. If these benefits can be utilised for sensing applications, swarms of femto-spacecraft could offer specific functional benefits as a low-cost and high-risk alternative to these standard approaches.

Phenomena in the near-Earth space environment such as the geomagnetic and gravitational fields, and the evolution of atmospheric parameters, can exhibit dynamic and ambiguous spatial and temporal behaviours [126, 127]. This makes such phenomena difficult to characterise with a single satellite. Massively parallel, multi-point sensing can in principle deliver new insights by deconvolving these ambiguities, while facilitating simultaneous sampling at a range of spatial length-scales. In this investigation, a sensing strategy is proposed that would use hundreds or even thousands of femto-spacecraft dispersed from a larger carrier platform. The resulting 'cloud' of femto-spacecraft could form a highly interconnected space-based wireless sensor network (WSN) as has been discussed in previous chapters. Making use of the relative navigation strategies that would be available by employing the developments in Chapters 3-6, individual sensors would be tied to the femto-spacecraft position, adding considerable value to the measurements obtained. In the same way that MEMS sensors on femto-spacecraft would be of lower quality, the relative position of femto-spacecraft would be far coarser than large spacecraft equipped with custom instrumentation. The key benefit is however the vast number of measurements that could be obtained simultaneously, and the new applications that could be enabled [15].

The proposed approach can be summarised as follows. During operations a cloud of femto-spacecraft would disperse into neighbouring orbits to their carrier spacecraft. Over time, this cloud architecture would facilitate sampling at a range of spatial length-scales as the femto-spacecraft drift from the carrier. Sequentially dispersed femto-spacecraft could take measurements at the same point in space but at different times, helping to deconvolve spatial and temporal variations in sensed phenomena such as atmospheric and magnetic field measurements, as discussed in Chapter 2. Such a cloud could also provide measurement at a range of length-scales appropriate for rapidly changing fields where the dynamics of the field could be under-sampled between sensor spacings.

7.2 Multi-point Sensing Trade-off Methodology

To investigate this sensing strategy, the top-level trade-offs between this new strategy and the traditional approach are examined by increasing the number of femto-spacecraft while decreasing sensor and navigation accuracy for the measurement of typical phenomena in the near-Earth space environment, represented by both scalar and vector fields. A trade-off methodology is developed, demonstrated with a simple test function, and then applied to general scalar and vector field use cases for insight and analysis.

7.2.1 GPR model development via a 1D test function

Consider some arbitrary phenomena with uncharacterised behaviour over a region of space. In the absence of any errors, there exists in principle a minimum number of discrete sensing points distributed throughout the region that can sufficiently sample the phenomena to accurately characterise its behaviour. Then, by introducing measurement errors in sensing and position localisation errors, the performance of this characterisation is subsequently impacted for a given number of sensing points. The effect of such errors would be expected to at best degrade or at worst make characterisation impossible with the same number of sensing points. By increasing the number of sensing points distributed throughout the region of interest, the characterisation performance can improve, as more measurements can be used to help bound uncertainties.

Let the following one-dimensional test function, as shown in Fig. 7.1, represent some arbitrary phenomena of interest over the arbitrary length-scale x such that $0 \leq x \leq 10$.

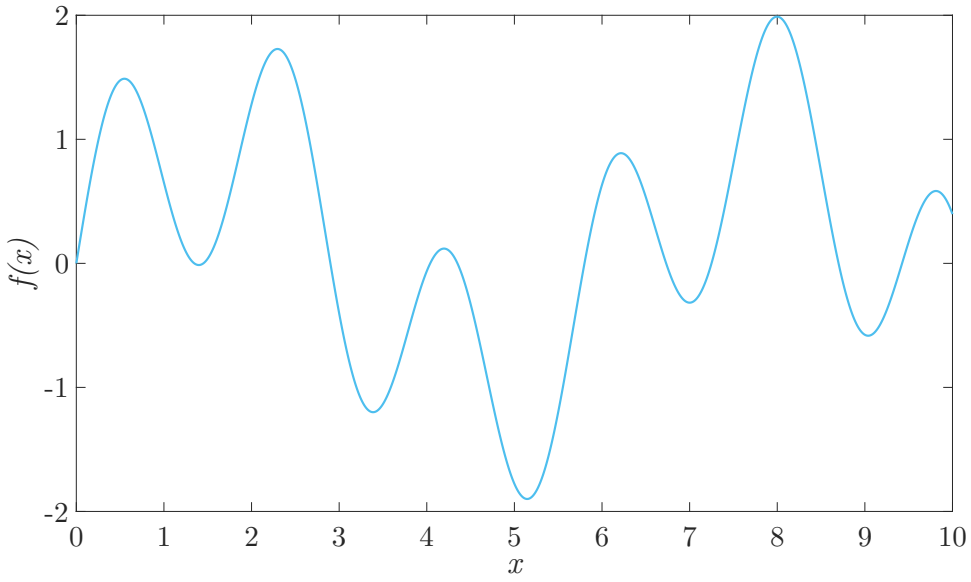


Figure 7.1: Test function $f(x) = \sin(x) + \sin(\frac{10x}{3})$

Discrete measurements can be simulated with a randomly distributed measurement error (δf), 1D location error (δx) and total number of discrete data points (n) across this length-scale, where n corresponds to the number of femto-spacecraft in the cloud. Using Gaussian process regression (GPR), the function behaviour can be modelled from this discrete and noisy data,

generalising the behaviour of the function over the length-scale considered [128]. A Gaussian process is a set of finite random variables with a joint Gaussian distribution. With a GPR model, the behaviour of the function can then be predicted in the locations that have not been sampled for reconstruction (characterisation) of the function over the entire length-scale of interest. This technique, also known as 'kriging' in statistics, is common in machine learning applications, where GPR models can be trained using sets of observations of phenomena. As n increases, characterisation of the phenomena of interest can then improve, as more measurements improve the model's predictive capabilities.

GPR can therefore be used to characterise phenomena across an entire region of interest given measurements taken at discrete points within that region. This technique is effectively using GPR as a prediction model to interpolate values at locations that have not been sampled. This is possible by generating latent variables, that is, those that can be estimated from measured data but which are not directly observed as measurement data. By simulating many different combinations of measurement environments for different test cases (across a range of δf, δx and n), the quality of the reconstructions from each GPR model generated for each test case can be compared with the true function for all combinations across the solution space. The characterisation performance of each model can then be analysed to assess the effect of using greater numbers of less accurate measurements, as in line with the femto-spacecraft swarm philosophy.

For the test function $f(x)$, it is considered that $n = 50$ discrete measurements (observations) are required in the length-scale $0 \leq x \leq 10$ for reconstruction (characterisation) in the absence of any errors (noise-less observations). Then, 100 different test case measurement environment combinations are simulated from low to high δf, δx and n. Across these environments n is varied from 50 to 1000 (by a scaling factor of 20). Measurement and position errors are varied across the noise levels $0.1 \leq \zeta \leq 0.5$ (by a scaling factor of 5) by simulating the observations:

$$\hat{f}(x_i) = f(x_i)(1 + \zeta\chi) \tag{7.1}$$

and:

$$\hat{x}_i = x_i(1 + \zeta\chi) \tag{7.2}$$

where χ is a zero mean normally distributed random variable of variance one. These noise levels are chosen for general comparison across a range of low to high errors, incorporating representative errors as considered in relative positioning simulations from Chapter 3 and from experimental results in Chapter 6. The training sets used to train each of the GPR models is in this case given by the n discrete sensor measurements and position estimates which in simulation are randomly generated. As each set is a Gaussian process, it can be defined by its mean function $m(x)$ and covariance function $k(x, x')$ [128], where:

$$m(x) = \mathbb{E}(f(x)) \tag{7.3}$$

$$k(x, x') = \mathbb{E}((f(x) - m(x))(f(x') - m(x'))) \tag{7.4}$$

where $\mathbb{E}$ is the mathematical expectation (or expected value), defined as the sum of all possible values from a random variable. The Gaussian process can then be denoted by:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{7.5}$$

where the operator ($\sim$) here and in the rest of this chapter represents a parameter being 'distributed according to'. The GPR model is used to predict and generate the function $f(x)$ given new inputs, which in this case is many coordinates x across the entire length-scale considered. Fundamentally, the model enables prediction of $f(x)$ for values of x not sampled. The linear regression model used takes the form [128]:

$$y = f(x, \beta) + \varepsilon \tag{7.6}$$

where x is the independent input variable, f is the function value and y is the response (or target) value from the model. In this model, it is assumed that y varies from the true (unknown) function $f(x)$ due to additive noise ε of the form:

$$\varepsilon \sim \mathcal{N}(0, \sigma^2) \tag{7.7}$$

where $\mathcal{N}$ denotes a normal (Gaussian) distribution and the error variance (σ^2) and coefficient β are estimated using the training set data. Now, implementing latent variables and an explicit basis function $h(x)$, the GPR model is given by:

$$h(x)^T\beta + f(x) \tag{7.8}$$

where an instance of a response y_i can be modelled probabilistically as:

$$P(y_i|f(x_i), x_i) \sim \mathcal{N}(y_i|h(x_i)^T\beta + f(x_i), \sigma^2) \tag{7.9}$$

The covariance $k(x, x')$ can be found by a set of kernel parameters that are used to determine the influence that observations have on the model. In effect, this determines the predictive function smoothness. This overall approach is implemented in MATLAB by simulating discrete noisy measurements of phenomena at random positions and using the *fitrgp* function to train GPR models from this data. In simulation, the performance of a GPR model can be analysed against the true function, but in practice the GPR model (or similar method from the distributed sensing approach) would be all that is available, as the true function would be unobservable. It is therefore of great interest to understand how best to characterise phenomena in general with this approach, which is now examined in simulation.

Figure 7.2 illustrates nine examples across the complete range of these 100 measurement environment test cases simulated using different combinations of δf, δx and n. In the top left, a

small number of accurate sensors are employed (low δf, δx, n), through to the bottom right where a large number of less accurate sensors are used (high δf, δx, n). The true function $f(x)$ is shown in blue, the sample data points are shown in purple and the reconstruction from each GPR model generated for each test case is shown in red. The reconstruction parameters for the nine test cases displayed are listed in Table 7.1. Noise levels in observations of the function $f(x)$ and 1D position x are represented by ζ_f and ζ_x respectively.

