



6G BRAINS Deliverable D6.4

Integrated Simultaneous Localisation and Mapping

Editor:	John Cosmas
Deliverable nature:	Report (R) & Prototype (P)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	31 st December 2023
Actual delivery date:	31 st January 2024
Suggested readers:	Executives in 5G/6G Communication OEM industries
Version:	1.0
Total number of pages:	168
Keywords:	localisation, sub 6GHz Time of Arrival, Location from Landmarks, Simultaneous Localisation and Mapping data fusion, federated learning-based localization to overcome multipath; beacon positioning signal design to unlock strict time synchronisation; secure mutual localization using iterative trilateration; Blockchain-based solution for IIoT to increase scalability while ensuring privacy.

Abstract

This deliverable reports on the different experiments that were performed to improve the reliability and accuracy of 5G/6G localisation techniques. It ranges from: sub 6GHz ToA localisation measurement campaigns; localisation from environment landmarks using LIDAR point clouds that could potentially have been sensed from a 6G sensing system; Simultaneous Localisation and Mapping (SLAM) two-step localization system to improve the performance of the data fusion by data combination and selection; federated learning based localization using visible light signals to establish communication and determine the location of the receiving device to overcome the effects of multipath; a beacon positioning signal design which is used to unlock strict time synchronisation; a secure mutual localization system that uses iterative trilateration to determine location when continuous Line of Sight (LoS)

access between at least three position-calibrated anchor nodes and the user equipment (UE) is not available for determining positions; a Blockchain-based solution for IIoT by utilizing sharding and the Interplanetary File System (IPFS) to efficiently store, process, and retrieve data to thereby increase the scalability of IIoT solutions while ensuring privacy.

[End of abstract]

Disclaimer

This document contains material, which is the copyright of certain 6G BRAINS consortium parties, and may not be reproduced or copied without permission.

In case of Public (PU):

All 6G BRAINS consortium parties have agreed to full publication of this document.

In case of Restricted to Programme (PP):

All 6G BRAINS consortium parties have agreed to make this document available on request to other framework programme participants.

In case of Restricted to Group (RE):

All 6G BRAINS consortium parties have agreed to full publication of this document. However this document is written for being used by <organisation / other project / company etc.> as <a contribution to standardisation / material for consideration in product development etc.>.

In case of Consortium confidential (CO):

The information contained in this document is the proprietary confidential information of the 6G BRAINS consortium and may not be disclosed except in accordance with the consortium agreement.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the 6G BRAINS consortium as a whole, nor a certain part of the 6G BRAINS consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, accepting no liability for loss or damage suffered by any person using this information.

The EC flag in this document is owned by the European Commission and the 5G PPP logo is owned by the 5G PPP initiative. The use of the EC flag and the 5G PPP logo reflects that 6G BRAINS receives funding from the European Commission, integrated in its 5G PPP initiative. Apart from this, the European Commission and the 5G PPP initiative have no responsibility for the content of this document.

The research leading to these results has received funding from the European Union Horizon 2020 Programme under grant agreement number 101017226 – 6G BRAINS – H2020-ICT-2020-2. The content of this document reflects only the author’s view and the Commission is not responsible for any use that may be made of the information it contains.

Impressum

[Full project title]	Bring Reinforcement-learning Into Radio Light Network for Massive Connections
[Short project title]	6G BRAINS
[Number and title of work-package]	WP6: Hybrid Radio-optical Wireless 3D Localization System for Ultra-dense Machine Type Communications
[Number and title of task]	T6.1: High Resolution Radio-Light 3D Simultaneous Localization and Mapping (SLAM) and Fusion for Dynamic Ultra-dense Network T6.2: Sub Decimetre Accuracy UE Location Measurement T6.3: AI-driven Mutual Localisation Sensing in Dynamic Ultra-dense D2D Network T6.4: Blockchain Distributed Mutually Assured Secure Location Ledger for Ad-Hoc Networks T6.5: Dynamic D2D Localisation Based Intelligent VNF and Optimised Content Placement in Edge Caches for Multi-agent RL Resource Allocation
[Document title]	D6.4 Integrated Simultaneous Localisation and Mapping
[Editor: Name, company]	John Cosmas, Brunel University London
[Work-package leader: Name, company]	John Cosmas, Brunel University London
[Estimation of PM spent on the Deliverable]	16

Copyright notice © 2024 Participants in 6G BRAINS project

Executive summary

Localisation of User Equipment to within millimetre accuracy and 99.999% reliability remains a critical requirement for both 5G and future 6G networks since future application scenarios with degree of localization precision can be an essential enabler that opens up a lot of new possibilities, for example, tracking of animals in indoor farming scenarios or autonomous movements of robots as well as transportation systems in large warehouses and production halls. This deliverable reports on the tangible progress of results towards these objectives.

This deliverable summarises how all the different localization technologies could work together to 99.9999% reliability obtain localisation with centimetre accuracy.

It reports on results of a sub 6G measurement campaign to measure distance to centimetre accuracy using time of arrival and the repeatability of measurement results. It provides a detailed description of the experimental equipment that was used. Two experimental configurations were explored (1) Same UE Tx and Ref Tx antennas, (2) Separate UE Tx and Ref Tx antennas. It explores and attempts to quantify the source of errors and unreliability of measurements. Experiments were performed with and without EM absorbers to assess the effects of multipath. A detailed description of the format of the 5G FAPI format of the measurement data was presented and used to write a packet sniffing data extraction Python program to capture every single sample measurement from each measurement section. The time series and histogram of ToA of the measurement data results were plotted and analysed to identify any sources of error and calculate the degree of accuracy.

Furthermore, it reports on results of measuring location using an environmental imaging and sensing system. The motivation for this study is to estimate what localisation accuracy could be expected to be obtained from a 6G Communication and Sensing system. Since a 6G Communication and Sensing system has not yet been delivered a LIDAR and 360 camera was used to emulate the imaging and environment sensing system. The LIDAR system captured a point cloud of its environment and a co-located 360-degree camera capture RGB image. A Neural Network was used to identify landmarks in the environment which consisted of square, rectangular, circular and triangular shapes situated at known locations. The shapes of these landmarks were deliberately chosen to be identical to the type of shapes typically used for street signs. Once four or more of the shapes had been recognized the distance from the LIDAR to the centre point of the landmark was measured, which were using to estimated position of the LIDAR using triangulation.

It also reports on data combination and selection for a Simultaneous Localisation and Mapping (SLAM) system. A two-step localization system to improve the performance of the data fusion method is presented consisting of offline and online phases. A selection process is integrated in the localization system based on classification using a Deep Neural Network (DNN) model, trained offline. The model is applied online by taking the Received Signal Strength (RSS) received from both considered technologies, namely: WiFi 2.4 GHz and Optical Wireless Communications (OWC).

Additionally, it reports on federated learning-based localization using visible light signals to establish communication and determine the location of the receiving device. As visible light signals traverse through the space, they encounter various obstacles, such as walls, furniture, and other objects. These obstacles create multipath effects, wherein the signals reflect, refract, or scatter, giving rise to multiple signal paths to reach the receiver. The multipath phenomenon can lead to signal interference and propagation abnormalities, contributing to

inaccuracies in the localization process. The presence of these challenges lead to high localization errors in terms of accuracy, reliability, and scalability. To overcome these limitations and enhance the efficiency of VLC-based localization in dynamic indoor environments, machine learning techniques are needed for data-based localization to mitigate this problem. By adopting machine learning techniques, the performance of VLC-based localization can be significantly improved.

Furthermore, it reports on a beacon positioning signal design which is used to unlock strict time synchronisation. Most of position accuracy application are in condition of a rich hardware resources. Specifically, a high sampling rate is usually required to sample the input signal to avoid the aliasing effect, which leads to increased implementation complexity and power consumption. To anticipate the increasing of IoT device number in the future 6G network, to solve this issue, an indoor beacon construction and low sampling rate positioning scheme is proposed based on on-off keying (OOK) modulation pulse pairs. Different from the traditional time of arrival (TOA)-based positioning scheme, this positioning scheme does not require stringent time synchronization. Moreover, the proposed scheme achieves high positioning performance with a low sampling rate and low bandwidth. Specifically, inspired by the ease of implementing both communication and illumination services with small modulation bandwidth, the OOK modulation is exploited in the considered VLP systems. Then, the OOK-based pulse pairs are proposed to design the positioning beacon signal without the synchronization requirement.

It reports on a secure mutual localization system that is validated through simulation experiments. Localisation schemes are highly dependent on continuous Line of Sight (LoS) access between at least three position-calibrated anchor nodes and the user equipment (UE) for which positions need to be determined. Iterative trilateration in a multi-device environment is developed which uses neighbouring UEs that act as anchor nodes once they have determined their position beforehand. Since a light-weight secure messaging scheme is used in this multi-device environment where a UE's identity is unknown to neighbouring UEs, then a trust framework is required that excludes malicious UEs from the collective localisation process. An simulation experiment is performed which demonstrates the feasibility of the messaging scheme for different setups of number of UEs, spatial scenarios with varying levels of obstructedness, wireless communication latencies, and different communication ranges.

Finally, it provides a detailed description on how to deploy Blockchain-based solution for IIoT. The proof-of-concept deployment starts with an overview on how to increase the scalability of IIoT solutions while ensuring privacy. The recommended blockchain-based architecture utilizes sharding and the Interplanetary File System (IPFS) to efficiently store, process, and retrieve data. This sustainable design choice is considered to make it suitable for large-scale implementations. It is concluded through a proof-of-concept implementation that the underlying blockchain consensus mechanism is the primary factor limiting the IIoT scalability. The implementation and testing were conducted on a Harmony test network connected to a laboratory-scale IIoT testbed. The proposal evaluated the proposed architecture with widely employed blockchain consensus mechanisms. With a block latency of just 2.13 seconds, the architecture outperforms existing systems and can process up to 4000 transactions. This demonstrates the effectiveness and efficiency of the proposed solution for extensive IIoT deployments through in-depth evaluations and performance comparisons.

List of authors

Company	Author	Contribution
Brunel University	John Cosmas	Section 1
Brunel University	John Cosmas	Section 2
Brunel University	Kareem Ali, John Cosmas	Section 3
RunEL	Zion Hadad, Arik Salomon, Almog Karamani, Israel Koffman	Section 3
Brunel University	Prabhveer Mujral, Ali Mahbas, John Cosmas	Section 4
ISEP	Wafa Njima, Xiaodong Liu, Xun Zhang, Frédéric Amiel	Sections 5, 6, 7
Deutsche Telekom	Matthias Weh	Section 8
Brunel University	Nawar Jawad, John Cosmas	Section 9
Brunel University	John Cosmas	Section 10

Table of Contents

Executive summary	5
List of authors.....	7
Table of Contents	8
List of figures	12
List of tables	16
Abbreviations	17
1 Introduction.....	21
2 Simultaneous Localisation and Mapping Strategy.....	22
3 Sub 6GHz measurement campaign in Diverse Environments.....	23
3.1 Aim of Experiment	23
3.2 Experimental Setup	24
3.2.1 Remote Unit System Setup	24
3.2.2 Initial Experimental Setup	25
3.3 Experimental Results and Analysis	28
3.3.1 Experiment 1: Same UE Tx and Ref Tx antennas	28
3.3.2 Experiment 2: Separate UE Tx and Ref Tx antennas	43
3.4 Detailed Measurement Data Results and Analysis.....	53
3.4.1 Capturing the Measurement Results	53
3.4.2 Capturing the Measurement Results	57
4 Environment Imaging and Sensing.....	67
4.1 Aim of Experiment	67
4.2 Background Theory.....	67
4.2.1 Applications of localisation	67
4.2.2 Localisation Techniques	68
4.2.3 LIDAR technology & localisation using LIDAR	69
4.2.4 Position Estimation Algorithms.....	72
4.2.5 Object Detection Algorithms.....	72
4.3 Experimental Setup	73
4.3.1 Introduction.....	73
4.3.2 System Scope and Requirements.....	73
4.3.3 Setting up the System Environment	76
4.3.4 Creating Landmarks.....	76

4.3.5	Creating a 3D Camera Mount.....	77
4.3.6	Interfacing with the LIDAR using Python	78
4.3.7	Training YOLOv5 for object Detection	81
4.3.8	Using Strictly LIDAR for Pose Estimation.....	87
4.3.9	Processing YOLOv5 on Point Clouds.....	88
4.3.10	Extracting Distances from Point Cloud.....	89
4.3.11	Implementation of Kalman Filters.....	90
4.3.12	Trilateration Estimation using Distances	92
4.3.13	Displaying position on a 2D Map.....	94
4.3.14	Position Tracking with Live LIDAR Streams	96
4.3.15	Utilising Both 360 Camera and LIDAR for Position Estimation	96
4.3.16	Processing YOLOv5 on 360-degree Images.....	97
4.3.17	Fusing Camera Images with Point Clouds	98
4.4	Experimental Results and Analysis	98
4.4.1	Comparing Position Accuracy.....	98
4.5	Comparing against the Requirements.....	100
4.6	Testing Against the KPIs.....	101
4.7	System Limitations.....	102
5	Data combination and data selection for SLAM improvement	103
5.1	Aim of experiment	103
5.2	Hybrid localization system using WiFi (2.4GHz) and OWC.....	104
5.3	Simulation results and analysis	105
6	Federated Learning (FL)-based localization	108
6.1	Aim of experiments	108
6.2	Centralized learning and Federated learning background.....	110
6.3	Use case and description of FL framework	111
6.4	Validation of the proposed framework based on a publicly available database	113
7	Novel Beacon Positioning Signal Design To Unlock the Strict Time Synchronization....	115
7.1	System Model and Positioning Beacon	115
7.2	Pulse Reconstruction and Position Estimation.....	118
7.3	Laboratory Experimental Testbed	122
7.4	Experimental results and analysis	123
8	Secure Mutual Localisation Simulation.....	126
8.1	Aim of Experiment.....	126

- 8.2 Background Theory..... 126
 - 8.2.1 Abstract Scenario 126
 - 8.2.2 Iterative Trilateration 127
 - 8.2.3 Digital Signature Scheme 128
 - 8.2.4 Public Key List..... 128
 - 8.2.5 Messages and Message Format 129
 - 8.2.6 Message Verification..... 131
 - 8.2.7 Events 132
- 8.3 Experimental Simulation Setup 133
 - 8.3.1 General assumptions..... 133
 - 8.3.2 Simulation implementation 134
 - 8.3.3 Statistics collection..... 137
- 8.4 Experimental Simulation Results and Analysis..... 137
 - 8.4.1 List of experiments..... 137
 - 8.4.2 Effectiveness of secure messaging scheme 138
 - 8.4.3 Localisation success..... 139
 - 8.4.4 Messaging footprint 144
- 9 Optimised Architecture for Block Chain Integration in Industrial IoT 147
 - 9.1 Architecture 147
 - 9.1.1 Raspberry Pi Module for Data Collection and Processing 147
 - 9.1.2 Decentralized IPFS Storage and Blockchain Smart Contract Module 147
 - 9.1.3 User DApp Module for Data Retrieval and Monitoring 148
 - 9.2 Aim of Experiment 148
 - 9.2.1 Overall Aim 148
 - 9.2.2 Research Question 1 (RQ1): Privacy and Trust in IIoT 148
 - 9.2.3 Research Question 2 (RQ2): Integration of Blockchain with IIoT: 148
 - 9.2.4 Research Question 3 (RQ3): Evaluation of Blockchain Integration in IIoT..... 148
 - 9.3 Background Theory..... 148
 - 9.3.1 Technological Evolution 148
 - 9.3.2 Industrial IoT (IIoT) and Industry 4.0..... 149
 - 9.3.3 Challenges in IIoT and Role of Blockchain..... 149
 - 9.3.4 Scope of Research 149
 - 9.4 Research Motivations 149
 - 9.4.1 Privacy and Trust Mechanisms..... 149

- 9.4.2 Performance Evaluation 149
- 9.5 Experimental Setup 150
 - 9.5.1 Proof-of-Concept Implementation..... 150
 - 9.5.2 Key Components and Functions..... 150
- 9.6 Experimental Procedure 150
- 9.7 Experimental Results and Analysis 150
 - 9.7.1 Performance Evaluation and Proposed Architecture 150
 - 9.7.2 Challenges and Future Directions 151
- 10 Conclusions..... 152
- References..... 154
- Appendix I Casino Cards setup..... 158
- Appendix II Power up DU & RU Sequence. 161
- Appendix III Operate DU and RU for Distance measurement 162
- Appendix IV Position estimation..... 163
 - IV.1 Appendix IV-1. Position 1 Estimation 163
 - IV.2 Appendix IV-2. Position 11 Estimation 165
 - IV.3 Appendix IV-3. Position 16 Estimation 167

List of figures

<i>Figure 3.1.1: Technical Team from RunEL</i>	23
<i>Figure 3.2.1: Remote Unit</i>	24
<i>Figure 3.2.2: Graphic User Interface of Results</i>	25
<i>Figure 3.2.3: TDoA Experimental Setup for Location Measurement</i>	25
<i>Figure 3.2.4: Experimental Set Up</i>	26
<i>Figure 3.2.5: 5G Remote Unit and Splitter</i>	27
<i>Figure 3.2.6: TDoA Experimental Setup with directional UE antenna for Location Measurement</i>	27
<i>Figure 3.3.1: TDoA Experimental Setup for Location Measurement with single UE Tx Antenna Array</i>	29
<i>Figure 3.3.2: Test 1 - Measured Distance between adjacent points without correction factor of 1.5 for single UE transmit antenna</i>	30
<i>Figure 3.3.3: Test 1 - Measured Distance between adjacent points with correction factor of 1.5 for single UE transmit antenna</i>	30
<i>Figure 3.3.4: Test 1 - Measured Distance from zero without correction factor of 1.5 for single UE transmit antenna</i>	31
<i>Figure 3.3.5: Test 1 - Measured Distance from zero with correction factor of 1.5 for single UE transmit antenna</i>	32
<i>Figure 3.3.6: Test 2 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna</i>	33
<i>Figure 3.3.7: Test 2 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna</i>	33
<i>Figure 3.3.8: Test 2 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna</i>	34
<i>Figure 3.3.9: Test 2 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna</i> ..	35
<i>Figure 3.3.10: Test 3 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna</i>	36
<i>Figure 3.3.11: Test 3 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna</i>	36
<i>Figure 3.3.12: Test 3 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna</i>	37
<i>Figure 3.3.13: Test 3 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna</i> ..	38
<i>Figure 3.3.14: Tests 1, 2, 3 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna</i>	38
<i>Figure 3.3.15: Tests 1, 2, 3 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna</i>	39
<i>Figure 3.3.16: Tests 1, 2, 3 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna</i>	39
<i>Figure 3.3.17: Tests 1, 2, 3 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna</i>	40
<i>Figure 3.3.18: Test 1 with absorbers - Measured Distance between adjacent points without 1.5 correction factor for single UE transmit antenna</i>	41
<i>Figure 3.3.19: Test 1 with absorbers - Measured Distance between adjacent points with 1.5 correction factor for single UE transmit antenna</i>	41
<i>Figure 3.3.20: Test 1 with absorbers - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna</i>	42
<i>Figure 3.3.21: Test 1 with absorbers - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna</i>	43
<i>Figure 3.3.22: TDoA Experimental Setup for Location Measurement with single UE Tx Antenna Array and reference UE Tx Antenna Array</i>	43
<i>Figure 3.3.23: Experimental Set Up with EM absorbent material</i>	44
<i>Figure 3.3.24: Typical attenuation of EM absorbent material</i>	44
<i>Figure 3.3.25: Test 1 - Measured Distance between adjacent points without 1.5 correction factor using Ref Tx and UE transmit antenna</i>	45
<i>Figure 3.3.26: Test 1 - Measured Distance between adjacent points with 1.5 correction factor using Ref Tx and UE transmit antenna</i>	46

Figure 3.3.27: Test 1 - Measured Distance from zero without 1.5 correction factor using Ref Tx and UE transmit antenna 47

Figure 3.3.28: Test 1 - Measured Distance from zero with 1.5 correction factor using Ref Tx and UE transmit antenna 47

Figure 3.3.29: Test 2 with absorbers - Measured Distance between adjacent points without 1.5 correction factor for single UE transmit antenna 48

Figure 3.3.30: Test 2 with absorbers - Measured Distance between adjacent points with 1.5 correction factor for single UE transmit antenna 49

Figure 3.3.31: Test 2 with absorbers - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna 50

Figure 3.3.32: Test 2 with absorbers - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna 50

Figure 3.3.33: Test 1,2 - Measured Distance from adjacent points without 1.5 correction factor using Ref Tx and UE transmit antenna 51

Figure 3.3.34: Test 1, 2 - Measured Distance from adjacent points with 1.5 correction factor using Ref Tx and UE transmit antenna 51

Figure 3.3.35: Test 1, 2 - Measured Distance from zero without 1.5 correction factor using Ref Tx and UE transmit antenna 52

Figure 3.3.36: Test 1, 2, 3 - Measured Distance from zero with 1.5 correction factor using Ref Tx and UE transmit antenna 52

Figure 3.4.1: Frequency Distribution of measurements for RNTI=220, SECT=24 58

Figure 3.4.2: Time series of measurement data for RNTI=220, SECT=24 58

Figure 3.4.3: Frequency Distribution of measurements for RNTI=220, SECT=32 59

Figure 3.4.4: Time series of measurement data for RNTI=220, SECT=32 59

Figure 3.4.5: Frequency Distribution of measurements for RNTI=220, SECT=40 60

Figure 3.4.6: Time series of measurement data for RNTI=220, SECT=40 60

Figure 3.4.7: Frequency Distribution of measurements for RNTI=220, SECT=48 61

Figure 3.4.8: Time series of measurement data for RNTI=220, SECT=48 61

Figure 3.4.9: Frequency Distribution of measurements for RNTI=220, SECT=64 62

Figure 3.4.10: Time series of measurement data for RNTI=220, SECT=64 62

Figure 3.4.11: Frequency Distribution of measurements for RNTI=220, SECT=72 63

Figure 3.4.12: Time series of measurement data for RNTI=220, SECT=72 63

Figure 3.4.13: Frequency Distribution of measurements for RNTI=220, SECT=80 64

Figure 3.4.14: Time series of measurement data for RNTI=220, SECT=80 64

Figure 3.4.15: Frequency Distribution of measurements for RNTI=220, SECT=88 65

Figure 3.4.16: Time series of measurement data for RNTI=220, SECT=24 65

Figure 4.2.1: The CDFs of the position errors of the UWB and LIDAR-assisted UWB [4-16] 70

Figure 4.2.2: Illustration of the steps for the landmark detection using a stereo camera [4-18] 71

Figure 4.2.3: Source of errors in both Fast R-CNN and YOLO Detection [4-24] 73

Figure 4.3.1: Complete System Flowchart 75

Figure 4.3.2: Initial Environment State 76

Figure 4.3.3: Deployed Testing Environment 77

Figure 4.3.4: Ricoh Theta V camera [4-26] 77

Figure 4.3.5: Designing and Creating the camera mount 78

Figure 4.3.6: The Ouster OS1 LIDAR 78

Figure 4.3.7: Dell PowerEdge R740 [4-29] 80

Figure 4.3.8: Code required to interface with the LIDAR 80

Figure 4.3.9: Code to show the Reflectivity layer in real-time 80

Figure 4.3.10: Output of the Reflectivity layer 81

Figure 4.3.11: YOLOv5 Model Flow 82

Figure 4.3.12: Configuring the LIDAR on Ouster Studio 82

Figure 4.3.13: Code to move the 3D Plotter 83

Figure 4.3.14: Code and output in slicing LIDAR recordings to images 84

Figure 4.3.15: Using Roboflow to annotate images 84

Figure 4.3.16: Configuring the training file 85

Figure 4.3.17: YOLOv5 Training Cycle 85

Figure 4.3.18: Running the YOLOv5 Trainer 86

Figure 4.3.19: Confusion Matrix for the newly trained weights file	86
Figure 4.3.20: Precision Recall and the F1-Confidence.....	87
Figure 4.3.21: LIDAR Position Estimation	88
Figure 4.3.22: Configuring parameters for detection.....	88
Figure 4.3.23: YOLO output on LIDAR recording	88
Figure 4.3.24: YOLO output with a blocked shape	88
Figure 4.3.25: Measuring Distance	89
Figure 4.3.26: Code to calculate distance	90
Figure 4.3.27: LIDAR output with measured distance	90
Figure 4.3.28: Applying the KF on distances.....	91
Figure 4.3.29: Kalman filter implementation	91
Figure 4.3.30: Code implementing KF use	92
Figure 4.3.31: Process to calculate position using trilateration	92
Figure 4.3.32: Leica Disto D2 Laser Measuring tool.....	93
Figure 4.3.33: Mapping the system environment	93
Figure 4.3.34: Coding in the trilateration calculation	94
Figure 4.3.35: Displaying a shape	94
Figure 4.3.36: Displaying the exterior frame and plotter.....	94
Figure 4.3.37: Displaying circles around landmarks.....	95
Figure 4.3.38: Displaying the estimated position.....	95
Figure 4.3.39: Real time mapping of the estimated position	95
Figure 4.3.40: Changing the code to run in real-time	96
Figure 4.3.41: Camera and LIDAR sensor fusion flow	96
Figure 4.3.42: Wirelessly capturing images	97
Figure 4.3.43: 360-degree camera P-R and F1-Confidence Curves	97
Figure 4.3.44: Running YOLO through the 360-degree camera	98
Figure 4.4.1: Recording locations on the 3D plotter.....	99
Figure 4.4.2: Code calculates a mean average error between estimated & actual values	99
Figure 4.4.3: Estimated Position Error map	100
Figure 5.1.1: Data combination for localization.....	104
Figure 5.2.1: Architecture of the proposed hybrid localization system.	105
Figure 6.2.1: Illustrations of system model of a typical federated learning network.....	110
Figure 6.3.1: Hierarchy of federated learning infrastructure for location Estimation.....	112
Figure 6.4.1: Training performance Vs Validation performance	114
Figure 7.1.1: The VLP system model.....	116
Figure 7.1.2: Example of positioning beacon design with the four LEDs.....	118
Figure 7.2.1: The proposed high-precision pulse reconstruction structure.	119
Figure 7.3.1: The structure of the semi-physical experimental testbed.	123
Figure 7.4.1: The received beacon waveform at the center point.....	124
Figure 7.4.2: The CDF related to the location estimation error with and without high-precision pulse reconstruction.	124
Figure 7.4.3(a) The CDF versus the transmit SNR. (b) The CDF versus the transmit SNR.	125
Figure 8.2.1. Abstract scenario for secure messaging scheme.....	127
Figure 8.2.2. Iterative Trilateration.....	128
Figure 8.2.3. Public Key List distribution after device on-boarding	129
Figure 8.3.1. Grid scenario to derive different coverage levels	137
Figure 8.4.1. Fraction of agents with latest Public Key list version for one simulation run.....	138
Figure 8.4.2. Fraction of agents with latest Public Key list for varying communication ranges.....	139
Figure 8.4.3. Fraction of agents with latest Public Key list for varying fixed trilateration coverage levels.	139
Figure 8.4.4. Trilateration success for one simulation run	140
Figure 8.4.5. Trilateration success for varying communication ranges.....	141
Figure 8.4.6. Trilateration success for varying FTC levels.....	141
Figure 8.4.7. Trilateration success for varying wireless latencies	141
Figure 8.4.8. Trilateration success for varying number of devices within the scenario.....	142
Figure 8.4.9. Improvement of trilateration success for varying FTCs and communication ranges	143
Figure 8.4.10. Improvement of trilateration success for varying FTCs and number of emitted devices	143

Figure 8.4.11. Improvement of trilateration success for varying number of devices and communication ranges 144

Figure 8.4.12. Number of sent messages for varying FTC values..... 145

Figure 8.4.13. Number of sent messages for varying wireless latencies..... 145

Figure 8.4.14. Proportion of message types for varying communication ranges..... 146

Figure 9.1.1: Proposed System Architecture 147

List of tables

<i>Table 3.3.1: Test 1 - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna</i>	29
<i>Table 3.3.2: Test 1 - Numerical Results of Measured Distance from zero for single UE Transmit antenna</i>	31
<i>Table 3.3.3: Test 2 - Numerical Results of Measured Distance from adjacent points for single UE Transmit antenna</i>	32
<i>Table 3.3.4: Test 2 - Numerical Results of Measured Distance from zero for single UE Transmit antenna</i>	34
<i>Table 3.3.5: Test 3 - Numerical Results of Measured Distance from adjacent points for single UE Transmit antenna</i>	35
<i>Table 3.3.6: Test 3 - Numerical Results of Measured Distance from zero for single UE Transmit antenna</i>	37
<i>Table 3.3.7: Test 1 with absorbers - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna</i>	40
<i>Table 3.3.8 - Test 1 with absorbers - Numerical Results of Measured Distance from zero for single UE Transmit antenna</i>	42
<i>Table 3.3.9: Test 1 - Numerical Results of Measured Distance between Adjacent points using Ref Tx and UE Transmit antenna</i>	45
<i>Table 3.3.10: Test 1 - Numerical Results of Measured Distance from zero using Ref Tx and UE Transmit antenna</i>	46
<i>Table 3.3.11: Test 2 with absorbers - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna</i>	48
<i>Table 3.3.12: Test 2 with absorbers - Numerical Results of Measured Distance from zero for single UE Transmit antenna</i>	49
<i>Table 3.4.1: 5G FAPI by SCF – Number and Type of messages</i>	53
<i>Table 3.4.2: 5G FAPI by SCF – Data Fields</i>	54
<i>Table 3.4.3: Summary of Measurements of mean, stdev and 95% CI at 68cm.</i>	66
<i>Table 4.3.1: Proposed Functional and Technical Requirements</i>	73
<i>Table 4.3.2: Proposed Key Performance Indicators</i>	74
<i>Table 4.3.3: Ouster Data Layers</i>	79
<i>Table 4.3.4: LIDAR Configuration Parameters</i>	83
<i>Table 4.3.5: Measured Shape Coordinates</i>	93
<i>Table 4.4.1: Actual distances for each position on the 3D Plotter</i>	99
<i>Table 4.5.1: Testing the developed system against the Functional Requirements</i>	100
<i>Table 4.6.1: Testing the developed system against the KPIs</i>	101
<i>Table 5.3.1: Obtained Localization Errors Considering Both Environmental Configurations.</i>	106
<i>Table 5.3.2: Comparison of localization errors using different approaches considering the second configuration with SNR = 50 and sigma shadowing = 5.</i>	107
<i>Table 6.4.1: DNN Models' Configuration</i>	113
<i>Table 6.4.2: FL simulation settings</i>	113
<i>Table 6.4.3: Localization error comparison</i>	114

Abbreviations

5G	Fifth Generation (mobile/cellular networks)
5G PPP	5G Infrastructure Public Private Partnership
6G BRAINS	Internet of Radio Light (project)
AES	Advanced Encryption Standard
AoA	Angle of Arrival
AP	Access Point
API	Application Programming Interface
ARM	Acorn RISC Machine
BS	Base Station
CAPEX	Capital Expenditure
CDF	Cumulative distribution function
CL	Centralised Learning
CNN	Convolutional neural networks
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSP	Cross Stage Partial
DApp	Decentralised application
DL	Deep Learning
DNN	Deep Neural Network
DU	Distributed Unit
ED	End Device
EKF	Extended Kalman filter
EM	Electromagnetic
FAPI	Functional Application Platform Interface
FIFO	First In First Out

FL	Federated Learning
FPGA	Field Programmable Gate Array
FTC	Fixed trilateration coverage
GPS	Global Positioning System
ID	Identity
IDE	Integrated Development Environment
IIoT	Industrial IoT
IMU	Inertial Measurement Unit
IoT	Internet of Things
IP	Internet Protocol
IPFS	Inter-Planetary File System
IPIN	Indoor Positioning and Indoor Navigation
ITS	Intelligent Transportation Systems
JSON	Java Script Object Notation
JSON-RPC	JSON Remote procedure call
KF	Kalman filter
KPI	Key Performance Indicator
LeGO-LOAM	lightweight and ground optimized lidar odometry and mapping
LED	Light Emitting Diode
LIDAR	Light Detection and Ranging
LOAM	Lidar odometry and mapping
LOS	Line of Sight
LTS	Long Term Support
LUT	Look up table
M2M	Machine to Machine
MAE	Mean Absolute Error

MAP	Maximum a posteriori
MIMO	Multiple Input Multiple Output
NMS	Non-maximum suppression
OOK	on-off keying
OPEX	Operational Expenditure
OFDM	Orthogonal Frequency Division Multiplexing
OS	Operating System
OTFS	Orthogonal Time Frequency Space
PD	Photodetector
PK	Public Keys
QoE	Quality of Experience
QoS	Quality of Service
OWC	Optical Wireless Communications
PHY	Physical
RAM	Random Access Memory
R&D	Research and Development
R-CNN	Region-based Convolutional Neural Networks
RFID	Radio-frequency identification
RNTI	Radio Network Temporary Identifier
RPi	Raspberry Pi
RSRP	Reference Signal Received Power
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
RU	Remote Unit
Rx	Receiver
SCF	Small Cell Forum

SDK	Software Development Kit
SDN	Software Defined Networks
SISO	Single Input Single Output
SLAM	Simultaneous Localisation and Mapping
SNR	Signal to Noise Ratio
SU	Super User
SW	Software
SWOT	Strengths, Weaknesses, Opportunities, and Threats analysis
TCP	Transmission Control Protocol
TDoA	Time Difference of Arrival
ToA	Time of Arrival
Tx	Transmit
UDP	User Datagram Protocol
UE	User Equipment
UWB	Ultra Wideband
VANETS	Vehicular ad-hoc networks
VLC	Visible Light Communications
VLP	Visible Light Positioning
YOLO	You Only Look Once

1 Introduction

Section 1 introduces the reader to the experiments that are reported in this deliverable to increase localisation reliability and accuracy.