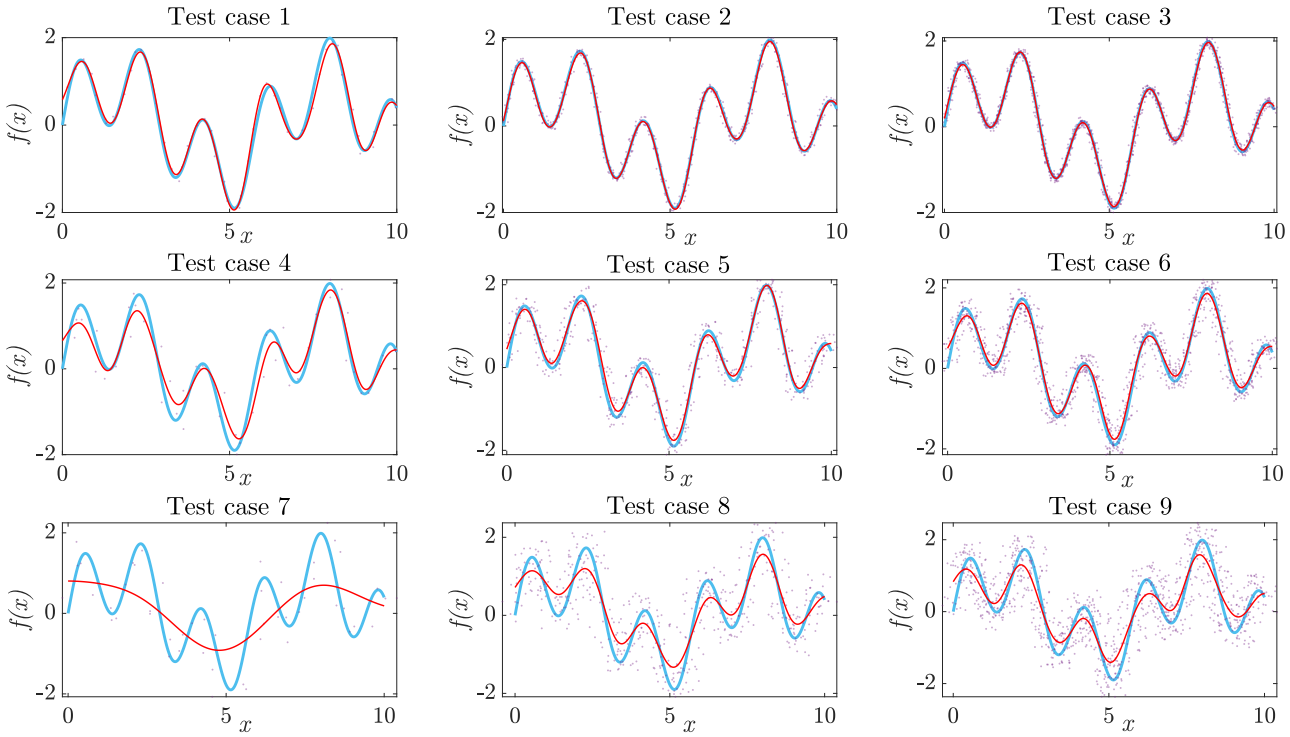


FIGURE 7.2: RECONSTRUCTION OF TEST FUNCTION $f(x)$

TABLE 7.1: RECONSTRUCTION PARAMETERS OF TEST FUNCTION $f(x)$

Test case	n	ζ_f	ζ_x
1	50		
2	500	0.1	0.1
3	1000		
4	50		
5	500	0.25	0.25
6	1000		
7	50		
8	500	0.5	0.5
9	1000		

It is clear from this example that in general terms, increasing the number of sensors distributed throughout a region (and thereby reducing spacing between discrete sample points) improves

reconstruction performance for a given measurement and localisation error level. This is witnessed across the top row within test cases 1-3, where as n increases, the predicted function generated from each GPR model aligns better with the true function $f(x)$. In the absence of errors, it is obvious that more measurements would improve characterisation. It is to be expected that the same applies in general for a given level of measurement and position error, as more measurements help probabilistically to improve GPR model performance.

Intuitively, reconstruction performance worsens as error levels increase for a given number of sensor points. This can be observed in the three columns of test cases in Fig. 7.2, with the combinations of test cases 1, 4 and 9; 2, 5 and 8; and 3, 6 and 9. In each case, the increasing error level worsens general reconstruction performance. As seen in the bottom left plot for test case 7, with a low n and high error combination, this can lead to extremely poor reconstruction performance. Clearly, increasing the number of sensors not only compensates for this by bounding uncertainties for the GPR models, but can also deliver comparable performance to lower n and lower error scenarios. From less accurate measurements, these are compensated for with a sufficiently increased number of measurements.

To quantify and visualise this trade-off, from 100 total measurement environment combinations simulated, a three-dimensional surface plot is generated with the quality of the reconstruction represented in the dependent vertical axis as a discrepancy factor, and the two independent variables of the total number of sensor points and the noise levels in both measurement and position. The discrepancy factor is a performance metric that is calculated based on the level of discrepancy between the true function and the GPR model reconstruction. The lower the discrepancy factor, the better the reconstruction is overall. In this scenario, a discrepancy factor is calculated for each of the 100 measurement environments and normalised between a value of 0 (perfect) and 1 (worst of all the measurement environment samples simulated). The surface plot shown in Fig. 7.3 is generated by displaying the discrepancy factor across the two-dimensional parameter space, then interpolating a multivariate function in reconstruction performance with increasing n and ζ. Individual test cases, or measurement environment samples, are represented by black dots.

This surface plot provides general insight into the performance and trade-offs involved across the combinations of measurement environments. The lowest regions of the surface represent the best expected reconstruction performance. Interpreting Fig. 7.3, for this test function the best performance is seen as the number of sensors increases with the lowest errors, as expected intuitively. In general terms, performance worsens as the total error increases. However, there are many regions where a higher number of less accurate measurements perform comparably with a lower number of more accurate measurements. Broadly, this shows that a higher number of less accurate sensors can 'match' the performance of a low number of accurate sensors. Additionally, this indicates that as the errors in measurement increase, to achieve similar performance, increasing the number of sensors can compensate asymptotically before adding more sensors cannot compensate further, as indicated by tracking the error with n for a given 2D

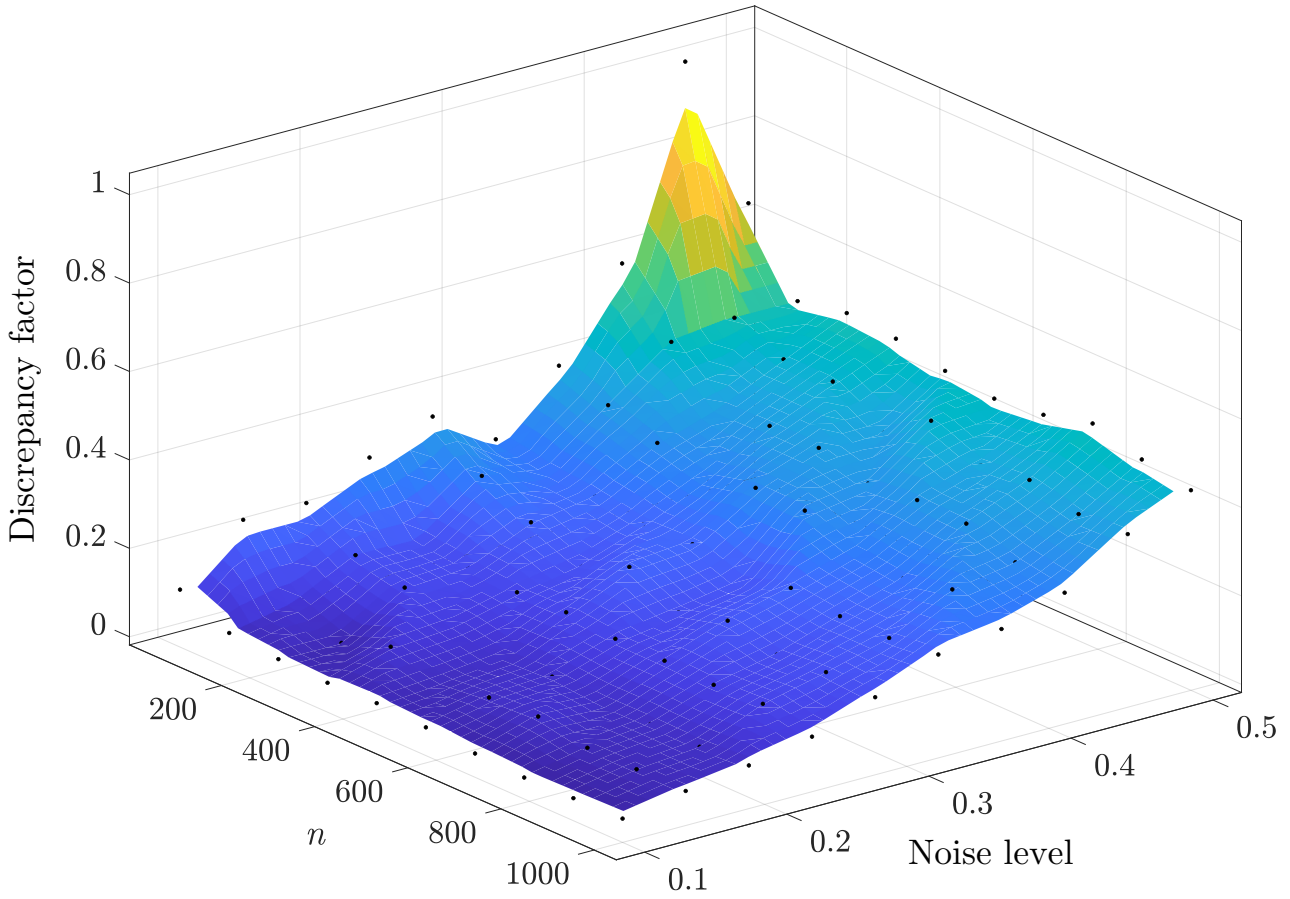


Figure 7.3: Surface plot showing generalised reconstruction performance of test function $f(x)$

discrepancy factor plane.

7.2.2 Scalar field reconstruction

Using the methodology developed above for the test function $f(x)$, the reconstruction of both scalar and vector fields is now considered, representative of key phenomena of interest in the near-Earth space environment, for example, investigating the upper atmosphere or elements of the gravitational and geomagnetic fields [124, 127]. The approach is extended to scalar and vector fields using a series of simple test models. In simulation, sensors are randomly dispersed throughout the region of space to be sampled. This is representative of a femto-spacecraft cloud diffusing through space and as it evolves, dispersing further from the carrier spacecraft. Again, the number of femto-spacecraft, and the errors in sensor measurement and positioning are varied across test cases.

For the field reconstruction procedure using GPR, for these applications, sparse, distributed and noisy measurements of scalar and vector properties are input, and then GPR models are generated that predict and characterise the behaviour of the field across the entire region sampled. From the GPR models developed, the fields are then reconstructed and the results are compared with the true test models, analysing how well this large n, high error strategy

scales in different scenarios.

For scalar field reconstruction a two-dimensional test model presented as an arbitrary scalar field across a region of latitude $0 \leq \phi \leq 90°$ and longitude $0 \leq \theta \leq 90°$ shown in Fig. 7.4. With this arbitrary scalar field, the number of sensors is increased across all simulations again by a factor of 20 (from 50 to 1000), and the measurement and positioning error are varied across the noise levels $0.1 \leq \zeta \leq 0.5$ (by a scaling factor of 5). Figure 7.5 illustrates nine different reconstructions covering this range of measurement combinations, where the sample points are shown in lilac. The reconstruction parameters for the nine test cases displayed are listed in Table 7.2, along with the normalised discrepancy factors for each test case. Noise levels in observations of the function scalar field parameter T, latitude and longitude are represented by ζ_T, ζ_ϕ and ζ_θ respectively.