Section 2 provides a summary of how all the different localization technologies could work together to obtain localisation with centimetre accuracy and 99.9999% reliability.

Section 3 reports on results of a sub 6G measurement campaign to measure distance to centimetre accuracy using time of arrival and the repeatability of measurement results.

Section 4 reports on results of measuring location using an environmental imaging and sensing system. The motivation for this study is to estimate what localisation accuracy could be expected to be obtained from a 6G Communication and Sensing system.

Section 5 reports on data combination and selection for a Simultaneous Localisation and Mapping (SLAM) system. A two-step localization system to improve the performance of the data fusion method is presented consisting of offline and online phases.

Section 6 reports on federated learning-based localization using visible light signals, which adopts machine learning techniques, to improve the performance of VLC-based localization, which has been impaired as a result of signal multipath propagation due to scatterers in environment.

Section 7 reports on a beacon positioning signal design which is used to unlock strict time synchronisation.

Section 8 reports on a secure mutual localization system to support localisation schemes, which are highly dependent on continuous Line of Sight (LoS) access between at least three position-calibrated anchor nodes that is validated through simulation experiments.

In section 9 provides a detailed description on how to deploy Blockchain-based solution for IIoT by utilizing sharding and the Interplanetary File System (IPFS) to efficiently store, process, and retrieve data to thereby increase the scalability of IIoT solutions while ensuring privacy.

In section 10 an overall summary of conclusions to the accomplishments reported in the deliverable are provided.

2 Simultaneous Localisation and Mapping Strategy

Localization of UEs using Time of Arrival (ToA), Angle of Arrival (AoA) and Received Signal Strength (RSS) techniques can be used to obtain an accuracy of less than 1cm, however it is highly dependent on the continuous direct line of sight access between the gNB access points and the UE. If there is not direct line of sight access to four or more gNB access points then location ambiguity is introduced and so other techniques should be used to maintain localization such as dead reckoning using Inertial Measurement Unit (IMU) or using iterative multi-lateration or using position from landmarks. Since the performance of ToA localisation is also susceptible to the effects of multipath propagation of the transmitted signal, then alternative AoA and RSS techniques or location from landmarks should be used.

If there is not direct line of sight access to four or more gNB access points to obtain location using ToA or RSS then an alternative solution is to use Angle of Arrival (AoA) from more than two access points or AoA and distance from one access point, so that some form of interim measure for obtaining location can continue to be made. Note that distance can also be estimated using Orthogonal Time Frequency Space (OTFS) modulation due to its operation in the delay and Doppler domain as opposed to using Orthogonal Frequency Division Multiplexing (OFDM) which operates in the time and frequency domain.

If there is no direct line of sight access to any gNB access points, then Position from landmarks calculates position from the landmarks identified from within point cloud data of a sensed environment obtained for example from a LIDAR system or a Communication system with RADAR sensing capabilities. In order to generate as complete a reconstruction of environment this point cloud needs to be constructed from many point clouds sensed from the perspective of many different access points between UEs using distributed ledgers. These sets of point clouds recorded from different locations in the environment can then be integrated to obtain a more complete sensing of the environment.

3 Sub 6GHz measurement campaign in Diverse Environments

The cm level UE position accuracy measurement campaign is a four-phase process starting with Phase 1, which was reported in [3-1], and followed by Phases 2, 3 and 4 which is reported in this deliverable:

- PHASE-1 First to understand how the data with the Time of Arrival (ToA) will be provided to prepare your UE location measurement SW and test setup for the real-life test at Brunel
- PHASE-2 – After that to prepare the LAB for the 5G system upgrade by RunEL
- PHASE-3 – RunEL will send an Engineer to BRUNEL to upgrade the 5G Network
- PHASE-4 – Execution of the Test at BRUNEL

3.1 Aim of Experiment

The aim of the experiment is to measure the accuracy of Time of Arrival measurements between UE Tx Antenna and gNB Rx Antenna in order to measure distance. The laboratory was set up with the assistance of three engineers from RunEL, namely: Arik Salamon, Almog Karamani and Zion Hadad, shown from left to right in Figure 3.1.1

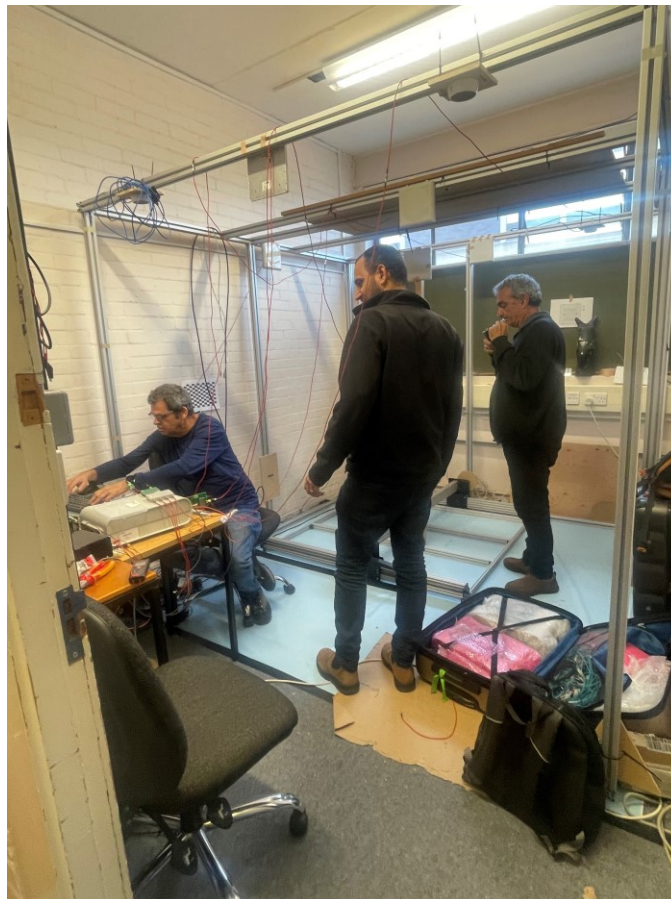


Figure 3.1.1: Technical Team from RunEL

3.2 Experimental Setup

3.2.1 Remote Unit System Setup

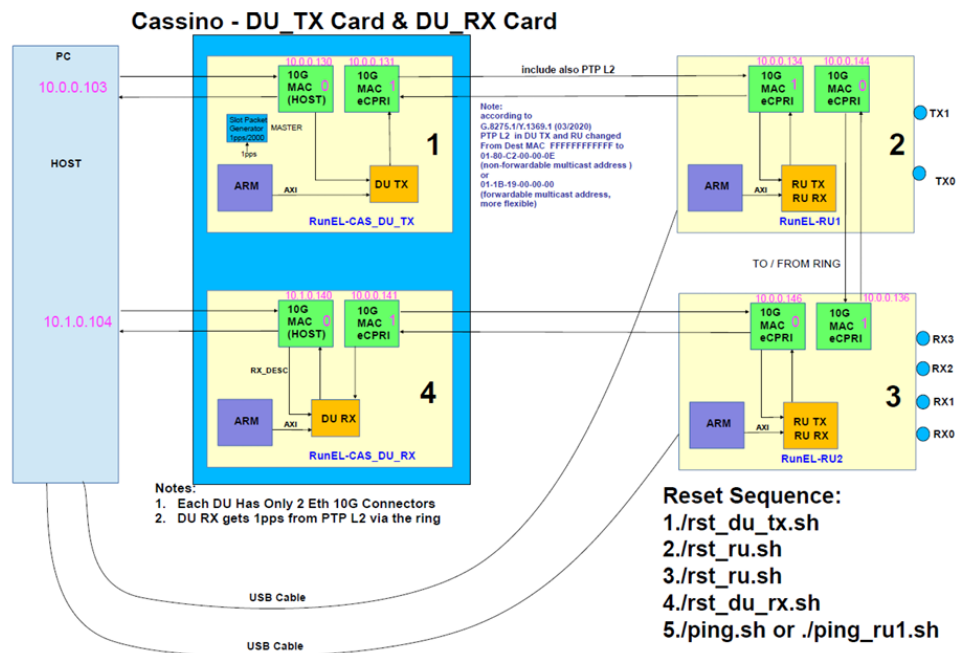


Figure 3.2.1: Remote Unit

The requirement for the Host PC is Linux Ubuntu 18.04 or 20.04 of which Brunel University is using the latter. The PC has 2x RunEL DU Cards (Cassino cards) inserted in the PCI slots of the Host server, one using as DU_TX the second card as DU_RX. The PC has 2 x 10G Ethernet ports, one should get IP 10.0.0.103, the second port should get IP 10.1.0.104.

The RunEL Cards (Cassino cards) has 4 x 10G Ethernet, of which we are using only 2 x Ethernet ports for each card. Connect with Small Form-factor Pluggable SFP+ Transceivers and Optic cables according to Figure 3.2.1. In the host we should have SU (Super User) user with the name "runel".

The RunEL (Cassino Cards) Driver then needs to be setup and the Ethernet fifo card drivers have to be activated and the whole system booted up, as explained in Appendix I, and the DU and RU powered up in a specific sequence, as explained in Appendix II.

For taking distance measurements the downlink, distance program and uplink shells need to be initiated in the specific order, as explained in Appendix III. The results are collected through Runel's distance software, which displays the results of averaging of the measurement data as shown in Figure 3.2.2 below.

As shown in the figure, it is divided into sections. There are 2 Tx Antenna, 4 RX Antenna. The 2 Tx antenna each one has 4 sections

- [24,32,40,48]
- [64,72,80,88]

The 4 Rx antenna each receive in 2 sections

- [24,64]
- [32,72]

- [40,80]
- [48,88]

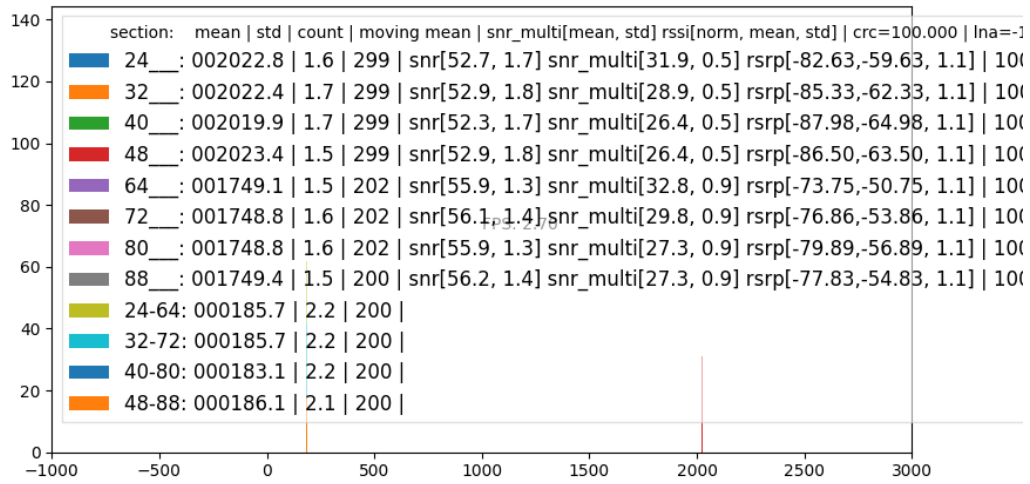


Figure 3.2.2: Graphic User Interface of Results

3.2.2 Initial Experimental Setup

The Time Difference of Arrival (TDoA) experimental setup is shown in Figure 3.2.3. The Remote unit has IP address 10.1.0.140. and the Distributed Unit (Dell 740 Server) 10.1.0.104.

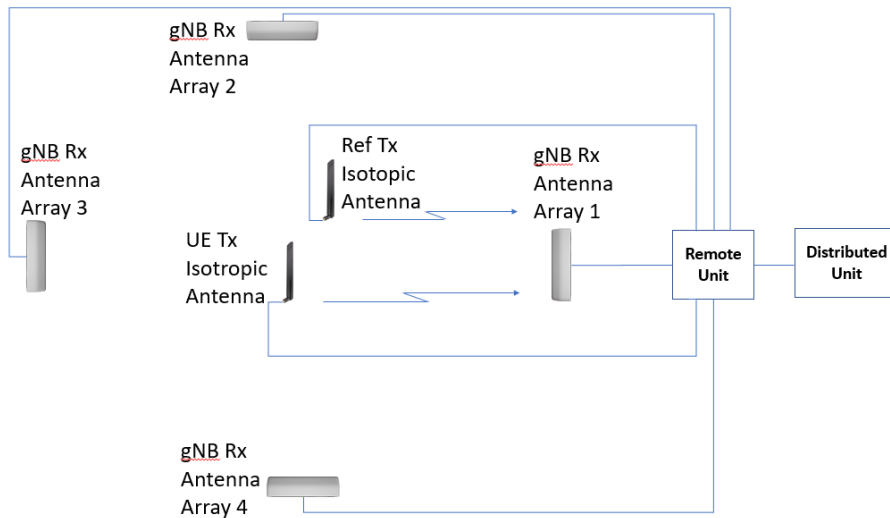


Figure 3.2.3: TDoA Experimental Setup for Location Measurement

Initially Isotropic Antennas were intended for the Ref Tx Antenna and the UE Tx Antenna so that all four gNB Rx Antenna Arrays would be able to receive the OFDM signal from them, as shown in Figure 3.2.3 but the received signal was not sufficiently above the noise floor to detect and process them.