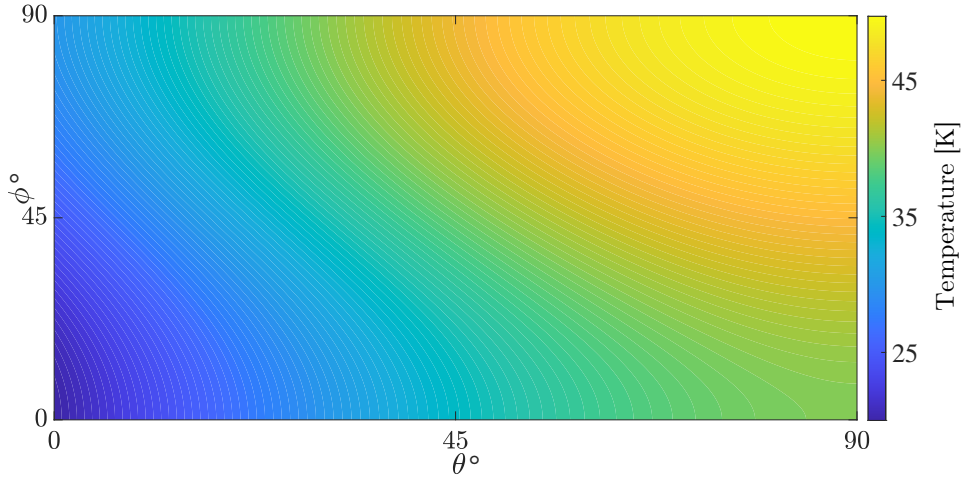


FIGURE 7.4: SCALAR FIELD TEST MODEL $T(\theta, \phi) = 10 \sin^2 \theta + 20(1 + \sin \phi)$

TABLE 7.2: RECONSTRUCTION PARAMETERS OF SCALAR FIELD TEST MODEL

Test case	n	ζ_T	ζ_θ	ζ_ϕ	Discrepancy factor
1	50				0.006
2	500	0.1	0.1	0.1	0.001
3	1000				0.001
4	50				0.526
5	500	0.25	0.25	0.25	0.049
6	1000				0.122
7	50				0.813
8	500	0.5	0.5	0.5	1
9	1000				0.444

Figure 7.5 indicates a general trend of improved reconstruction as n increases at a given error level, with each normalised discrepancy factor representing the relative performance with

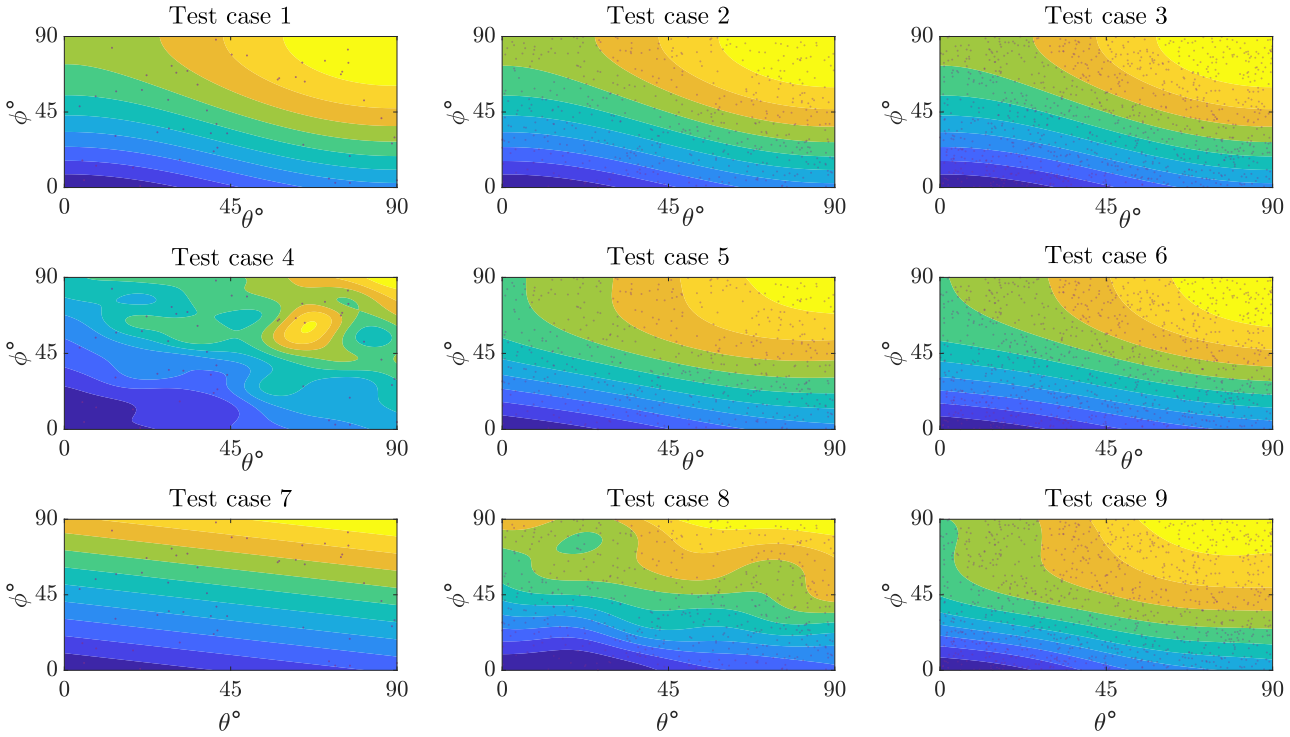


FIGURE 7.5: RECONSTRUCTION OF SCALAR FIELD TEST MODEL

respect to the other test conditions. The lowest discrepancy factors, and therefore best reconstructions, are observed in test cases 1-3, for the lowest errors in measurement and position. Increasing n across the top row of Fig. 7.5 improves the reconstruction, as can be expected for a given error level. The three columns of test cases in Fig. 7.5 demonstrate again that increasing the error level worsens general reconstruction performance for a given number of discrete measurements. This further highlights the ability of many lower accuracy measurement samples to compensate for fewer high accuracy samples across a 2D region of space for scalar quantities.

The worst reconstruction from this random simulation is test case 8, not test case 7 as would be expected in general with an average of many simulated test cases. These random cases show that significantly increasing the number of distributed sensors compensates for cases where measurement inaccuracies can result in high sensitivity to characterisation errors. The comparable performance between test cases 4 and 9 highlights the trade-off: 1000 high-error measurements reconstructed the scalar field better than 50 mid-error measurements.

7.2.3 VECTOR FIELD RECONSTRUCTION

For vector field reconstruction, a test model is used representative of a two-dimensional section of an arbitrary vector field in space. The simple field model used is given by $V(\theta, \phi) = (\sin(\phi), \cos(\theta))$, as shown in Fig. 7.6. As with the procedure for the scalar field test model, a series of discrete measurements are taken and used to reconstruct the field, significantly increasing the number of sensors and the accuracy of their measurements across the region of interest, which is $-180 \leq \phi \leq 180°$ and $-180 \leq \theta \leq 180°$.

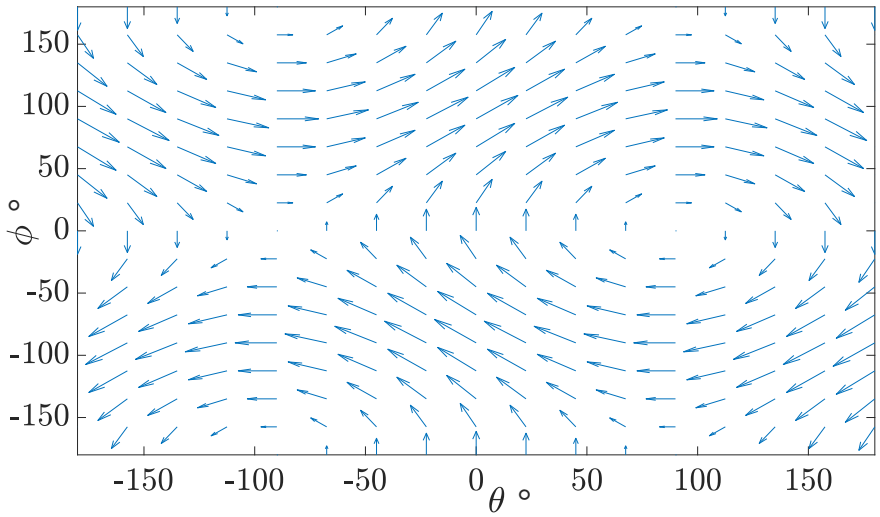


FIGURE 7.6: VECTOR FIELD TEST MODEL $V(\theta, \phi) = (\sin(\phi), \cos(\theta))$

In this case the sensor is measuring both magnitude and direction of a quantity. For illustration, 50 measurements of this field are illustrated in Fig. 7.7, showing an example of the type of discrete data the GPR model uses to generate general characterisations across the entire region the sensors are scattered within.

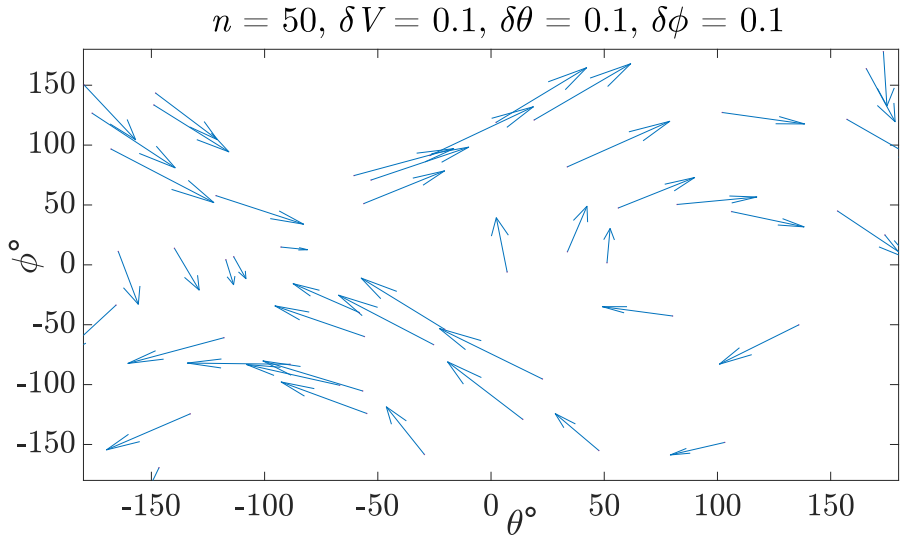


FIGURE 7.7: 50 MEASUREMENTS OF $V(\theta, \phi)$

Nine measurement environment test case reconstructions are shown in Fig. 7.8, again with the number of sensors increased by a factor of 20 and the measurement and positioning error noise level by a factor of 5. Discrete measurement points are shown by the lilac dots, with the field reconstruction overlaid. Note that for clarity, unlike in Fig. 7.7, the discrete vector measurements are not displayed.