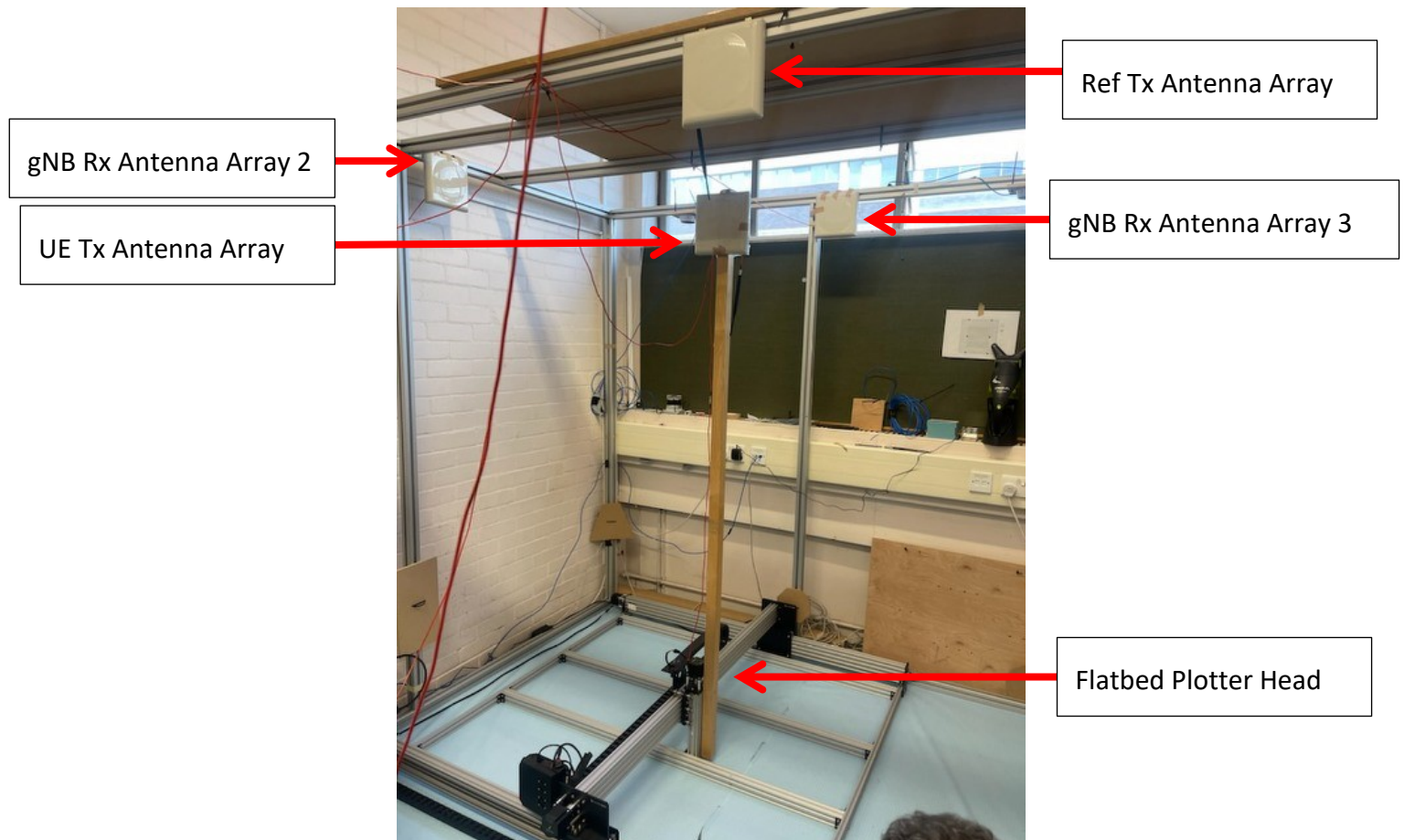


Figure 3.2.4: Experimental Set Up

Figure 3.2.4 shows the experimental set from the view point of gNB Rx Antenna Array 1, which is directed towards Ref Tx Antenna Array and the UE Tx Antenna Array, which is fixed to the flat bed plotter that is able to move co linearly closer or further from it. In the background can be seen gNB Rx Antenna Array 2 and gNB Rx Antenna Array 3. The Ref Tx Antenna Array is located at a known distance from the gNB Rx Antenna Array 1.

Instead, a directional antenna array was needed for the Ref Tx Antenna and UE Tx Antenna so that a sufficiently strong received signal above the noise floor could be obtained, as shown in Figure 3.2.6, the remote unit of which is shown in Figure 3.2.5, the circuit of which is shown in Figure 3.2.6.

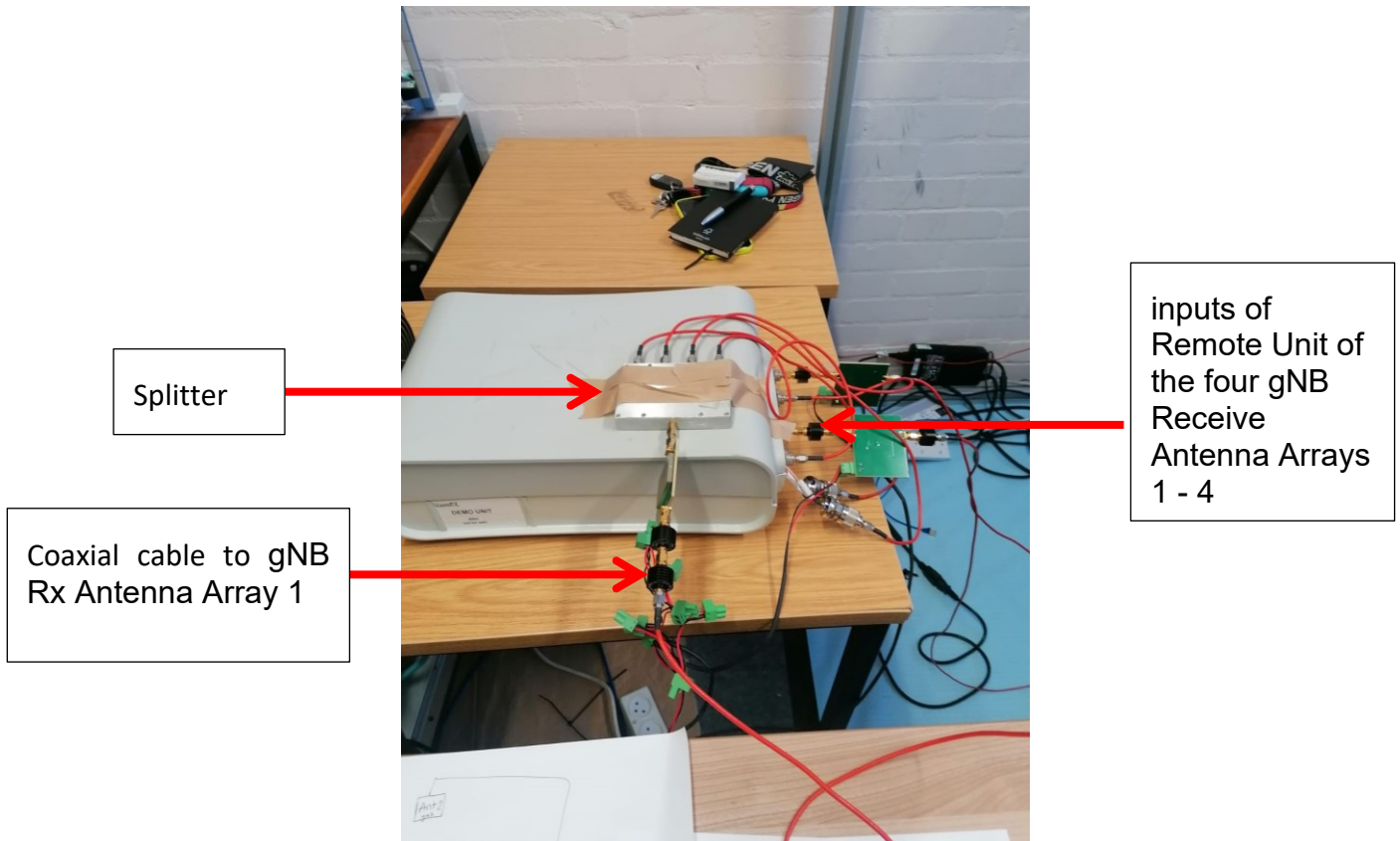


Figure 3.2.5: 5G Remote Unit and Splitter

Since the system was calibrated using coaxial cable end-to-end connections, compensation is required to be made for the segment of the end-to-end connection that was replaced by free space transmission.

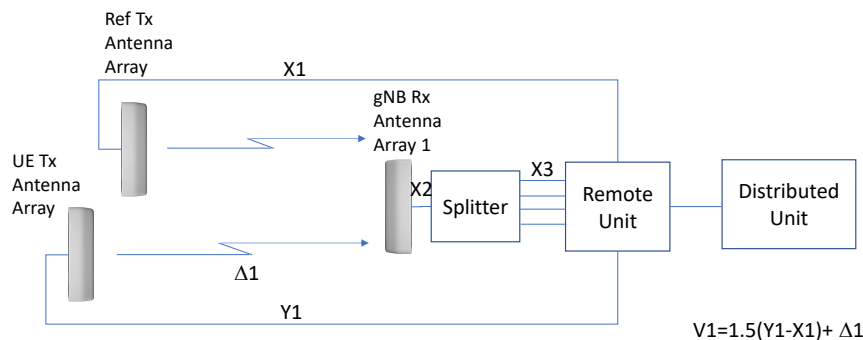


Figure 3.2.6: TDoA Experimental Setup with directional UE antenna for Location Measurement

The speed of light in vacuum is $c = 2.998 \times 10^8$ m/s, whereas the speed of an electrical signal in coaxial cable is about 2/3 of this, which is 1.998×10^8 m/s. The system is calibrated over coax with 0.66 c.

The calibration is complicated to calculate as it should include the delays in splitters, filters, Clocks synchronization etc. The best way to do a calibration as follows:

1. Calibration With Coax: put two different coax length (1 meter and 2 meters for example) between TX and RX and calibrate the measured distance according to the difference between the coax length (1 meter in the example).

2. Measurement Over the air: put the UE antenna in two different positions (1 meter and 2 meters for example) between TX and RX and measure the distance according the measure range which should be 2/3 of the measurement in case 1 (Therefore you need to multiply it by a correction factor of 1.5)

$(Y1 - X1)$ is the difference in path lengths of the coax cables between the UE Tx Antenna Array and Ref Tx Antenna Array, compensated by the ratios of speed of light in free space and the speed of light in coax cable = 1.5.

$\Delta 1$ is the difference in path lengths in free space between the UE Tx Antenna Array and Ref Tx Antenna Array.

$V1$ is the compensation required due to the different path lengths and propagation times in different media.

$$V1=1.5(Y1-X1)+ \Delta 1$$

Where:

$X1$ is the propagation time in coax cable between the Remote Unit and the UE Tx Antenna

$Y1$ is the propagation time in coax cable between the Remote Unit and the Ref Tx Antenna

$\Delta 1$ is the difference in propagation time in Free Space between UE Tx Antenna and the Ref Tx Antenna Array

A Flat Bed Plotter, presented in [3-2], is used to move the UE Tx Antenna Array.

3.3 Experimental Results and Analysis

3.3.1 Experiment 1: Same UE Tx and Ref Tx antennas

System Setup 1:

A directional antenna array was needed for the Ref Tx Antenna and UE Tx Antenna so that a sufficiently strong received signal above the noise floor could be obtained by the gNB Rx Antenna Array, which was split into four with a Splitter and fed into the intended inputs of Remote Unit of the four gNB Receive Antenna Arrays 1 - 4, as shown in Figure 3.2.1. Since the antenna array was not able to be beamsteered, then the UE Tx Antenna Array, Ref Tx Antenna Array and gNB Rx Antenna Array were required to be co-linear with each other, as shown in Figure 3.3.1 in order to receive a sufficiently strong signal above the noise floor to signal process the OFDM for obtaining ToA.

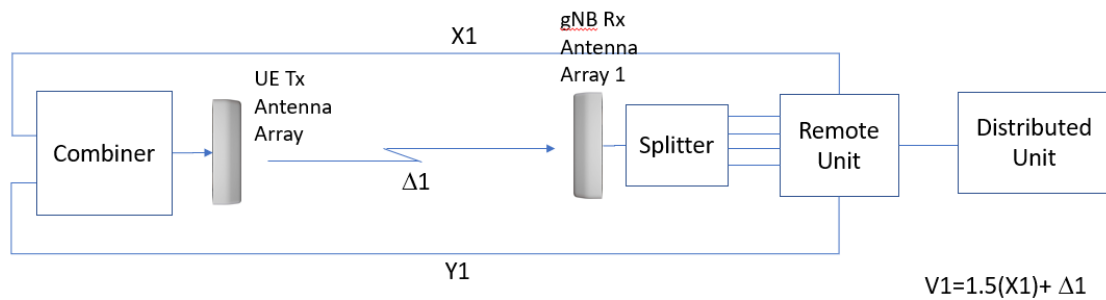


Figure 3.3.1: TDoA Experimental Setup for Location Measurement with single UE Tx Antenna Array

Zion and Kareem put the capabilities of the system through the following three tests in order to evaluate its performance. In order to make up for the lack of time during the integration meeting to collect additional data, measurements were taken at regular intervals of 20 centimeters. In addition, the EM absorbers had not yet been delivered, which meant that they could not be used during the integration meeting to absorb multipath reflections. Three tests were performed Test 1, 2 and 3 without absorbers (due to unavailability) for repeatability. One test was performed with absorbers when they were delivered.

3.3.1.1 Test 1

Table 3.3.1: Test 1 - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna

Test 1 (Zion & Kareem) Measured Distance Between Adjacent												
x (cm)	x (mm)	cm	Mean TX	Ratio of premitivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)
0	0		1889	1.5	10	15	10	-5	10	5	50.0%	25%
20	200	20	1899		12	18	8	-2	8	2	20.0%	5%
40	400	20	1911		14	21	6	1	6	1	10.0%	2%
60	600	20	1925		14	21	6	1	6	1	7.5%	1%
80	800	20	1939		106	159	-86	139	86	139	86.0%	139%
100	1000	20	2045			0						
120	1200	20										
TOTAL:		120				156	234					
							Average		23.2	29.6		
							STDEV		31.4	54.7		

The measured data are presented in Table 3.3.1 above, which corresponds to the experimental setup 1. Based on the data presented in the table, the recorded distance travelled was 120 centimetres, whereas the measured distance amounted to 156 cm, with increments of 20 centimetres. Based on the results obtained, it was determined that the mean score was 23.2. Nevertheless, upon considering the correction factor of 1.5, the score exhibited a notable increase to 29.6. Similarly, in accordance with the standard deviation, the initial value was recorded as 31.4, however, it subsequently exhibited an increase to 54.7.

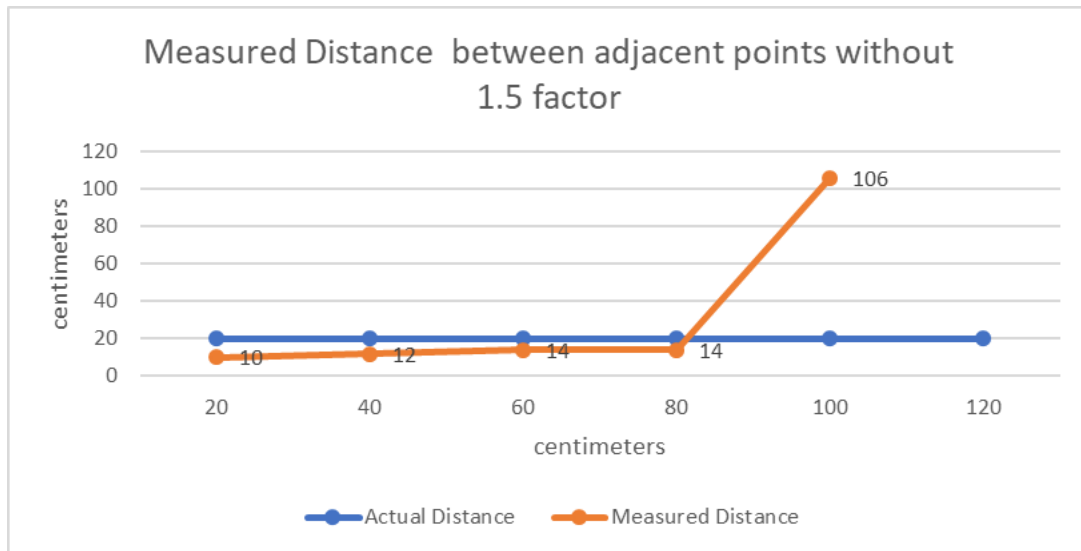


Figure 3.3.2: Test 1 - Measured Distance between adjacent points without correction factor of 1.5 for single UE transmit antenna

The illustration that is displayed above shows that the distance that is measured between two points that are adjacent to one another closely approximating the distance that is actually present when the correction factor of 1.5 is not applied to the measurement. Despite this, it is clear that after a distance of 60 cm, the measurements that were recorded began to diverge progressively more from the actual distance.