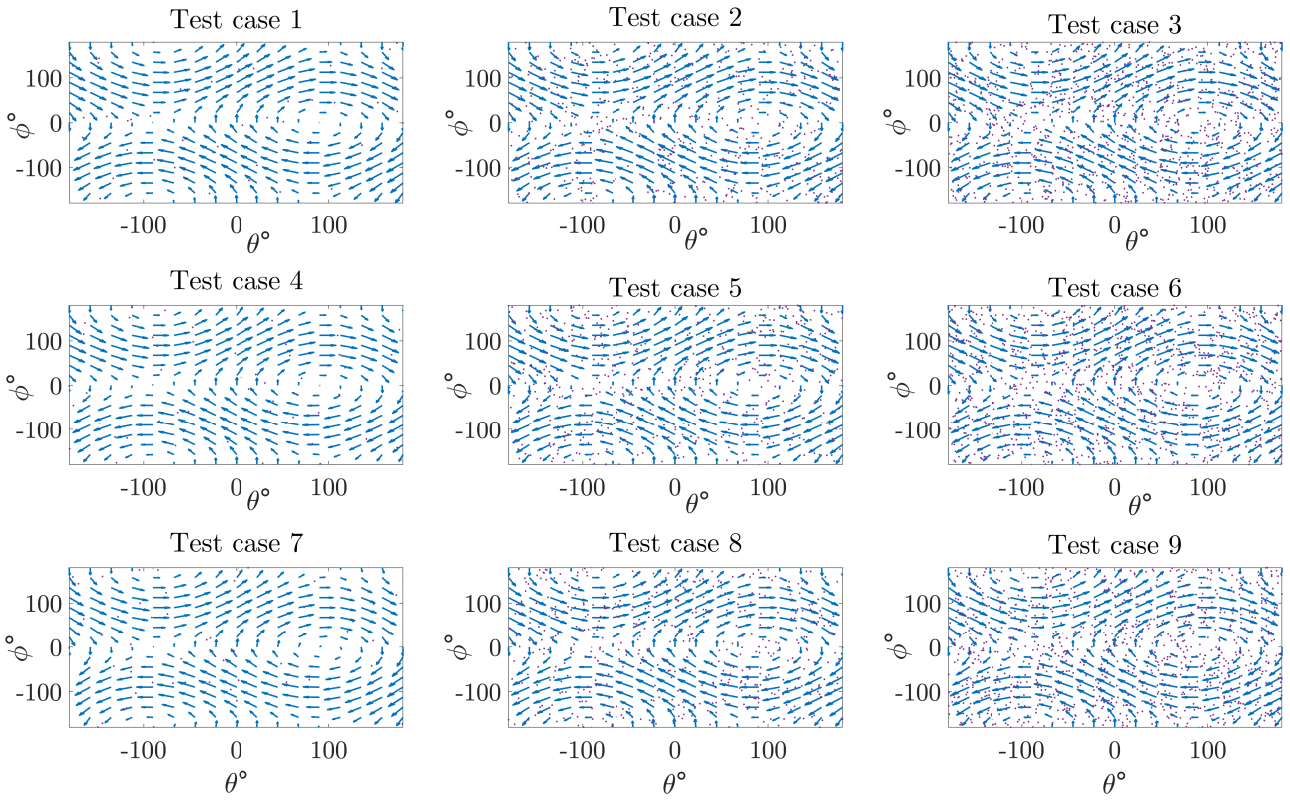


FIGURE 7.8: RECONSTRUCTION OF VECTOR FIELD TEST MODEL

TABLE 7.3: RECONSTRUCTION PARAMETERS OF VECTOR FIELD TEST MODEL

Test case	n	ζ_V	ζ_θ	ζ_ϕ	Discrepancy factor
1	50				0.4
2	500	0.1	0.1	0.1	0.04
3	1000				0.03
4	50				0.8
5	500	0.25	0.25	0.25	0.01
6	1000				0.06
7	50				1
8	500	0.5	0.5	0.5	0.2
9	1000				0.1

The discrepancy factor trends in the vector field test model reconstruction indicate the same trends for the scalar field test model, of improved performance as n increases at a given level of measurement and location error, and the ability of many less accurate measurements to trade-off with fewer accurate ones. In these cases, the reconstruction performance quantified through the discrepancy factors aligns with what can be predicted, showing the worst reconstruction in test cases 7, and the best reconstructions of the same order of magnitude in test cases 2, 3, 5 and 6. Test case 9 has four times better reconstruction accuracy than test case 1, highlighting

the effect of this approach, where in this random case despite considerably higher sensor and location inaccuracy, the vastly larger number of sensors has performed better.

From the result here and in Section 7.2.2, it is clear that this approach to sensing could be useful for scalar and vector field phenomena in orbit. Exact performance characteristics and use case comparisons for specific applications is left as an open area of study for future work. However, the methodology developed demonstrates the compensation that this approach makes over smaller numbers of more accurate sensing measurements when trying to characterise phenomena over large regions of interest. Increased number of sensors compensate for the discrete nature of distributed sensing and works to reduce sensitivity to errors in measurements.

7.2.4 Sampling the magnetic field in LEO

As an example use case, the femto-spacecraft cloud is considered to sample the magnetic field in LEO across a length-scale of several hundred kilometres, with on-board magnetometers coarsely measuring the field magnitude and direction. In this specific scenario it is considered that the sensor cloud is distributed across a region of South America, from the equator to 30° south in latitude (ϕ), and between 60-90° west in longitude (θ), at an orbit altitude of 400 km. Note that this example is not for the purposes of proposing magnetic field reconstruction with this approach, but rather to indicate that real-time measurements in orbit may be used to identify live space weather events. This is presented as a top-level demonstration of how this methodology may work in LEO.

Using the World Magnetic Model (WMM) [129] as a truth reference model, discrete, noisy sensor data is simulated, spread randomly across this region. The same procedures outlined earlier in this chapter are then used to reconstruct the Earth's magnetic field across the area that the swarm is dispersed over. Figure 7.9 shows the reconstruction of this region simulated by using 5000 femto-spacecraft with field intensity measurement error noise level of $\zeta = 0.25$ and $\zeta_\theta = \zeta_\phi = 0.25$. For clarity, discrete sensor points are not shown. This illustrates the WMM field intensity in nano-Tesla (nT) and field directions, with GPR model reconstructions. This example shows in general terms how with a large network of distributed sensors, phenomena of interest, such as the magnetic field as shown here, could be coarsely sampled to characterise behaviours over a large region of space.

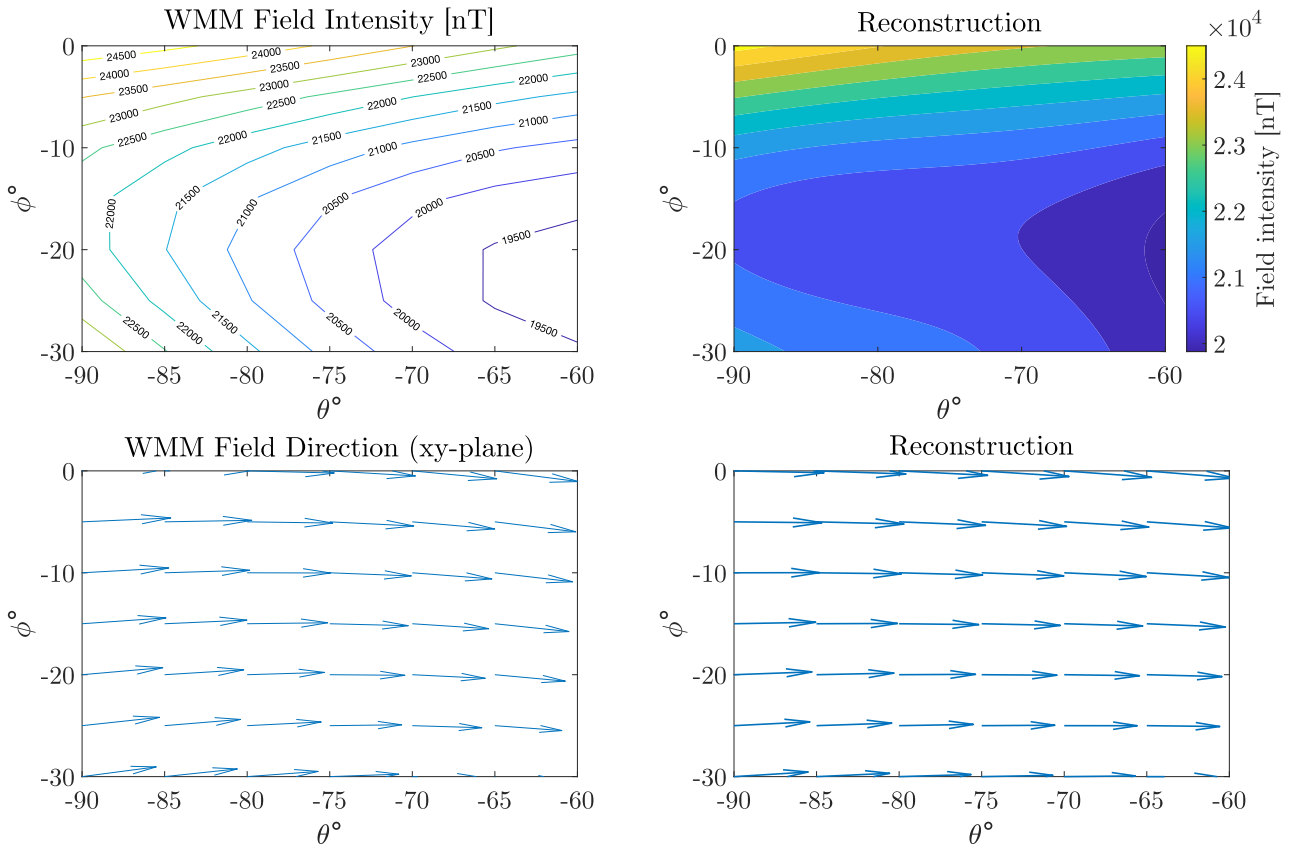


FIGURE 7.9: RECONSTRUCTION OF MAGNETIC FIELD REGION IN LEO

7.3 CHAPTER SUMMARY

This chapter presented a novel distributed sensing strategy using femto-spacecraft swarms as vast sensing clouds in orbit. The proposed concept offers a low-cost alternative to traditional sensing methods in the near-Earth space environment by using many lower quality sensors. The trade-off between number of sensors, sensor accuracy and positioning accuracy has been examined for a series of illustrative test cases for scalar and vector fields.

A methodology has been developed to reconstruct sensed phenomena across a region of interest using sparse, discrete, and noisy sensor data, along with an associated lower accuracy of position localisation. It is clear from the results that large numbers of lower quality measurement can perform well as a direct trade-off for fewer more accurate measurements. This architecture also offers the potential for higher resolution data than fewer measurement points, collected over a range of spatial length-scales. When accumulated over time, this could potentially deconvolve spatial and temporal dynamics and ambiguities if tested operationally for dynamic fields.

Further applications of this work include operational use cases such as data services for in-situ monitoring of space weather and improving the understanding of the structure of the upper atmosphere. This sensing strategy could also provide data at new length-scales to improve the understanding of the near-Earth space environment, both for space science, and for operational applications.

Chapter 8

Conclusions and Future Work

In this thesis, novel strategies have been developed and experimentally demonstrated for range-based relative navigation of a swarm of centimetre-scale femto-spacecraft. This has focused specifically on using only coarse range estimates available from network communications via RSSI data shared between interconnected swarm members, but many of the approaches presented are not limited to the use of one particular range metric. The principal advantage of this approach is that relative navigation is possible through normal swarm network operations that would be required to function in orbit. Relative positioning algorithms using range constraints have been detailed for centralised, decentralised and distributed computation. The range-based relative navigation approaches described in this work could be implemented within femto-spacecraft swarms to enable novel applications in space with a unique design methodology that could provide enhanced satellite applications in the near future.

8.1 Conclusions

The relative positioning algorithms detailed in Chapter 3 presented a variety of range-based techniques within highly interconnected networks using centralised, decentralised or distributed computation. This was presented as a relative positioning problem where only inter-spacecraft range estimates, along with any anchor positions known in advance, are available. For this implementation, it is assumed that range estimates are available from RSSI data via network communications, but the algorithms developed would work in general with any range metric.

The semi-definite programming (SDP) algorithm using convex optimisation techniques has been tested to be robust to a high degree of error in the inter-spacecraft ranges. The principal advantage of the SDP algorithm is its centralised implementation, providing optimisation for every femto-spacecraft given all available range estimates as constraints. This approach makes full use of the increased number of communication links in a centralised network, probabilistically increasing the usefulness of individual range estimates. Simulation-based results in centralised test cases with the SDP algorithm demonstrate an ability to operate with a noise level ζ in the range estimates of up to 40% with only 10% of the swarm as anchors.

Results for the SDP algorithm also demonstrate improved performance with anchor placement towards the edge of the swarm across all simulated noise levels. This indicates that swarm deployment techniques to enable anchors (such as GPS-equipped femto-spacecraft) to congregate towards the edges of the swarm may in principle achieve improved positioning over random dispersal. Decentralised test cases with the SDP algorithm demonstrated a straightforward trade-off bewteen computation time and positioning accuracy with swarm size. In larger networks, decentralised computation may be a desirable extension to ensure real-time relative navigation operationally. However, the polynomial time complexity would ultimately limit the practical implementation of this approach with larger networks, necessitating distributed approaches.

The non-linear least squares (NLLS) algorithm presented a method for distributed computation in larger swarms, using strict trilateration-based conditions to ensure robust positioning with noisy range estimates. In its intended implementation of combining positioning data between clusters of femto-spacecraft distributively, relative positions would be transformed into specific reference frames using a singular value decomposition frame transformation technique. Simulation based results for the NLLS algorithm demonstrated full robust positioning up to a ζ of 20%, above which expected fewer localisations per algorithm sample can be expected for a given network size.

Finally, the normal probability density function (NPDF) algorithm utilises the uncertainty in the range estimates directly, treating estimates as annuli in 2D or spherical shells in 3D based on the range-model confidence in the estimates. For large networks, a region of confidence emerges by summing NPDF functions over the possible solution space in a technique that helps to rule out outlier measurements. This was implemented for experimental testing with small numbers of development kits along with the SDP algorithm as discussed below.

Relative navigation techniques were presented in Chapter 4 from the scenario of a swarm of femto-spacecraft being deployed in LEO from a single larger carrier spacecraft, such as a CubeSat. Two main ejection strategies are presented for a swarm where members have no translational control and simply passively drift in orbits neighbouring the carrier. Firstly, a controlled ejection strategy may be more useful in being able to bound uncertainties on femto-spacecraft positions and for sensing applications offer a reliable method of ensuring evenly distributed sensors. More stochastically, a simultaneous ejection of the entire swarm in many directions around the carrier could be used to detect local phenomena quickly, with the swarm tending to align either in front or behind of the carrier in the along-track direction periodically.

The Kalman filter techniques presented using relative positioning algorithm (simulated using the SDP algorithm) outputs continuously as partial state observers demonstrated in simulation the ability with time to improve real-time state estimation over discrete time relative positioning algorithm outputs alone. Similarly, using the algorithms for initial relative orbit determination (IROD) or for more occasional synchronisation procedures, may offer a less computationally

intensive approach when using an extended Kalman filter that implements range estimates directly. In both cases, and for both ejection strategies considered, relative navigation performance is improved over discrete relative positioning algorithm outputs alone. The methods presented would provide a low-cost relative positioning system for a swarm of femto-spacecraft that, while of limited computational resource individually, can operate collectively for many space-based applications. The relative navigation demonstrated using the centralised algorithm and Kalman filtering in different swarm dispersal scenarios highlights opportunities for their practical implementation in low Earth orbit.

Without practical implementation and testing, it can only be predicted that the methods in chapters 3 and 4 are practically viable, especially for using network RSSI data as a range metric. A key contribution of this thesis is experimental demonstration, which required an implementation of swarm networking for testing. As presented in Chapter 5, a fully interconnected network protocol was developed using small development kits as femto-spacecraft proxies. The development kits and sensor packs used were representative of the available processing and sensing capabilities at the femto-spacecraft scale. The MCU used is also currently implemented within a custom PCB ChipSat design in development within the Space and Exploration Technology Group at the University of Glasgow. This means that the networking and sensing developments in this work can be easily utilised for future work.

The embedded software development for network implementation depended upon modifications to existing two-way communications support for the TI CC1310 MCU. The contribution of this work was to implement an address filtering technique along with a common radio packet payload structure for all communications to enable full interconnectivity. The specific parameters for implementation were designed to focus on accumulating network RSSI data for experimental demonstration of relative navigation, but the application of this functionality extends to more general swarm operations such as distributed sensing. Modifications to the existing protocol to support sensor and IMU data integration were tested and demonstrated to work reliably.

Chapter 6 built upon the outcomes of chapters 3-5 by presenting the results of a testing programme for experimental demonstration of relative navigation of a swarm of femto-spacecraft. As in previous chapters, the specific focus was the use of RSSI data as a range metric, but the procedures are applicable to other methods, such as distributed sensing testing or other range measurement techniques, and potentially to other terrestrial applications such as in GPS-denied environments or indoors. Conclusions can be drawn with consideration that testing was static and limited to 2D over a 120 m $\times$ 60 m sports pitch. The path loss model (PLM) experimentally derived to convert RSSI data into an approximation for a range measurement highlighted the considerable inaccuracies in raw RSSI values. Since the resolution of RSSI was limited to 1 dBm, this becomes a larger factor as range increases.

In general, testing indicated that RSSI is not a reliable metric for simply determining the range between two communicating devices, in this test process using development kits on a sports

pitch. This inaccuracy can be well approximated by range annuli, as shown by the NPDF algorithm implementation, but even then this does not have a high reliability. The utility of RSSI data is only apparent when combined with the PLM and all development kits' many communication links. The quadratic increase in the number of communications links as the number of development kits increases linearly provides many constraints for the algorithms presented in Chapter 3. Probabilistically, this makes errors in individual RSSI values significantly less capable of introducing positioning inaccuracy as the network size increases. It is only then that the relative positioning algorithms' ability to use all these constraints to provide position estimates becomes clear.

Experimentally, the PLM was used to convert RSSI values into range estimate inputs for both the SDP and NPDF algorithms, both for relative positioning and relative navigation demonstrations. The results demonstrated the viability of this method, through fusing coarse constraints to provide a relative positioning system for a swarm of femto-spacecraft entirely via networking.

Relative navigation would be invaluable to swarm applications in providing an estimate of where sensors are when measurements are recorded, as was discussed in Chapter 7. The distributed in-situ sensing analysis in simulation highlights the potential for a network to compensate for sensing and positioning inaccuracies through the high numbers of femto-spacecraft utilised.

In summary, this thesis demonstrated the viability of range-based relative navigation methods for femto-spacecraft swarms, from the development of relative positioning algorithms, implementation of navigation filtering approaches, development of a network communications protocol for femto-spacecraft hardware, experimental demonstration of the methods presented, and investigation of distributed sensing approaches. The experimental results obtained for relative positioning demonstrate the feasibility of the methods presented and offer key insights into the main challenges and limitations of this approach in practice, as a starting point in future work towards in-orbit implementation.

8.2 Future Work

This thesis has focused on the development and experimental validation of relative navigation strategies for a swarm of femto-spacecraft. To facilitate this, necessary simplifying assumptions were made in simulation and analysis that carry over into the experimental work. In this section, the limitations of the thesis are discussed with a series of ideas and recommendations for future work.

The relative positioning algorithms developed and tested in Chapter 3 consider highly and fully interconnected networks of femto-spacecraft for real-time centralised computation, or many clusters for decentralised or distributed computation. In this thesis, centralised computation was assumed to be the role of a larger carrier spacecraft that deploys the swarm. Decentralised and distributed computation has not been tested experimentally in this work. From

the simulation-based analysis of algorithm time complexity in Chapter 3, further investigation into the realistic implementation considerations for centralised approaches is required. This would be predominantly in terms of the available computing power for a centralised spacecraft, but also the implications of this for the number of femto-spacecraft in a network before a centralised approach becomes infeasible.

The extent to which individual femto-spacecraft could contribute to the operations in terms of computational loads is an open question for further research. For the distributed NLLS algorithm, it is assumed that individual femto-spacecraft are able to perform low-level data accumulation and mathematical operations. Future work may consider the role of a femto-spacecraft in overall swarm operations beyond simple networking and individualised sensing roles to make full use of the operational benefits of swarming systems, in terms of self-organisation and sensor fusion capabilities. As part of the consideration of the role of individual femto-spacecraft, relative navigation could utilise distributed Kalman filtering and consensus filter methods as have previously been proposed in the literature for distributed systems [4, 130].

The analysis of relative navigation techniques considered two primary ejection methods, but did not study the potential solution space or swarm evolution beyond several orbits. Further analysis could examine the scope of potential ejection velocities, both in terms of variations in magnitude and direction. This would give further insight into swarm evolution and the volume of space that a swarm would be able to operate in with these methods. The instantaneous swarm ejection strategy did not consider the practical limitations of the deployer's geometry or attitude, which would introduce restrictions for the ejection directions. Consideration of these limitations would provide a more realistic understanding of the swarm structure for a known deployer, such as a CubeSat.

As with the algorithm computation and filtering implementation, the network protocol developed has been limited by testing with a relatively small number of development kits with respect to the potential number of femto-spacecraft intended in swarm use cases. Distributed network protocol development to support distributed sensing applications would be a useful extension to the existing implementation. This could facilitate extended swarm operations when the swarm spatial structure extends outwith the maximum communication range of individual members. Similarly, other additions to the current protocol would add robustness for use in orbit. It was assumed in this work that communication links are undirected for the purposes of quickly accumulating RSSI data, but directed communications could be used as a simple error checking procedure. With the use of an IMU for implementing a femto-spacecraft attitude determination and control system, additions to the software to support filtering techniques would be necessary. For relative navigation, if femto-spacecraft share their relative attitude in addition to RSSI data when transmitting throughout the network, this information could in principle be used to compensate for directive antenna losses that can be expected. In most scenarios, a scaling factor accounting for the expected degree of directive loss could be applied to the path loss modelling. RSSI data in this work was obtained using TI EasyLink with a radio packet

protocol. Other wireless technologies and protocols should be considered for internet-of-things applications, such as LoraWAN, presenting operational trade-offs and for other range-based approaches such time and angle of arrival methods that may become useful at this length-scale in the future.