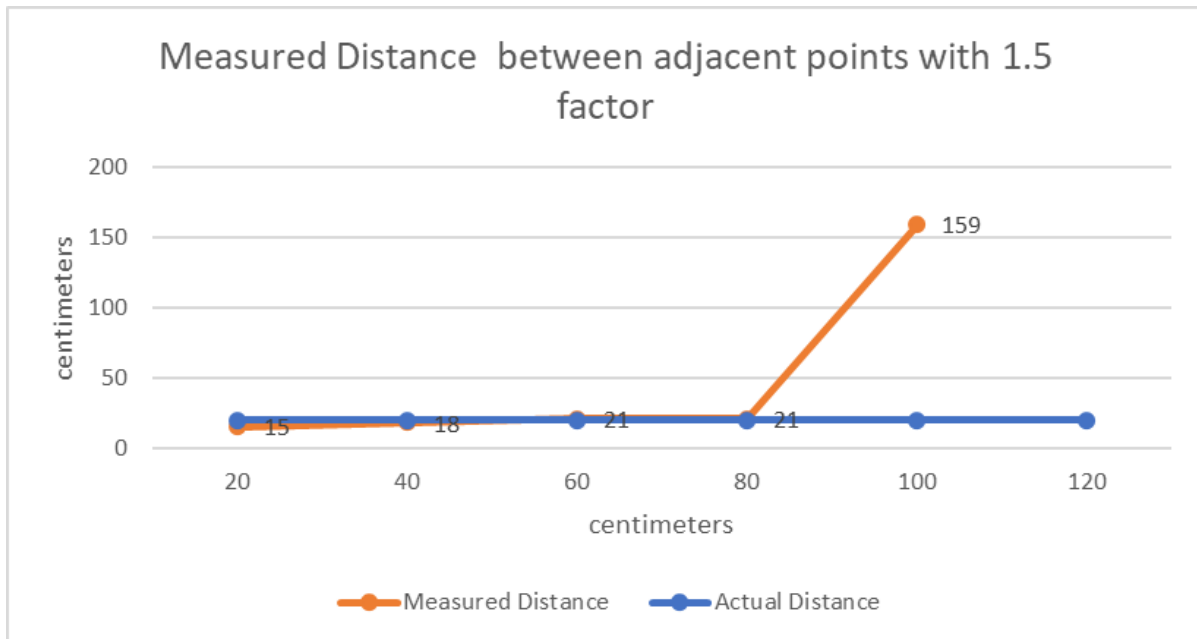


Figure 3.3.3: Test 1 - Measured Distance between adjacent points with correction factor of 1.5 for single UE transmit antenna

The provided illustration demonstrates that the measured distance between two adjacent points closely approximates the actual distance when a correction factor of 1.5 is applied. The application of this factor resulted in improved accuracy, as the measurement error decreased from 10% to 2%. However, it is evident that beyond a distance of 60 cm, the recorded measurements exhibited an increasingly noticeable deviation from the true distance.

Table 3.3.2: Test 1 - Numerical Results of Measured Distance from zero for single UE Transmit antenna

Test 1 (Zion & Kareem) Measured Distance From Zero												
x (cm)	x (mm)	cm	Mean TX	Ratio of premitivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		1889	1.5								
20	200	20	1899		10	15	-10	-5	10	5	50.0%	25%
40	400	20	1911		22	33	-18	-7	18	7	45.0%	18%
60	600	20	1925		36	54	-24	-6	24	6	40.0%	10%
80	800	20	1939		50	75	-30	-5	30	5	37.5%	6%
100	1000	20	2045		156	234	56	134	56	134	56.0%	134%
120	1200	20										
TOTAL:		120			274	411						
									Average	27.6	31.4	
									STDEV	15.7	51.3	

The measured data are presented in the Table 3.3.2 above, which corresponds to experimental setup. According to the table's data, the recorded distance traveled was 100 cm, while the measured distance was 274 cm, with increments of 20 cm. Based on the obtained results, the mean average was determined to be 38.8. However, after considering the correction factor of 1.5, the score increased significantly to 64.2. In accordance with the standard deviation, the initial value was initially recorded as 49.5, but it subsequently increased to 76.9.

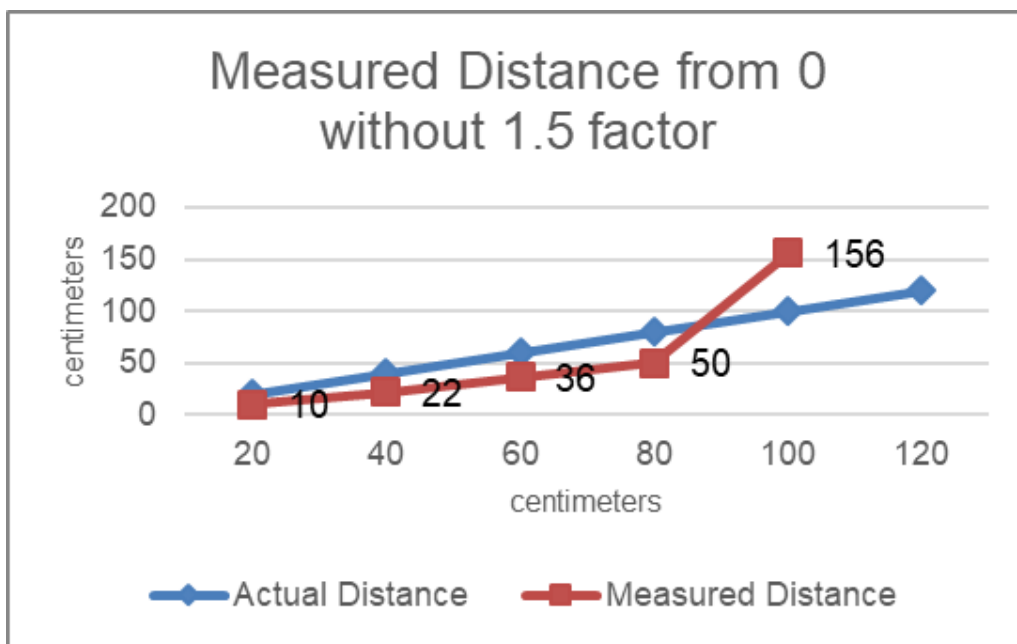


Figure 3.3.4: Test 1 - Measured Distance from zero without correction factor of 1.5 for single UE transmit antenna

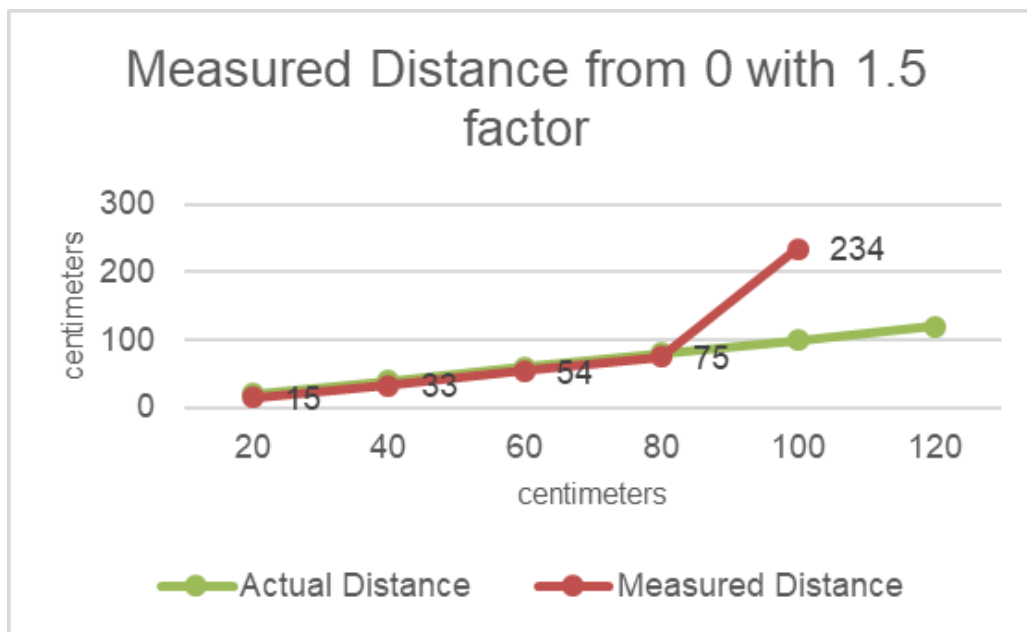


Figure 3.3.5: Test 1 - Measured Distance from zero with correction factor of 1.5 for single UE transmit antenna

When the 1.5 correction factor is not taken into account, the measurements that start from 0 produce results that are unsatisfactory, as the diagram illustrates. In spite of this, the outcomes showed signs of improvement after the 1.5 correction factor was taken into account.

3.3.1.2 Test 2

The test 2 was performed to assess repeatability of measurement results.

Table 3.3.3: Test 2 - Numerical Results of Measured Distance from adjacent points for single UE Transmit antenna

Test 2 (Zion & Kareem) Measured Distance Between Adjacent											
x (mm)	cm	Mean TX	Ratio of permittivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)
0		1983	1.5	11	16.5	-9	-3.5	9	3.5	45.0%	18%
200	20	1994		16	24	-4	4	4	4	10.0%	10%
400	20	2010		15	22.5	-5	2.5	5	2.5	8.3%	4%
600	20	2025		9	13.5	-11	-6.5	11	6.5	13.8%	8%
800	20	2034		11	16.5	-9	-3.5	9	3.5	9.0%	4%
1000	20	2045				0					
1200	20										
Σ:	120			62	93						
							Average	7.6	4.0		
							STDEV	2.7	1.3		

The measured data are presented in the Table 3.3.3 presented above, which corresponds to experimental setup. According to the table's data, the recorded distance travelled was 100 cm, while the measured distance was 62 cm, with increments of 20 cm. Based on the obtained results, the mean average was determined to be 7.6. However, after considering the 1.5 correction factor, the average reduced significantly to 4. In accordance with the standard deviation, the initial value was initially recorded as 2.7, but it subsequently decreased to 1.3.

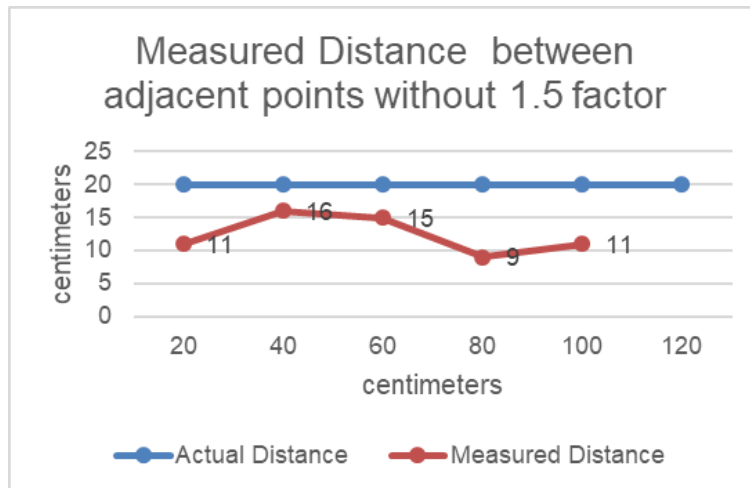


Figure 3.3.6: Test 2 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna

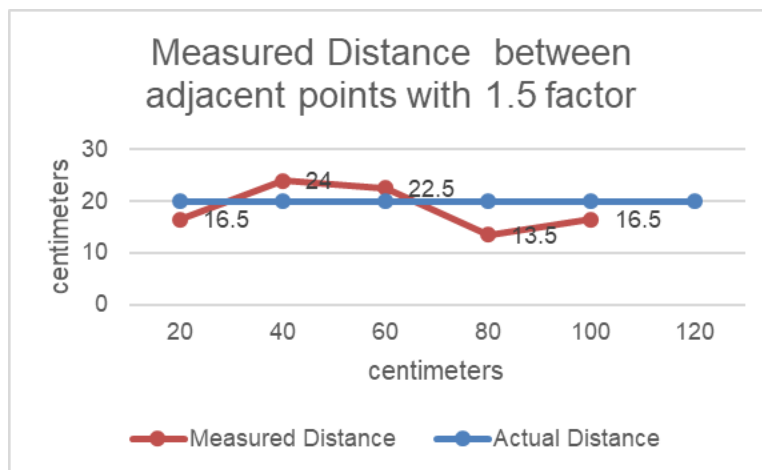


Figure 3.3.7: Test 2 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna

Figure 3.3.7 above shows, the measuring distance from adjacent without the 1.5 correction factor results in poor results, hence when the 1.5 correction factor was considered the results improved tremendously.

Table 3.3.4: Test 2 - Numerical Results of Measured Distance from zero for single UE Transmit antenna

Test 2 (Zion & Kareem) Measured Distance From Zero												
x (cm)	x (mm)	cm	Mean TX	Ratio of permittivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		1983	1.5								
20	200	20	1994		11	16.5	-9	-3.5	9	3.5	45.0%	18%
40	400	20	2010		27	40.5	-13	0.5	13	0.5	32.5%	1%
60	600	20	2025		42	63	-18	3	18	3	30.0%	5%
80	800	20	2034		51	76.5	-29	-3.5	29	3.5	36.3%	4%
100	1000	20	2045		62	93	-38	-7	38	7	38.0%	7%
120	1200	20										
TOTAL:		120			193	289.5						
									Average	21.4	3.5	
									STDEV	10.7	2.1	

The measured data are presented in the Table 3.3.4 presented above, which corresponds to experimental setup. According to the table's data, the recorded distance travelled was 100 cm, while the measured distance was 193 cm (when offset from antenna is taken account - this is not shown on table), with increments of 20 cm. Based on the obtained results, the mean average was determined to be 22.2. However, after considering the 1.5 correction factor, the average increased significantly to 39.3. In accordance with the standard deviation, the initial value was initially recorded as 13.2, but it subsequently increased to 24.8.



Figure 3.3.8: Test 2 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna

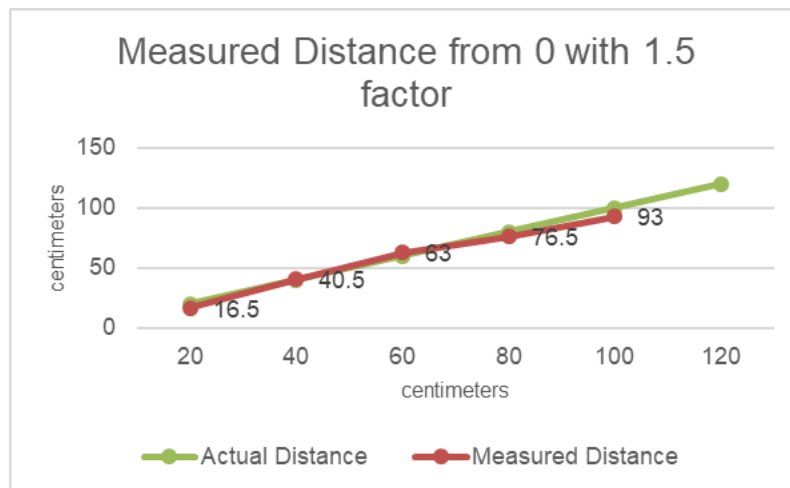


Figure 3.3.9: Test 2 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna

The diagram that is displayed above shows that the obtained results are of poor quality when the measuring distance between adjacent points is not adjusted by a correction factor of 1.5. This results in the measurement being inaccurate. On the other hand, when the 1.5 correction factor is taken into account, the results show a significant improvement.

3.3.1.3 Test 3

Test 3 was performed to assess repeatability of measurement results.

Table 3.3.5: Test 3 - Numerical Results of Measured Distance from adjacent points for single UE Transmit antenna

Test 3 (Zion & Kareem) Measured Distance Between Adjacent												
x (cm)	x (mm)	cm	Mean TX	Ratio of premitivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)
0	0		1982	1.5	12	18	-8	-2	8	2	40.0%	10%
20	200	20	1994		16	24	-4	4	4	4	10.0%	10%
40	400	20	2010		16	24	-4	4	4	4	6.7%	7%
60	600	20	2026		8	12	-12	-8	12	8	15.0%	10%
80	800	20	2034		12	18	-8	-2	8	2	8.0%	2%
100	1000	20	2046				0					
120	1200	20										
TOTAL:		120			64	96						
							Average		7.2	4.0		
							STDEV		3.0	2.2		

The measured data are presented in the Table 3.3.5 presented above, which corresponds to experimental setup. According to the table's data, the recorded distance travelled was 100 cm, while the measured distance was 64 cm, with increments of 20 cm. Based on the obtained results, the mean average was determined to be 7.2. However, after considering the 1.5 correction factor, the average reduced significantly to 4. In accordance with the standard deviation, the initial value was initially recorded as 3, but it subsequently decreased to 2.2.

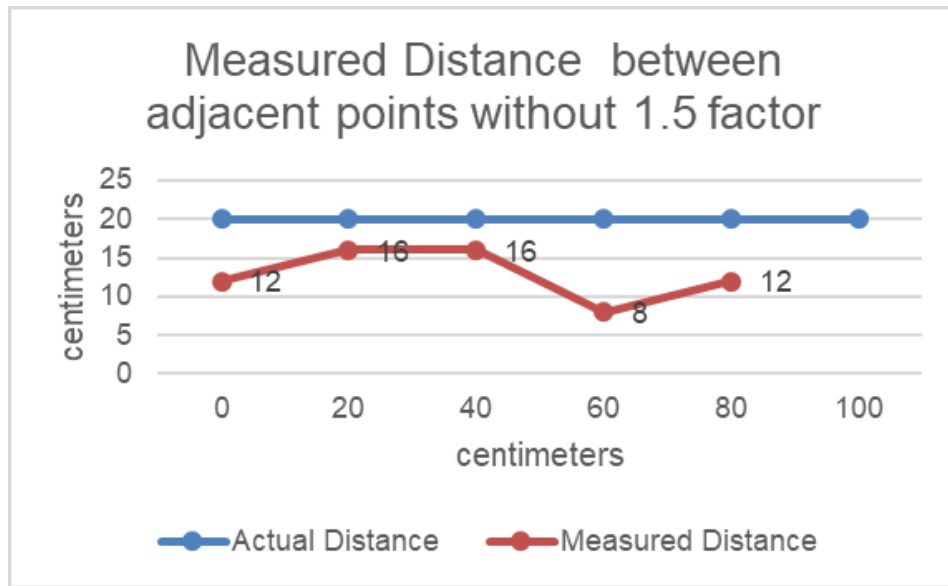


Figure 3.3.10: Test 3 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna

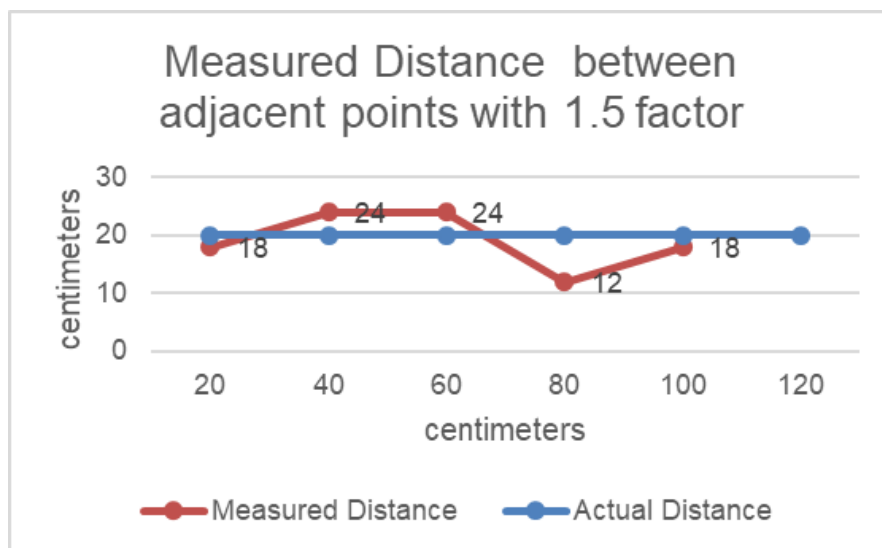


Figure 3.3.11: Test 3 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna

The diagram that is displayed above shows that the obtained results are of poor quality when the measuring distance between adjacent points is not adjusted by a correction factor of 1.5. This results in the measurement being inaccurate. On the other hand, when the 1.5 correction factor is taken into account, the results show a significant improvement as it reduced the error from 8% to 2% at 80 cm.

Table 3.3.6: Test 3 - Numerical Results of Measured Distance from zero for single UE Transmit antenna

Test 3 (Zion & Kareem) Measured Distance From Zero												
x (cm)	x (mm)	cm	Mean TX	Ratio of premitivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		1982	1.5								
20	200	20	1994		12	18	-8	-2	8	2	40.0%	10%
40	400	20	2010		28	42	-12	2	12	2	30.0%	5%
60	600	20	2026		44	66	-16	6	16	6	26.7%	10%
80	800	20	2034		52	78	-28	-2	28	2	35.0%	3%
100	1000	20	2046		64	96	-36	-4	36	4	36.0%	4%
120	1200	20										
TOTAL:		120			200	300						
									Average	20.0	3.2	
									STDEV	10.4	1.6	

The measured data are presented in the Table 3.3.6 presented above, which corresponds to experimental setup. According to the table's data, the recorded distance traveled was 100 cm, while the measured distance was 200 cm, with increments of 20 cm. Based on the obtained results, the mean average was determined to be 23.2. However, after considering the 1.5 correction factor, the average increased significantly to 40.8. In accordance with the standard deviation, the initial value was initially recorded as 13.9, but it subsequently increased to 26.2.

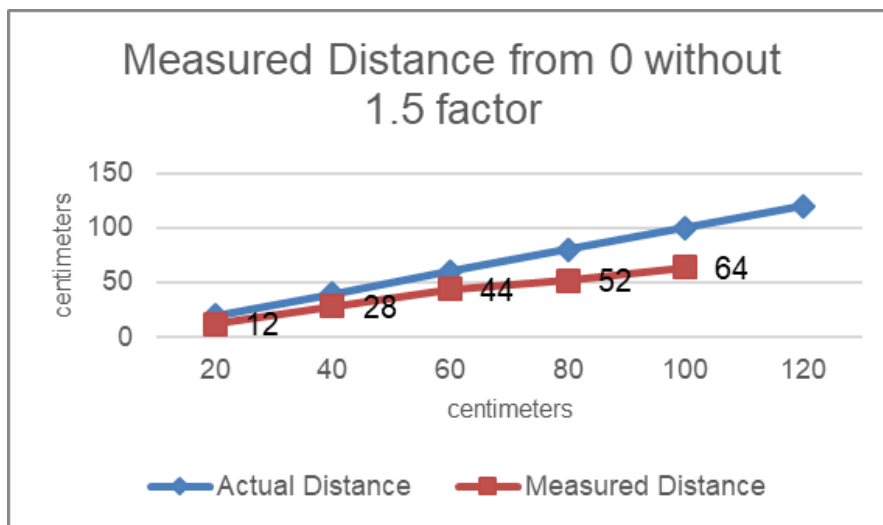


Figure 3.3.12: Test 3 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna

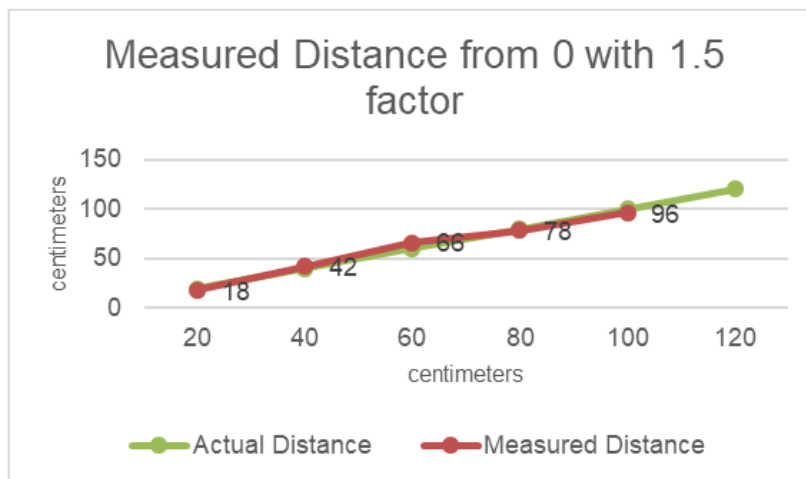


Figure 3.3.13: Test 3 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna

The diagram that is displayed above shows that the obtained results are of poor quality when the measuring distance between adjacent points is not adjusted by a correction factor of 1.5. This results in the measurement being inaccurate. On the other hand, when the 1.5 correction factor is taken into account, the results show a significant improvement.

Repeatability comparison of the above 3 tests:

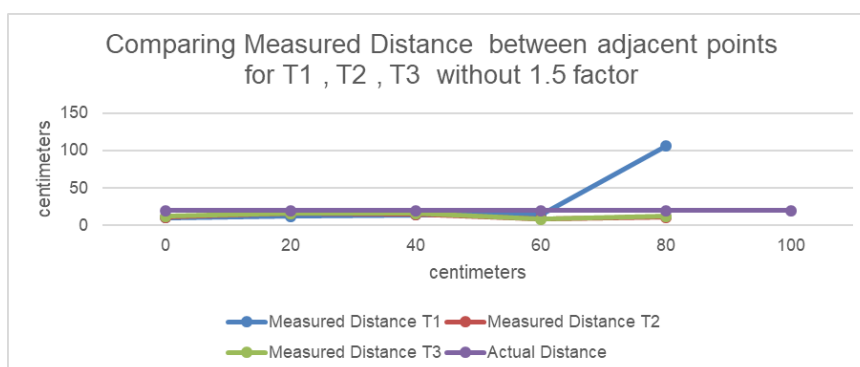


Figure 3.3.14: Tests 1, 2, 3 - Measured Distance from adjacent points without 1.5 correction factor for single UE transmit antenna

The measured distance Test1 (T1) exhibited a high level of stability and demonstrated similarity to the other measured distances. Nevertheless, it is evident that T1 experiences a significant increase beyond the 60cm mark. In general, upon comparing these tests, it becomes evident that the data exhibits a consistent pattern, thereby enabling us to infer that the data possesses reliability and reproducibility.

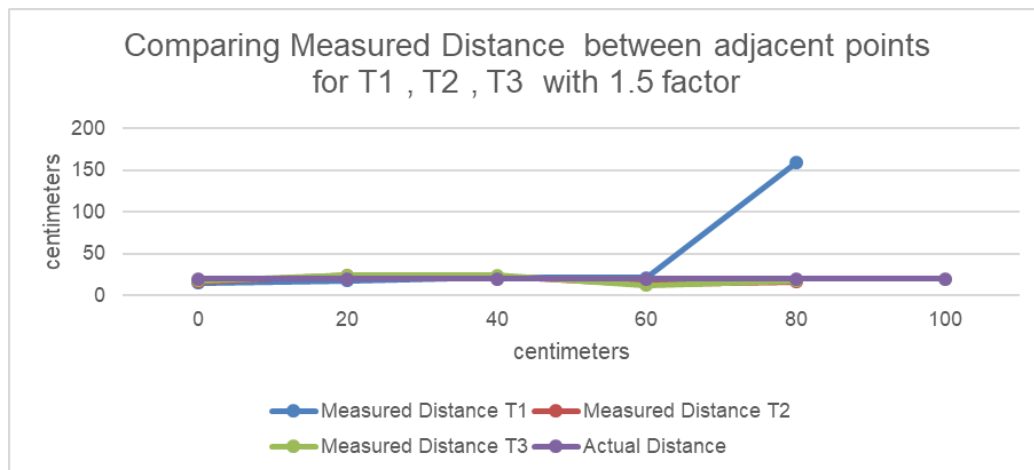


Figure 3.3.15: Tests 1, 2, 3 - Measured Distance from adjacent points with 1.5 correction factor for single UE transmit antenna

The observed distance T1 exhibited a high level of stability and demonstrated similarity to the other measured distances. Nevertheless, it is evident that T1 experiences a significant increase beyond the 60cm mark. In general, upon comparing these tests, it becomes evident that the data exhibits a consistent pattern, thereby enabling us to infer that the data possesses reliability and reproducibility.

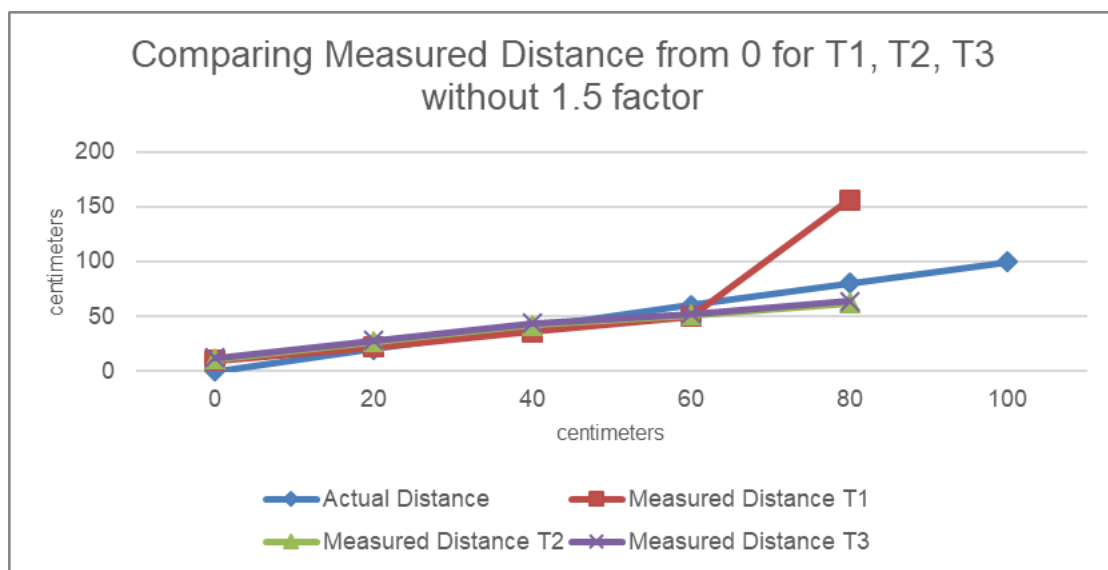


Figure 3.3.16: Tests 1, 2, 3 - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna

The measured distance T1 was very stable and similar to the other measured distance. However, it can be seen that T1 increases drastically after 60cm. Overall, by comparing these tests it can be seen that the data follows a mutual trend, which allows us to assume that the data is reliable and reproducible.

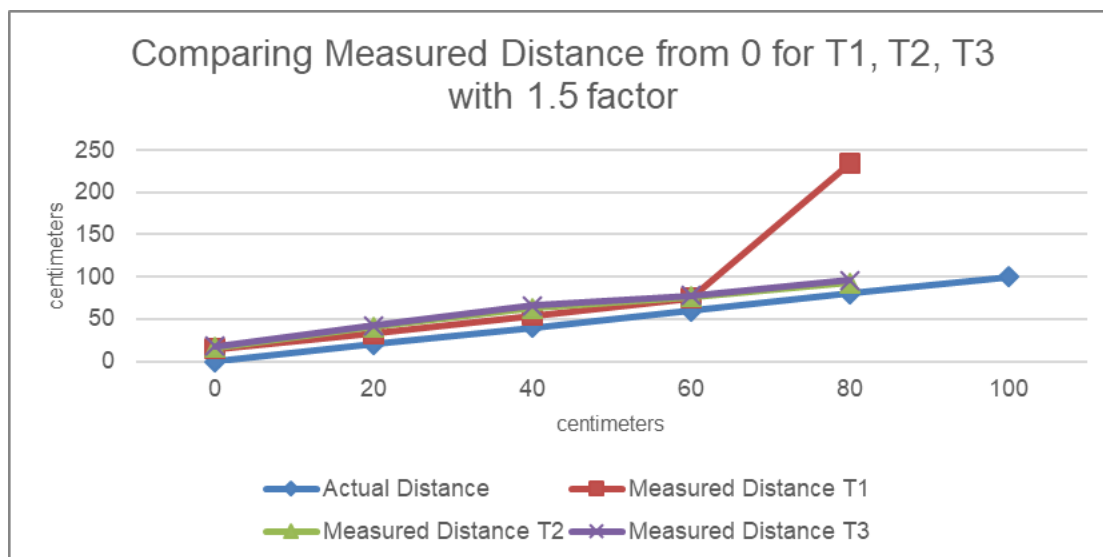


Figure 3.3.17: Tests 1, 2, 3 - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna

The measured distance T1 was very stable and similar to the other measured distance. However, it can be seen that T1 increases drastically after 60cm. Overall, by comparing these tests it can be seen that the data follows a mutual trend, which allows us to assume that the data is reliable and reproducible.

3.3.1.4 Test using Absorbers to limit multipath

The following two tests were performed for repeatability by the Brunel Team using the same system setup with an addition of EM absorbers to reduce any multipath.