In experimental demonstration, the main limitations are that testing is static, 2D, restricted to a 120 m $\times$ 60 m area and to a relatively small number of femto-spacecraft proxies. As discussed in Chapter 6, the effect of dynamic and 3D femto-spacecraft swarms may require consideration of additional factors, such as the effect of antenna orientation and anisotropy in 3D path loss modelling for future work and testing. Increasing the number of development kits used may also offer further insight into the in-orbit implications and accuracy of this technology. The effect of antenna orientation and gain on RSSI within the development of a path loss model and relative positioning results is neglected in this thesis. This decision was made on the basis of the testing being limited to 2D, where with careful placement of the antennas the radiation is in 2D generally undirected. Also, as the testing was static, development kits could be placed with known orientations within the testing environment. This meant that directionality effects could be considered, where they were found to be within path loss model confidence bounds.

In further testing or implementation in space, femto-spacecraft would be moving dynamically in 3D space. As anisotropy in radiation patterns is more significant in 3D, a consideration for future work would be whether this effect should be accounted for when modelling the relationship between RSSI and range, or to treat the antenna as undirected and ignore this effect. If the antenna is not rigid, and therefore cannot be straightforwardly associated with the spacecraft body with respect to its orientation, the latter option would be expected to be the most suitable. The spacecraft body itself can be expected to cause some degree of directive loss and interference, which should be considered with antenna modelling and characterisation procedures. Future work could investigate the effect of spacecraft attitude in 3D experimentally to determine whether improvements can be made to the range model and positioning accuracy.

The methods outlined in Chapter 6 for developing a PLM focus on terrestrial interference factors, but for use in space, modelling that takes account of expected loss behaviour in the space environment would be necessary. Even with this model, as has been found with extensive testing in this thesis, it is likely that varying environmental factors would have an impact. This means that pre-implementation, the model is not optimised for a given testing environment. One way this could be compensated for in LEO is by using a proportion of the swarm anchors with GPS. If anchors share RSSI data between other anchors before a localisation sequence starts, the model could be adaptively tuned based on the more accurate knowledge of ranges between these spacecraft. Such an adaptive model would provide data for further use in the future. This could be simulated by modelling the radio environment in the vacuum of space to provide a test-bed for further testing and analysis of different swarm applications. Adaptive path loss modelling, and extended simulations of how this could fluctuate in orbit, is a key point of analysis for further study and practical implementation of this work.

In summary, for experimental demonstration, extended 3D and dynamic testing with larger networks would offer greater insight into in-orbit performance than was possible with the testing outlined in this thesis, and is a suggested starting point for future work. The consideration of greater length-scales and antenna orientation variations that would be possible in 3D testing would provide answers to these open implementation questions. Larger networks may also provide insight into the upper limits of positioning accuracy with RSSI-based relative positioning. It can be supposed from the simulation and experimental results that a larger number of femto-spacecraft would improve positioning accuracy.

Finally, the in-situ sensing analysis in Chapter 7 does not consider time-varying parameters necessary for deconvolution of temporal ambiguities when sensing phenomena distributively. Further applications of this approach include operational use cases such as data services for in-situ monitoring of space weather and improving the understanding of the structure of the upper atmosphere. This sensing strategy could also provide data at new length-scales to improve the understanding of the near-Earth space environment, both for space science and operational applications. Using the sensor and IMU support for the network protocol developed in Chapter 5, this sensing strategy could be tested experimentally, along with 3D dynamic testing of relative navigation techniques.

References

[1] "CubeSat Technology Past and Present: Current State-of-the-Art Survey." `https://ntrs.nasa.gov/api/citations/20210000201/downloads/TP-20210000201.pdf`. NASA Technical Report, Online, last accessed May 2023.

[2] T. R. Perez and K. Subbarao, "A Survey of Current Femtosatellite Designs, Technologies, and Mission Concepts," *Journal of Small Satellites*, vol. 5, no. 3, pp. 467–482, 2016.

[3] S. Janson and D. Barnhart, "The Next Little Thing: Femtosatellites," *The Small Satellite Conference*, Technical Session VI, Logan, Utah, 2013.

[4] F. Y. Hadaegh, S. J. Chung, and H. M. Manohara, "On Development of 100-gram-class spacecraft for swarm applications," *IEEE Systems Journal*, vol. 10, no. 2, pp. 673–684, 2016.

[5] T. Timmons, G. Bailet, J. Beeley, and C. McInnes, "Mars Atmospheric Characterization with a ChipSat Swarm," *Journal of Spacecraft and Rockets*, vol. 58, pp. 1453–1460, 2021.

[6] T. Timmons, G. Bailet, J. Beeley, and C. R. Mcinnes, "Solar System Swarm Probes: An Earth-based Technology Demonstrator," in *IPPW 2021*.

[7] T. Timmons, J. Beeley, G. Bailet, and C. R. Mcinnes, "Range-Based Relative Navigation for a Swarm of Centimeter-Scale Femto-spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 45, no. 9, pp. 1583–1597, 2022.

[8] T. Timmons, J. Beeley, G. Bailet, and C. R. Mcinnes, "Experimental Testing of Range-based Relative Positioning Strategies for a Swarm of Centimetre-scale Femto-spacecraft," *73rd International Astronautical Congress*, IAC-21-B4.IP.68691, Paris, France, 2022.

[9] T. Timmons, J. Beeley, G. Bailet, and C. R. Mcinnes, "Massively Parallel In-Situ Sensing using Femto-Spacecraft Clouds," *72nd International Astronautical Congress*, IAC-21-B4.7.4.64209, Dubai, UAE, 2021.

[10] A. Filella, F. Nadal, C. Sire, E. Kanso, and C. Eloy, "Model of Collective Fish Behavior with Hydrodynamic Interactions," *Physical Review Letters*, vol. 120, no. 19, pp. 198–204, 2018.

[11] W. Bialek, A. Cavagna, I. Giardina, T. Mora, E. Silvestri, M. Viale, and A. M. Walczak, "Statistical mechanics for natural flocks of birds," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 109, no. 13, pp. 4786–4791, 2012.

[12] C. J. Verhoeven, M. J. Bentum, G. L. Monna, J. Rotteveel, and J. Guo, "On the origin of satellite swarms," *Acta Astronautica*, vol. 68, no. 7-8, pp. 1392–1395, 2011.

[13] M. D'Errico, ed., *Distributed Space Missions For Earth System Monitoring*. El Segundo, California: Microcosm Press, Springer, 1st ed., 2013.

[14] F. R. Hainsworth, "Precision and Dynamics of Positioning by Canada Geese Flying in Formation," *Journal of Experimental Biology*, vol. 128, no. 1, pp. 445–462, 1987.

[15] Z. R. Manchester and M. A. Peck, "Stochastic space exploration with microscale spacecraft," *AIAA Guidance, Navigation, and Control Conference*, Portland, Oregon, 2011.

[16] T. Alfriend, *Spacecraft Formation Flying: Dynamics, Control and Navigation*. Oxford: Elsevier, 1st ed., 2010.

[17] J. A. Atchison and M. Peck, "A Millimeter-Scale Lorentz-Propelled Spacecraft," *AIAA Guidance, Navigation, and Control Conference*, Hilton Head, South Carolina, 2007.

[18] Z. Manchester, M. Peck, and A. Filo, "KickSat: A Crowd-Funded Mission To Demonstrate The World's Smallest Spacecraft," *Annual AIAA/USU Conference on Small Satellites*, SSC13–IX–5, Logan, Utah, 2013.

[19] D. Barnhart, T. Vladimirova, M. Sweeting, R. Balthazor, L. Enloe, H. Krause, T. Lawrence, M. Mcharg, J. Lyke, J. White, and A. Baker, "Enabling Space Sensor Networks with PCBSat," *21st Annual AIAA/USU Conference on Small Satellites*, Logan, Utah, 2007.

[20] P. Fortescue, G. Swinerd, and J. Stark, *Spacecraft Systems Engineering*. Wiley, 4th ed., 2011.

[21] "NASA, CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers." `https://www.nasa.gov/sites/default/files/atoms/files/nasa_csli_cubesat_101_508.pdf`. Online, last accessed May 2023.

[22] "Nanosats Database." `https://www.nanosats.eu`. Online, last accessed May 2023.

[23] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, and R. Leonardi, "Towards the thousandth CubeSat: A statistical overview," *International Journal of Aerospace Engineering*, 2019.

[24] R. A. Deepak and R. J. Twiggs, "Thinking Outside the Box : Space Science Beyond the CubeSat," *Journal of Small Satellites*, vol. 1, no. 1, pp. 3–7, 2011.

[25] F. Davoli, C. Kourogiorgas, M. Marchese, A. Panagopoulos, and F. Patrone, "Small satellites and CubeSats: Survey of structures, architectures, and protocols," *International Journal of Satellite Communications and Networking*, vol. 37, no. 4, pp. 343–359, 2019.

[26] J. Schoolcraft, A. Klesh, and T. Werne, "MarCO: Interplanetary Mission Development on a CubeSat Scale," *AIAA SpaceOps 2016 Conference*, Daejeon, Korea, 2016.

[27] Y. Prevereaud, F. Sourgen, D. Mimoun, A. Gaboriaud, J.-L. Verant, and J.-M. Moschetta, "Predicting the Atmospheric Re-Entry of Space Debris Through the Qb50 Entrysat Mission," *6th European Conference on Space Debris*, Darmstadt, Germany, 2013.

[28] S. Caldwell, "NASA Nodes: Network Operation Demonstration Satellites." `https://www.nasa.gov/centers/ames/engineering/projects/nodes.html`. Online, last accessed May 2023.

[29] A. Budianu, R. T. Rajan, S. Engelen, A. Meijerink, C. J. Verhoeven, and M. J. Bentum, "OLFAR: Adaptive topology for satellite swarms," *62nd International Astronautical Congress*, Cape Town, South Africa, 2011.

[30] J. Meurisse, "The potential of CubeSats for space weather applications," *ISAE Supaeroo White Paper*, October, 2014.

[31] J. Atchison, "Microscale Atmospheric Reentry Sensors," *7th International Planetary Probe Workshop*, Barcelona, 2010.

[32] C. R. McInnes, "A continuum model for the orbit evolution of self-propelled 'smart dust' swarms," *Celestial Mechanics and Dynamical Astronomy*, vol. 126, no. 4, pp. 501–517, 2016.

[33] J. Barker and F. Rodriguez-Salazar, "Self-organizing smart dust sensors for planetary exploration," *Workshop on Nanosensors: Self-Organization and Swarm Robotics*, Boston, Massachusetts, 2008.

[34] L. Izquierdo and J. Tristancho, "Next generation of sensors for femto-satellites based on commercial-of-the-shelf," *AIAA/IEEE Digital Avionics Systems Conference*, Seattle, Washington, 2011.

[35] A. M. Hein, Z. Burkhardt, and T. M. Eubanks, "AttoSats: ChipSats, other Gram-Scale Spacecraft, and Beyond," *70th International Astronautical Congress*, Washington D.C., 2019.

[36] N. Tahri, C. Hamrouni, A. M. Alimi, W. K. H. Whfkqrorjlfdo, S. Lq, W. Ri, G. Ri, F. Hduwk, F. Dqg, R. Plolwdu, J. Dqg, V. R. Rq, F. Wklv, V. S. Yhklfoh, J. Vshflilf, F. Dqg, H. Fhoo, V. V. Kdyh, E. Ghyhorshg, D. Vhuylfh, and F. Txdolw, "Study of Current Femto-Satellite Approches and Services," 2013.

[37] Z. Hu, T. Timmons, L. Stamat, and C. McInnes, "Development of a 10g Femto-satellite with Active Attitude Control," *Proceedings of the 17th Reinventing Space Conference*, Belfast, Northern Ireland, 2019.

[38] J. A. Atchison and M. A. Peck, "A passive, sun-pointing, millimeter-scale solar sail," *Acta Astronautica*, vol. 67, no. 1-2, pp. 108–121, 2010.

[39] J. A. Atchison and M. A. Peck, "Length scaling in spacecraft dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 1, pp. 231–246, 2011.

[40] "NASA: KickSat-2." `https://www.nasa.gov/ames/kicksa`. Online, last accessed May 2023.

[41] V. H. Adams and M. Peck, "R-Selected Spacecraft," *Journal of Spacecraft and Rockets*, vol. 57, no. 1, 2019.

[42] "NASA Ames Research Center: Kicksat." `https://www.nasa.gov/ames/kicksat`. Online, last accessed May 2023.

[43] D. J. Bamhart, T. Vladimirova, and M. N. Sweeting, "Satellite miniaturization techniques for space sensor networks," *Journal of Spacecraft and Rockets*, vol. 46, no. 2, pp. 469–472, 2009.

[44] B. Stuurman and K. Kumar, "RyeFemSat: Ryerson University Femtosatellite Design and Testing," in *SpaceOps 2010 Conference Delivering on the Dream Hosted by NASA Marshall Space Flight Center*.

[45] "Pocket-PUCP Picosatellite." `https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=POCKETPU`. Online, last accessed May 2023.

[46] L. Yang, J. Guo, C. Fan, X. Song, S. Wu, and Y. Zhao, "The design and experiment of stardust femto-satellite," *Acta Astronautica*, vol. 174, pp. 72–81, 2020.

[47] E. Fernandez-Murcia, L. Izquierdo, and J. Tristancho, "A synthetic aperture antenna for femto-satellites based on commercial-of-the-shelf," *AIAA/IEEE Digital Avionics Systems Conference*, Seattle, Washington, 2011.

[48] J. Tristancho and J. Gutierrez-Cabello, "A probe of concept for femto-satellites based on commercial-of-the-shelf," AIAA/IEEE Digital Avionics Systems Conference, Seattle, Washington, 2011.

[49] T. Brashears, "SpinorSat." `https://medium.com/@trbrashears_90628/spinorsat-a-step-along-a-50-year-journey-to-the-stars-23ee4effe6ab`. Online, last accessed May 2023.

[50] D. J. Barnhart, T. Vladimirova, and M. Sweeting, "Satellite-on-a-chip Feasibility for Distributed Space Missions," *Proceedings of CANEUS*, Toulouse, France, 2006.

[51] R. Balthazor, M. HcHarg, C. L. Enloe, A. Wallerstein, K. Wilson, B. Rinaldi, R. Raynor, L. Scherliess, R. Schunk, R. Brown, and D. Barnhart, "Sensitivity of Ionospheric Specifications to In Situ Plasma Density Observations Obtained From Electrostatic Analyzers Onboard of a Constellation of Small Satellites," *AIAA/USU Small Satellite Conference*, Session IV, Logan, Utah, 2012.

[52] N. M. Horri, K. U. Kristiansen, P. Palmer, and M. Roberts, "Relative attitude dynamics and control for a satellite inspection mission," *Acta Astronautica*, vol. 71, pp. 109–118, 2012.

[53] T. Vladimirova, X. Wu, and C. P. Bridges, "Development of a satellite sensor network for future space missions," in *IEEE Aerospace Conference Proceedings*, (Big Sky, Montana), 2008.

[54] J. Cao, C. Clemente, C. R. McInnes, and J. J. Soraghan, "Solar Radiation Pressure Enabled Femtosatellite Based Earth Remote Sensing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 5, pp. 3450–3461, 2020.

[55] B. Streetman, "Exploration Architecture with Quantum Inertial Gravimetry and In-situ ChipSat Sensors," *NIAC Phase I Final Report*, Draper Laboratory, 2015.

[56] J. S. Umansky-Castro, K. G. Yap, and M. A. Peck, "ChipSats for Planetary Exploration: Dynamics and Aerothermal Modeling of Atmospheric Entry and Dispersion," *Frontiers in Astronomy and Space Sciences*, vol. 8, no. 7, pp. 1–17, 2021.

[57] A. M. Hein, K. F. Long, D. Fries, N. Perakis, A. Genovese, S. Zeidler, M. Langer, R. Osborne, R. Swinney, J. Davies, B. Cress, M. Casson, A. Mann, and R. Armstrong, "The Andromeda Study: A Femto-Spacecraft Mission to Alpha Centauri," 2017.

[58] L. Niccolai, M. Bassetto, A. A. Quarta, and G. Mengali, "A review of smart dust architecture, dynamics, and mission applications," *Progress in Aerospace Sciences*, vol. 106, pp. 1–14, 2019.

[59] A. Savvides, C. C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, Rome, Italy, 2001.

[60] V. Phillips, "NASA Concepts: Swarm/SODA." `https://www.nasa.gov/feature/swarmsoda`. Online, last accessed May 2023.

[61] M. Marszalek, O. Kurz, M. Drentschew, M. Schmidt, and K. Schilling, "Intersatellite links and relative navigation: Pre-conditions for formation flights with pico- and nanosatellites," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 3027–3032, 2011. 18th IFAC World Congress.

[62] B. S. Narayana, R. V. Prasad, V. S. Rao, L. Mottola, and T. V. Prabhakar, "Hummingbird : An Energy-Efficient GPS Receiver for Small Satellites," *Communications of the ACM*, vol. 65, no. 11, 2022.

[63] A. Tadanki and E. G. Lightsey, "Closing the Power Budget Architecture for a 1U CubeSat Framework," pp. 1–20, Technical Report, Georgia Tech, 2020.

[64] R. Magness, "Short-range RF Wireless (Proximity) Networks Technology Assessment for Space Applications," *ESA Tec-E Wireless Technology Dossier*, 2006.

[65] W. Bezouska and D. Barnhart, "Decentralized Cooperative Localization with Relative Pose Estimation for a Spacecraft Swarm," *IEEE Aerospace Conference Proceedings*, Big Sky, Montana, 2019.

[66] M. Sabatini, R. Volpe, and G. B. Palmerini, "Centralized visual based navigation and control of a swarm of satellites for on-orbit servicing," *Acta Astronautica*, vol. 171, pp. 323–334, 2020.

[67] S. D'Amico and J. R. Carpenter, *Satellite Formation Flying and Rendezvous*, ch. 63, pp. 1921–1946. John Wiley Sons, Ltd, 2020.

[68] L. Yang, X. Dong, and D. Xu, "Precise Navigation for GRACE Formation Flying," *International Symposium on Systems and Control in Aerospace and Astronautics (ISSCAA)*, Harbin, China, 2006.

[69] J. A. Christian, "Relative navigation using only intersatellite range measurements," *Journal of Spacecraft and Rockets*, vol. 54, no. 1, pp. 13–28, 2017.

[70] C. Gao, G. Zhao, and H. Fourati, *Cooperative Localization and Navigation: Theory, Research, and Practice*. CRC Press, 1st ed., 2019.

[71] B. Gong, S. Wang, M. Hao, X. Guan, and S. Li, "Range-based collaborative relative navigation for multiple unmanned aerial vehicles using consensus extended Kalman filter," *Aerospace Science and Technology*, vol. 112, p. 106647, 2021.

[72] W. Li, B. Jelfs, A. Kealy, X. Wang, and B. Moran, "Cooperative Localization Using Distance Measurements for Mobile Nodes," *Sensors*, vol. 21, no. 1507, pp. 1–18, 2021.

[73] J. Djugash, S. Singh, G. Kantor, and W. Zhang, "Range-Only SLAM for Robots Operating Cooperatively with Sensor Networks," *International Conference on Robotics and Automation*, Orlando, Florida, 2006.

[74] A. Sonnenburg, M. Tkocz, and K. Janschek, "Ekf-slam based approach for spacecraft rendezvous navigation with unknown target spacecraft," *IFAC Proceedings Volumes*, vol. 43, no. 15, pp. 339–344, 2010. 18th IFAC Symposium on Automatic Control in Aerospace.

[75] G. Mao, B. Fidan, and B. D. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[76] M. Laaraiedh, L. Yu, S. Avrillon, and B. Uguen, "Comparison of hybrid localization schemes using rssi, toa, and tdoa," in *17th European Wireless 2011 - Sustainable Wireless Technologies*, pp. 1–5, 2011.

[77] M. G. Vidal-Valladares and M. A. Díaz, "A femto-satellite localization method based on tdoa and aoa using two cubesats," *Remote Sensing*, vol. 14, no. 5, 2022.

[78] "ESA Navipedia: Reference Systems and Frames." `https://gssc.esa.int/navipedia/index.php/Reference_Systems_and_Frames`. Online, last accessed May 2023.

[79] H. Sanchez, D. Mcintosh, H. Cannon, C. Pires, J. Sullivan, S. D'Amico, and B. O'connor, "Starling1: Swarm Technology Demonstration," *32nd Annual AIAA/USU Conference on Small Satellites*, 2018.

[80] K. Matsuka, A. O. Feldman, E. S. Lupu, S. J. Chung, and F. Y. Hadaegh, "Decentralized formation pose estimation for spacecraft swarms," *Advances in Space Research*, 2020.

[81] J. Aspnes, T. Eren, D. K. Goldenberg, A. S. Morse, W. Whiteley, R. Yang, B. D. Anderson, and P. N. Belhumeur, "A theory of network localization," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1663–1677, 2006.

[82] P. Baran, "On Distributed Communication Systems," *The RAND Corporation*, 1964.

[83] L. Vandenberghe and S. Boyd, "Semidefinite programming," *Society for Industrial and Applied Mathematics 003*, 1996.

[84] "Positive Definite and Semidefinite Forms (Purdue University Lecture Series)." `https://www.math.purdue.edu/~eremenko/dvi/lect4.9.pdf`. Online, last accessed May 2023.

[85] L. Doherty, K. S. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, Anchorage, Alaska, 2001.

[86] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," *Third International Symposium on Information Processing in Sensor Networks (IPSN)*, Berkeley, California, 2004.

[87] A. M. C. So and Y. Ye, "Theory of semidefinite programming for sensor network localization," *Mathematical Programming*, vol. 109, no. 2-3, pp. 367–384, 2007.

[88] R. W. Ouyang, A. K. S. Wong, C. T. Lea, and V. Y. Zhang, "Received signal strength-based wireless localization via semidefinite programming," *GLOBECOM - IEEE Global Telecommunications Conference*, Honolulu, Hawaii, 2009.

[89] P. Biswas and T.-c. Wang, "Semidefinite Programming Based Algorithms for Sensor Network Localization." https://web.stanford.edu/~yyye/combined_rev3.pdf. Online, last accessed May 2023.