Table 3.3.7: Test 1 with absorbers - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna

Test 1 (Brunel Team) Measured Distance Between Adjacent													
x (cm)	x (mm)	cm	Mean TX	Ratio of permittivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)	
0	0	0	2058	1.5	10	15	0	5	0	5	0.0%	50%	
10	100	10	2068		7	10.5	-3	0.5	3	0.5	15.0%	3%	
20	200	10	2075		2	3	-8	-7	8	7	26.7%	23%	
30	300	10	2077		11	16.5	1	6.5	1	6.5	2.5%	16%	
40	400	10	2088		13	19.5	3	9.5	3	9.5	6.0%	19%	
50	500	10	2101		5	7.5	-5	-2.5	5	2.5	8.3%	4%	
60	600	10	2106		-9	-13.5	-19	-23.5	19	23.5	27.1%	34%	
70	700	10	2097		14	21	4	11	4	11	5.0%	14%	
80	800	10	2111		18	27	8	17	8	17	8.9%	19%	
90	900	10	2129		7	10.5	-3	0.5	3	0.5	3.0%	1%	
100	1000	10	2136		1	1.5	-9	-8.5	9	8.5	8.2%	8%	
110	1100	10	2137		15	22.5	5	12.5	5	12.5	4.2%	10%	
120	1200	10	2152										
TOTAL:		120				94	141						
							Average		5.7	8.7			
							STDEV		4.8	6.5			

The measured data are presented in Table 3.3.7 above, which corresponds to the experimental setup 1. Based on the data presented in the table, the recorded distance

travelled was 120 centimetres, whereas the measured distance amounted to 94 cm, with increments of 10 centimetres. Based on the results obtained, it was determined that the mean score was 5.7. Nevertheless, upon considering the 1.5 correction factor, the score exhibited a notable increase to 8.7. Similarly, in accordance with the standard deviation, the initial value was recorded as 4.8, however, it subsequently exhibited an increase to 6.5.

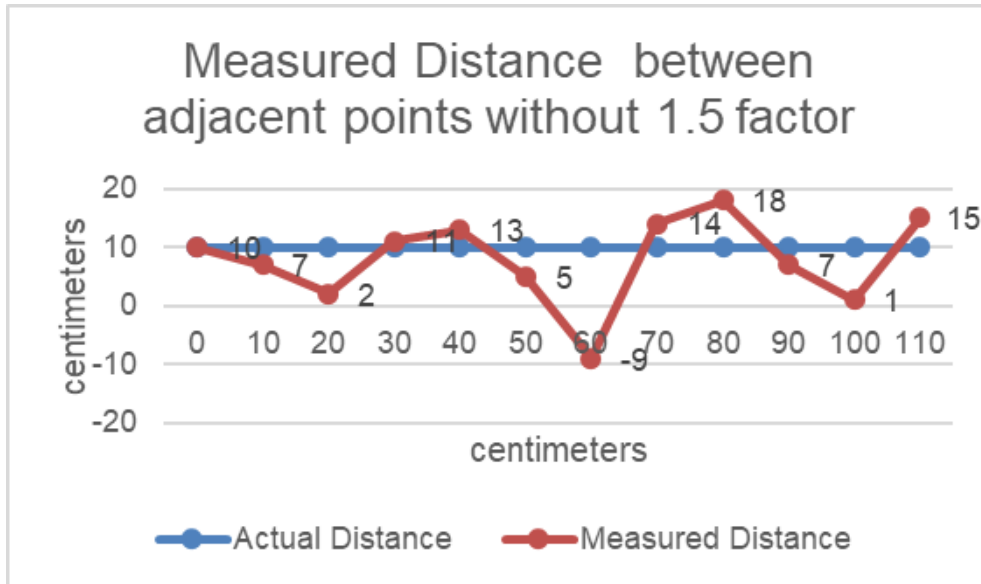


Figure 3.3.18: Test 1 with absorbers - Measured Distance between adjacent points without 1.5 correction factor for single UE transmit antenna

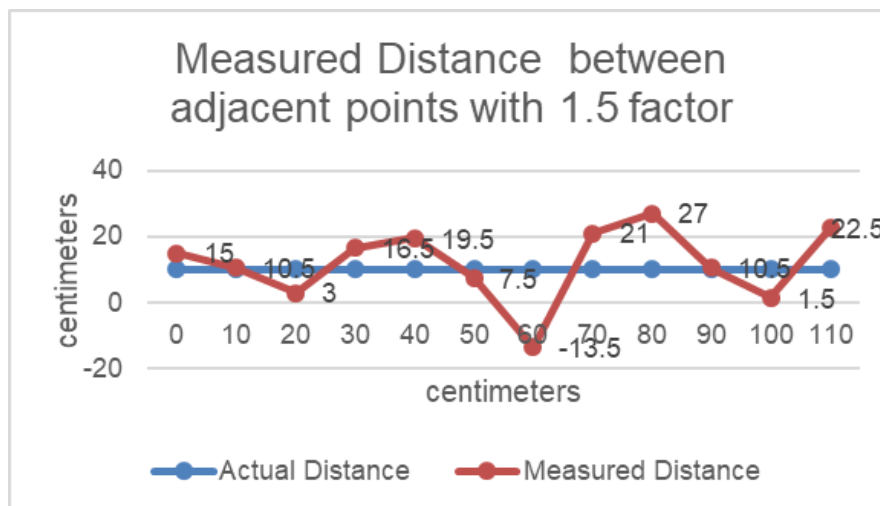


Figure 3.3.19: Test 1 with absorbers - Measured Distance between adjacent points with 1.5 correction factor for single UE transmit antenna

The diagram that is displayed above shows that the obtained results are of poor quality when the measuring distance between adjacent points is not adjusted by a correction factor of 1.5. On the other hand, when the 1.5 correction factor is taken into account, the results does not show significant improvement.

Table 3.3.8 - Test 1 with absorbers - Numerical Results of Measured Distance from zero for single UE Transmit antenna

Test 1 (Brunel Team) Measured Distance From Zero												
x (cm)	x (mm)	cm	Mean TX	Ratio of premitivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		2058	1.5								
10	100	10	2068		10	15	0	5	0	5	0%	50%
20	200	10	2075		17	25.5	-3	5.5	3	5.5	15%	28%
30	300	10	2077		19	28.5	-11	-1.5	11	1.5	37%	5%
40	400	10	2088		30	45	-10	5	10	5	25%	13%
50	500	10	2101		43	64.5	-7	14.5	7	14.5	14%	29%
60	600	10	2106		48	72	-12	12	12	12	20%	20%
70	700	10	2097		39	58.5	-31	-11.5	31	11.5	44%	16%
80	800	10	2111		53	79.5	-27	-0.5	27	0.5	34%	1%
90	900	10	2129		71	106.5	-19	16.5	19	16.5	21%	18%
100	1000	10	2136		78	117	-22	17	22	17	22%	17%
110	1100	10	2137		79	118.5	-31	8.5	31	8.5	28%	8%
120	1200	10	2152		94	141	-26	21	26	21	22%	18%
TOTAL:		120				581	871.5					
									Average	16.6	9.9	
									STDEV	10.4	6.3	

The measured data are presented in Table 3.3.8 above, which corresponds to the experimental setup 1. Based on the data presented in the table, the recorded distance travelled was 120 centimetres, whereas the measured distance amounted to 94 cm, with increments of 10 centimetres. Based on the results obtained, it was determined that the mean score was 16.6. Nevertheless, upon considering the 1.5 correction factor, the score exhibited a notable decrease to 9.9. Similarly, in accordance with the standard deviation, the initial value was recorded as 10.4, however, it subsequently exhibited an increase to 6.3.

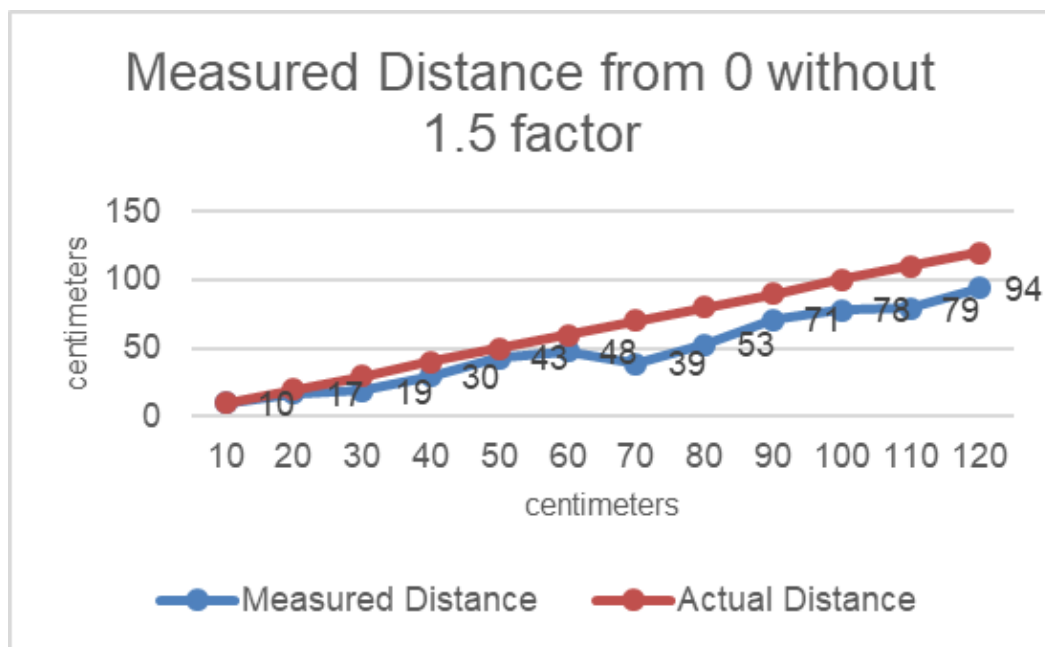


Figure 3.3.20: Test 1 with absorbers - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna

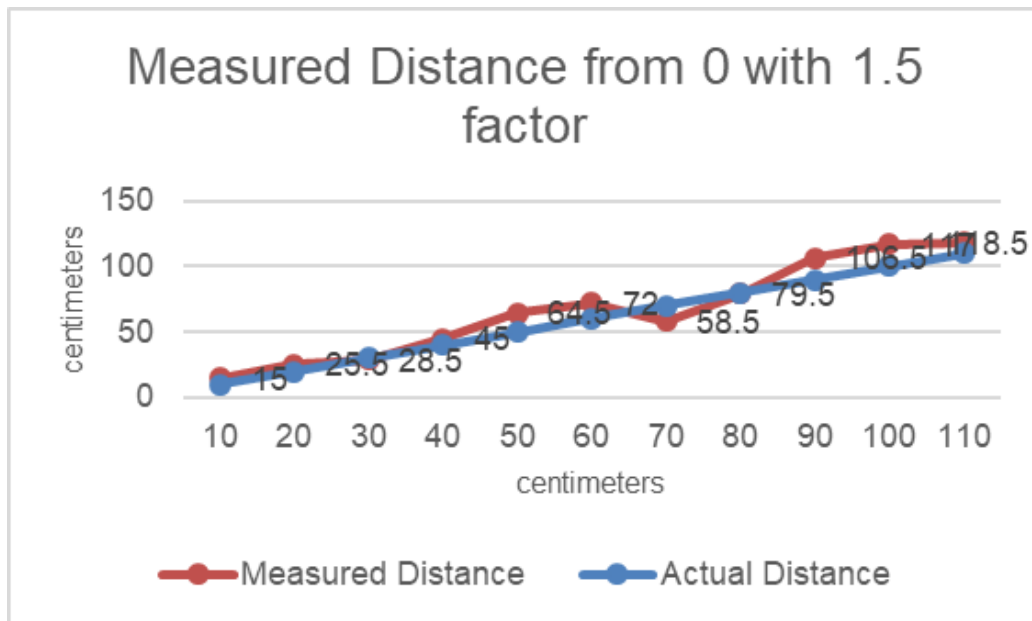


Figure 3.3.21: Test 1 with absorbers - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna

As depicted in the Figure 3.3.21 above, the measuring distance from 0 without the 1.5 correction factor yields poor results; consequently, when the 1.5 correction factor was taken into account, the results significantly improved.

3.3.2 Experiment 2: Separate UE Tx and Ref Tx antennas

System setup 2: EM Absorbers has been used throughout experiment 2

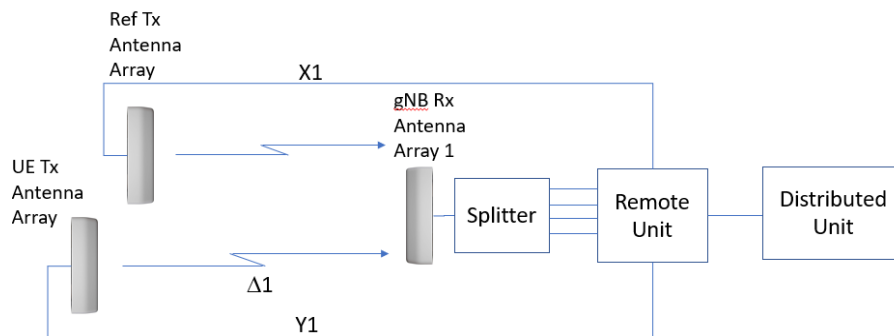


Figure 3.3.22: TDoA Experimental Setup for Location Measurement with single UE Tx Antenna Array and reference UE Tx Antenna Array



Figure 3.3.23: Experimental Set Up with EM absorbent material

The EM absorbent material used was:

RF EMI Absorbing Sheet ECCOSORB® QR-13AF Polyurethane Foam 24.000" (609.60mm) X 24.000" (609.60mm) X 0.250" (6.35mm)

[78135164 Laird Technologies EMI | RF and Wireless | DigiKey](https://www.digikey.com/en/products/detail/laird-technologies/78135164)

[RFP-DS-QR 13 AF 03082022.pdf \(laird.com\)](https://www.laird.com/Products/RF-EMI-Absorbing-Sheet-ECCOSORB-QR-13-AF-03082022.pdf)

Whose absorption characteristics were as shown in Figure 3.3.24 below

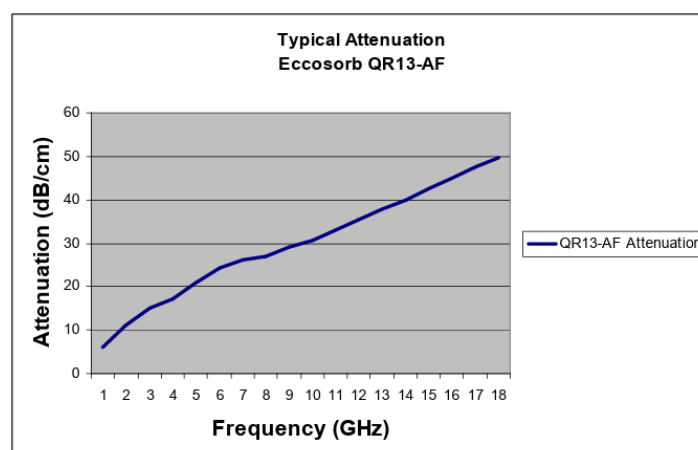


Figure 3.3.24: Typical attenuation of EM absorbent material

Table 3.3.9: Test 1 - Numerical Results of Measured Distance between Adjacent points using Ref Tx and UE Transmit antenna

Test 1 (Brunel Team) Measured Distance Between Adjacent														
x (cm)	x (mm)	cm	Mean TX	Reference TX mean	Rx Section	Ratio of permittivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)
0	0	0	1953	1742	122	1.5	7	10.5	-3	0.5	3	0.5	30.0%	5%
10	100	10	1960	1747	124		5	7.5	-5	-2.5	5	2.5	25.0%	13%
20	200	10	1965	1745	132		4	6	-6	-4	6	4	20.0%	13%
30	300	10	1969	1742	139		6	9	-4	-1	4	1	10.0%	3%
40	400	10	1975	1741	146		13	19.5	3	9.5	3	9.5	6.0%	19%
50	500	10	1988	1746	154		9	13.5	-1	3.5	1	3.5	1.7%	6%
60	600	10	1997	1747	161		1	1.5	-9	-8.5	9	8.5	12.9%	12%
70	700	10	1998	1745	163		7	10.5	-3	0.5	3	0.5	3.8%	1%
80	800	10	2005	1743	172		12	18	2	8	2	8	2.2%	9%
90	900	10	2017	1746	183		6	9	-4	-1	4	1	4.0%	1%
100	1000	10	2023	1749	186		7	10.5	-3	0.5	3	0.5	2.7%	0%
110	1100	10	2030	1750	193		14	21	4	11	4	11	3.3%	9%
120	1200	10	2044	1749	207									
TOTAL:		120					91	136.5						
										Average	3.9	4.2		
										STDEV	2.0	3.8		

The measured data are presented in Table 3.3.9 above, which corresponds to the experimental setup 1. Based on the data presented in the table, the recorded distance travelled was 120 centimetres, whereas the measured distance amounted to 136.5 cm, with increments of 10 centimetres. Based on the results obtained, it was determined that the mean score was 3.9. Nevertheless, upon considering the 1.5 correction factor, the mean increased to 4.2. Similarly, in accordance with the standard deviation, the initial value was recorded as 2, however, it subsequently exhibited an increase to 3.8.