[90] P. Biswas, T. C. Liang, K. C. Toh, Y. Ye, and T. C. Wang, "Semidefinite programming approaches for sensor network localization with noisy distance measurements," *IEEE Transactions on Automation Science and Engineering*, vol. 3, no. 4, pp. 360–371, 2006.

[91] S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming." http://cvxr.com/cvx/. Online, last accessed May 2023.

[92] J. Welch, "Discrete Algorithms for Mobile and Wireless Systems," in *CPSC 689: Lecture Series, Texas A&M*, 2009.

[93] D. Moore, "Robust Distributed Network Localization with Noisy Range Measurements," *Proceedings of ACM SenSys*, Baltimore, Maryland, 2004.

[94] R. Fisher, "A Comparison of Four Algorithms for Estimating 3-D Rigid Transformations," *Proceedings of the British Machine Vision Conference*, Birmingham, UK, 1995.

[95] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 9, no. 5, pp. 698–700, 1987.

[96] R. Mautz, W. Ochieng, G. Brodin, and A. Kemp, "3D wireless network localization from inconsistent distance observations," *Ad-Hoc and Sensor Wireless Networks*, vol. 3, no. 2-3, pp. 141–170, 2007.

[97] "Z-Tests and T-Tests." https://www.ncl.ac.uk/webtemplate/ask-assets/external/maths-resources/business/hypothesis-tests/z-tests-and-t-tests.html. Online, last accessed May 2023.

[98] "Six Mathematical Gems from the History of Distance Geometry." https://arxiv.org/pdf/1502.02816.pdf. Online, last accessed May 2023.

[99] T. Karlsson and M. Persson, "Outdoor localization in long range WSN using trilateration," 2017.

[100] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-D rigid body transformations: A comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, no. 5-6, pp. 272–290, 1997.

[101] K. C. Toh, M. Todd, and R. Tutuncu, "SDPT3 - A MATLAB software package for semidefinite programming." https://www.math.cmu.edu/~reha/Pss/guide3.0.pdf. Online, last accessed May 2023.

[102] J. Holly Hui, "Scalable sensor localization algorithms for wireless sensor networks," pp. 1 – 106, PhD Thesis, University of Toronto, 2005.

[103] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[104] W. Clohessy and R. Wiltshire, "Terminal Guidance System for Satellite Rendezvous," *ARS Journal*, vol. 31, no. 11, pp. 1516–1521, 1961.

[105] B. S. Kumar, A. Ng, K. Yoshihara, and A. De Ruiter, "Differential drag as a means of spacecraft formation control," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 2, pp. 1125–1135, 2011.

[106] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME, Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[107] T. Ebinuma, O. Montenbruck, E. G. Lightsey, and E. M. De, "Precise Spacecraft Relative Navigation using Kinematic Inter-Spacecraft State Estimates," *ION GPS Conference*, Portland, Oregon, 2002.

[108] "TI CC1310 Development Kit." `https://www.ti.com/tool/LAUNCHXL-CC1310`. Online, last accessed May 2023.

[109] "TI CC1310 Development Kit Sensors BoosterPack." `https://www.ti.com/tool/BOOSTXL-SENSORS`. Online, last accessed May 2023.

[110] "TI CC1310 Development Kit." `https://www.ti.com/lit/ds/symlink/cc1310.pdf`. Online, last accessed May 2023.

[111] "TI SENSEXL BoosterPack." `https://www.ti.com/lit/ug/slau666b/slau666b.pdf`. Online, last accessed May 2023.

[112] "TI-RTOS." `https://www.ti.com/tool/TI-RTOS-MCU`. Online, last accessed May 2023.

[113] "SimpleLink CC13x0 SDK." `https://www.ti.com/tool/SIMPLELINK-CC13X0-SDK`. Online, last accessed May 2023.

[114] "TI Sensor and Actuator Interface Layer." `https://software-dl.ti.com/simplelink/esd/plugins/sail/1.20.00.02/exports/docs/sail/UsersGuide/sail/introduction.html`. Online, last accessed May 2023.

[115] "TI CC1310 Software Development Kit." `https://www.ti.com/tool/download/SIMPLELINK-CC13X0-SDK`. Online, last accessed May 2023.

[116] "TI rfEasylinkEcho Project." `https://software-dl.ti.com/simplelink/esd/simplelink_cc13x0_sdk/2.20.00.38/exports/docs/proprietary-rf/proprietary-rf-users-guide/proprietary-rf-guide/rfEasyLinkEchoTx_README.html`. Online, last accessed May 2023.

[117] "TI CC112X RSSI and CS Response Time." https://www.ti.com/lit/an/swra413/swra413.pdf. Online, last accessed May 2023.

[118] "TI BMI160 Support Project." https://www.ti.com/tool/BOOSTXL-SENSORS#support-training. Online, last accessed May 2023.

[119] *Radio Communications Unit of the ICPT ARPL: The Radio Laboratory Handbook.* 1st ed.

[120] H. Friis, "A Note on a Simple Transmission Formula," *Proceedings of the IRE and Waves and Electrons*, pp. 254–256, 1946.

[121] "TI RSSI Interpretation and Timing." https://www.ti.com/lit/an/swra114d/swra114d.pdf. Online, last accessed May 2023.

[122] "Antenna Propagation for Dipole Antenna." https://www.electronics-notes.com/articles/antennas-propagation/dipole-antenna/build-ham-inverted-v-dipole-80-40-20-15-10-metres.php. Online, last accessed May 2023.

[123] *Radio Communications Unit of the ICPT ARPL: The Radio Laboratory Handbook.* 1st ed.

[124] J. B. Parham, O. V. Dessel, M. Kromis, P. Teng, A. Zosuls, B. Walsh, and J. Semeter, "Calibrating the Swarm: Networked Small Satellite Magnetometers for Auroral Plasma Science," *Small Satellite Conference*, Workshop 33, Logan, Utah, 2017.

[125] N. Sneeuw, "Satellite clusters for future gravity field missions," *Gravity, Geoid and Space Missions: IAG International Symposium*, Porto, Portugal, 2004.

[126] B. A. Maruca, J. A. A. Rueda, R. Bandyopadhyay, F. B. Bianco, A. Chasapis, R. Chhiber, H. Deweese, W. H. Matthaeus, D. M. Miles, R. A. Qudsi, M. J. Richardson, S. Servidio, M. A. Shay, D. Sundkvist, D. Verscharen, S. K. Vines, J. H. Westlake, and R. T. Wicks, "MagneToRE : Mapping the 3-D Magnetic Structure of the Solar Wind Using a Large Constellation of Nanosatellites," *Frontiers in Astronomy and Space Sciences*, vol. 8, no. 7, pp. 1–16, 2021.

[127] L. B. Clausen, J. B. Baker, J. M. Ruohoniemi, S. E. Milan, and B. J. Anderson, "Dynamics of the region 1 Birkeland current oval derived from the active magnetosphere and planetary electrodynamics response experiment (AMPERE)," *Journal of Geophysical Research: Space Physics*, vol. 117, no. 6, 2012.

[128] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning.* MIT Press, 2005.

[129] "The 2020 World Magnetic Model." https://www.ncei.noaa.gov/products/world-magnetic-model. Online, last accessed May 2023.

[130] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, 2005.

Appendix

A.1 Tetrahedra Volume Partial Derivatives

Given:

$$\mathbf{F} = \begin{bmatrix} \frac{\partial V}{\partial r_{12}} & \frac{\partial V}{\partial r_{13}} & \frac{\partial V}{\partial r_{14}} & \frac{\partial V}{\partial r_{23}} & \frac{\partial V}{\partial r_{24}} & \frac{\partial V}{\partial r_{34}} \end{bmatrix} \tag{8.1}$$

then, using the Cayler-Menger determinants:

$$\frac{\partial V}{\partial r_{12}} = -\frac{1}{36}\frac{r_{12}}{V} D(1,3,4;2,4,3) \tag{8.2}$$

$$\frac{\partial V}{\partial r_{13}} = -\frac{1}{36}\frac{r_{13}}{V} D(1,2,4;2,3,4) \tag{8.3}$$

$$\frac{\partial V}{\partial r_{14}} = -\frac{1}{36}\frac{r_{14}}{V} D(1,2,3;3,2,4) \tag{8.4}$$

$$\frac{\partial V}{\partial r_{23}} = -\frac{1}{36}\frac{r_{23}}{V} D(1,2,4;1,4,3) \tag{8.5}$$

$$\frac{\partial V}{\partial r_{24}} = -\frac{1}{36}\frac{r_{24}}{V} D(1,2,3;1,3,4) \tag{8.6}$$

$$\frac{\partial V}{\partial r_{34}} = -\frac{1}{36}\frac{r_{34}}{V} D(1,2,3;2,1,4) \tag{8.7}$$

where, in general:

$$D(1,2,...,N;1',2',...,N') = -\frac{1}{4}\begin{vmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & r^2_{11'} & r^2_{12'} & \dots & r^2_{1N'} \\ 1 & r^2_{21'} & r^2_{22'} & \dots & r^2_{2N'} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & r^2_{N1'} & r^2_{N2'} & r^2_{NN'} & 0 \end{vmatrix} \tag{8.8}$$

A.2 CALCULATION OF $\tilde{r}_{45}$ AND $\tilde{r}_{45'}$

Given:

$$r_{\tilde{45}} = \sqrt{\Delta h^2 + \Delta q^2} = \sqrt{(h_5 + h_4)^2 + \Delta q^2} \tag{8.9}$$

$$r_{\tilde{45'}} = \sqrt{\Delta h'^2 + \Delta q^2} = \sqrt{(h_5 - h_4)^2 + \Delta q^2} \tag{8.10}$$

where:

$$h_4 = \frac{3V_{(1,2,3,4)}}{A_{(1,2,3)}} \tag{8.11}$$

$$h_5 = \frac{3V_{(1,2,3,5)}}{A_{(1,2,3)}} \tag{8.12}$$

and $A_{(1,2,3)}$ is the triangle formed by enclosing the area of $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$:

$$A_{(1,2,3)} = \sqrt{-\frac{1}{16}\begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & r_{12}^2 & r_{13}^2 \\ 1 & r_{12}^2 & 0 & r_{23}^2 \\ 1 & r_{13}^2 & r_{23}^2 & 0 \end{vmatrix}} \tag{8.13}$$

and Δq, the distance between the projections orthogonally of $\mathbf{x}_4$ and $\mathbf{x}_5$ onto $A_{(1,2,3)}$, is:

$$\Delta q = \sqrt{\frac{8(D_4^2 r_{12}^2 + D_5^2 r_{13}^2 + 2D_4 D_5 D(1,2;1,3))}{(D(1,2;1,3))^2}} \tag{8.14}$$

and where $D_4 = D(1,2,3;1,3,4) - D(1,2,3;1,3,5)$ and $D_5 = D(1,2,3;1,2,4) - D(1,2,3;1,2,5)$. See Eq. (8.8) for the general $D(1,2,...,N;1',2',...,N')$ case.

A.3 Supplementary figures

(a) Development kits during laboratory network testing

(b) Outdoor relative navigation testing

Figure A.1: Development kits in testing