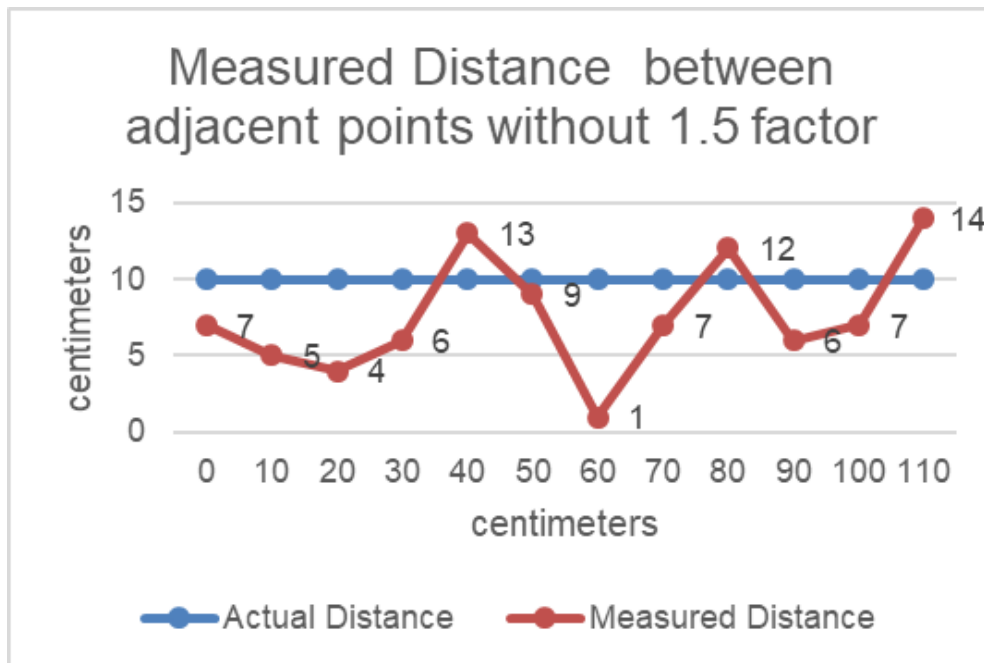


Figure 3.3.25: Test 1 - Measured Distance between adjacent points without 1.5 correction factor using Ref Tx and UE transmit antenna

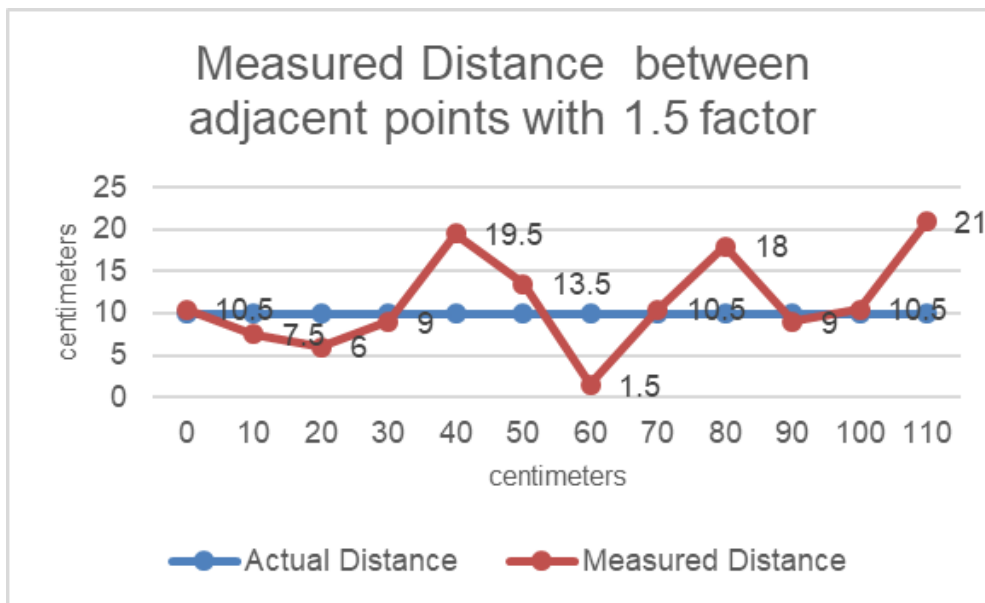


Figure 3.3.26: Test 1 - Measured Distance between adjacent points with 1.5 correction factor using Ref Tx and UE transmit antenna

The Figure 3.3.26 demonstrates that the measured distance between adjacent points is closer to the actual distance when the 1.5 correction factor is not considered; however, when the 1.5 correction factor is considered, the points are further from the actual distance.

Table 3.3.10: Test 1 - Numerical Results of Measured Distance from zero using Ref Tx and UE Transmit antenna

Test 1 (Brunel Team) Measured Distance From Zero														
x (cm)	x (mm)	cm	Mean TX	Reference TX mean	Rx Section	Ratio of premitivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		1953	1742	122	1.5								
10	100	10	1960	1747	124		7	10.5	-3	0.5	3	0.5	30%	5%
20	200	10	1965	1745	132		12	18	-8	-2	8	2	40%	10%
30	300	10	1969	1742	139		16	24	-14	-6	14	6	47%	20%
40	400	10	1975	1741	146		22	33	-18	-7	18	7	45%	18%
50	500	10	1988	1746	154		35	52.5	-15	2.5	15	2.5	30%	5%
60	600	10	1997	1747	161		44	66	-16	6	16	6	27%	10%
70	700	10	1998	1745	163		45	67.5	-25	-2.5	25	2.5	36%	4%
80	800	10	2005	1743	172		52	78	-28	-2	28	2	35%	3%
90	900	10	2017	1746	183		64	96	-26	6	26	6	29%	7%
100	1000	10	2023	1749	186		70	105	-30	5	30	5	30%	5%
110	1100	10	2030	1750	193		77	115.5	-33	5.5	33	5.5	30%	5%
120	1200	10	2044	1749	207		91	136.5	-29	16.5	29	16.5	24%	14%
TOTAL:		120					535	802.5						
										Average	20.4	5.1		
										STDEV	9.1	4.0		

The measured data are presented in Table 3.3.10 above, which corresponds to the experimental setup 1. Based on the data presented in the table, the recorded distance travelled was 120 centimetres, whereas the measured distance amounted to 91 cm, with increments of 10 centimetres. Based on the results obtained, it was determined that the mean was 20.4. Nevertheless, upon considering the 1.5 correction factor, the mean a notable decreased to 5,1. Similarly, in accordance with the standard deviation, the initial value was recorded as 9.1, however, it subsequently exhibited an increase to 4.

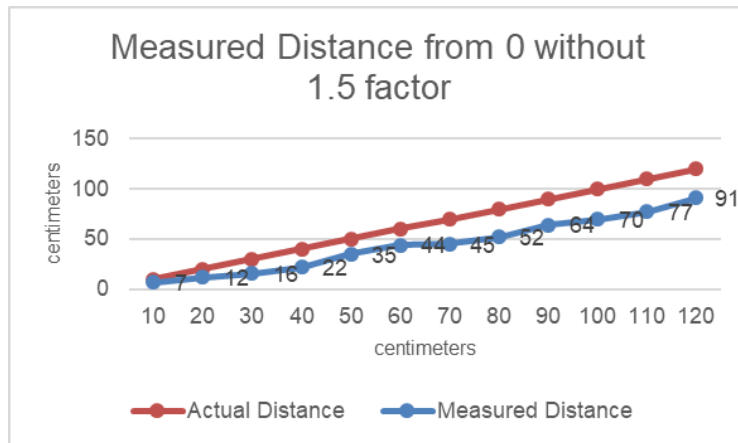


Figure 3.3.27: Test 1 - Measured Distance from zero without 1.5 correction factor using Ref Tx and UE transmit antenna

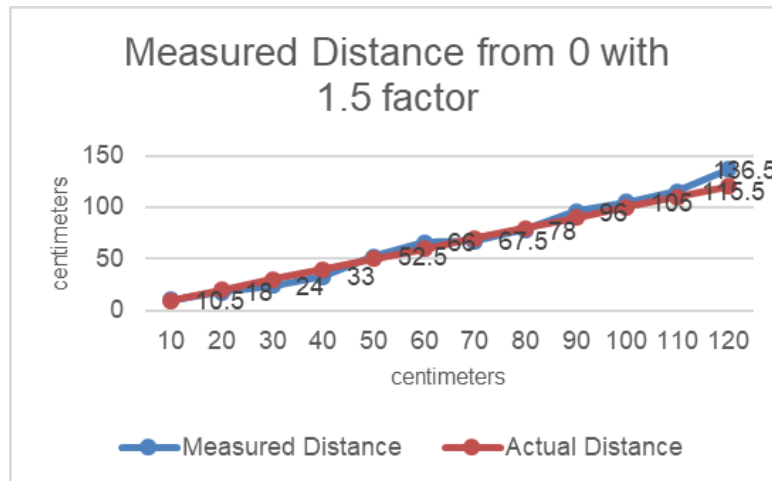


Figure 3.3.28: Test 1 - Measured Distance from zero with 1.5 correction factor using Ref Tx and UE transmit antenna

As illustrated in the aforementioned Figure 3.3.28, the measurement distance from the origin without considering the multiplication correction factor of 1.5 resulted in unsatisfactory outcomes. However, upon incorporating the 1.5 correction factor, a notable enhancement in the results was observed as it almost half the error for each distance travelled.

Table 3.3.11: Test 2 with absorbers - Numerical Results of Measured Distance between Adjacent points for single UE Transmit antenna

Test 2 (Brunel Team) Measured Distance Between Adjacent															
x (cm)	x (mm)	cm	Mean TX	Reference TX mean	Rx Section	Ratio of permittivity between air and OF (ROP)	Measured Distance between adjacent (D)	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from Adj with Abs(D)	% Error from Adj with Abs(ROP*D)	
0	0		1954	1744	122	1.5	6	9	-4	-1	4	1	40.0%	10%	
10	100	10	1960	1749	125		6	9	-4	-1	4	1	20.0%	5%	
20	200	10	1966	1746	132		4	6	-6	-4	6	4	20.0%	13%	
30	300	10	1970	1743	139		7	10.5	-3	0.5	3	0.5	7.5%	1%	
40	400	10	1977	1742	146		10	15	0	5	0	5	0.0%	10%	
50	500	10	1987	1746	153		8	12	-2	2	2	2	3.3%	3%	
60	600	10	1995	1747	160		1	1.5	-9	-8.5	9	8.5	12.9%	12%	
70	700	10	1996	1744	162		7	10.5	-3	0.5	3	0.5	3.8%	1%	
80	800	10	2003	1741	174		12	18	2	8	2	8	2.2%	9%	
90	900	10	2015	1745	182		7	10.5	-3	0.5	3	0.5	3.0%	1%	
100	1000	10	2022	1748	187		8	12	-2	2	2	2	1.8%	2%	
110	1100	10	2030	1749	192		14	21	4	11	4	11	3.3%	9%	
120	1200	10	2044	1748	207										
TOTAL:		120						90	135						
										Average	3.5	3.7			
										STDEV	2.2	3.5			

Table 3.3.11 (see above) displays the measured data based on the setup for a second attempt to determine if any improvements have been made and to see if the results are reproducible. According to the table, the measured distance was 90 cm and the travelled distance was 120cm in 10cm increments. The average mean was 3.5 and increased to 3.7 when the correction factor of 1.5 was considered. Similarly, the standard deviation was 2.2 and increased to 3.5.

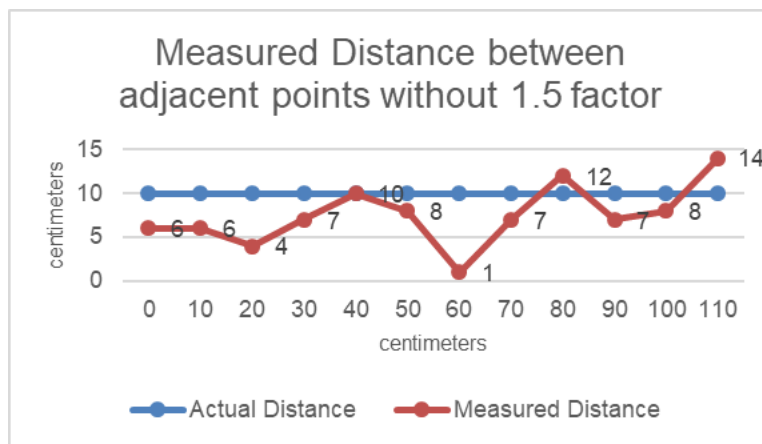


Figure 3.3.29: Test 2 with absorbers - Measured Distance between adjacent points without 1.5 correction factor for single UE transmit antenna

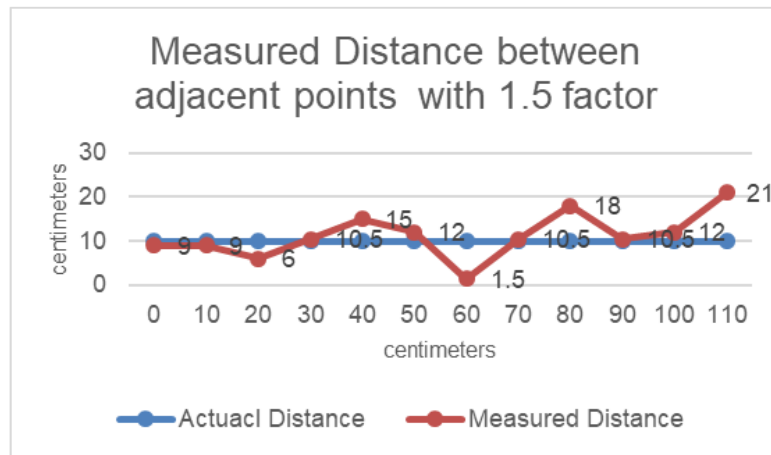


Figure 3.3.30: Test 2 with absorbers - Measured Distance between adjacent points with 1.5 correction factor for single UE transmit antenna

The Figure 3.3.30 above demonstrates that the measured distance between adjacent points is closer to the actual distance when the 1.5 correction factor is considered; however, when the 1.5 correction factor is not considered, the points are further from the actual distance.

Table 3.3.12: Test 2 with absorbers - Numerical Results of Measured Distance from zero for single UE Transmit antenna

Test 2 (Brunel Team) Measured Distance From Zero														
x (cm)	x (mm)	cm	Mean TX	Reference TX mean	Rx Section	Ratio of permittivity between air and OF (ROP)	measured distance from 0	ROP * D	ERROR With D	Error with ROP*D	Abs(ERROR) with D	Abs(ERROR) with ROP*D	% Error from 0 with Abs(D)	% Error from 0 with Abs(ROP*D)
0	0		1954	1744	122	1.5	7	10.5	-3	0.5	3	0.5	30%	5%
10	100	10	1960	1749	125		13	19.5	-7	-0.5	7	0.5	35%	3%
20	200	10	1966	1746	132		17	25.5	-13	-4.5	13	4.5	43%	15%
30	300	10	1970	1743	139		24	36	-16	-4	16	4	40%	10%
40	400	10	1977	1742	146		34	51	-16	1	16	1	32%	2%
50	500	10	1987	1746	153		42	63	-18	3	18	3	30%	5%
60	600	10	1995	1747	160		43	64.5	-27	-5.5	27	5.5	39%	8%
70	700	10	1996	1744	162		50	75	-30	-5	30	5	38%	6%
80	800	10	2003	1741	174		62	93	-28	3	28	3	31%	3%
90	900	10	2015	1745	182		69	103.5	-31	3.5	31	3.5	31%	4%
100	1000	10	2022	1748	187		77	115.5	-33	5.5	33	5.5	30%	5%
110	1100	10	2030	1749	192		91	136.5	-29	16.5	29	16.5	24%	14%
120	1200	10	2044	1748	207									
TOTAL:		120					529	793.5						
										Average	20.9	4.4		
										STDEV	9.6	4.0		

The subsequent Table 3.3.12 displays the measured data based on the setup for the second attempt to determine if any improvements have been made and if the results can be replicated. The table indicates that the measured distance was 91 cm and the travelled distance was 120 cm in 10 cm increments. The average mean was 20.9 and dropped to 4.4 when the correction factor of 1.5 was taken into account. Similarly, the standard deviation decreased from 9.6 to 4.

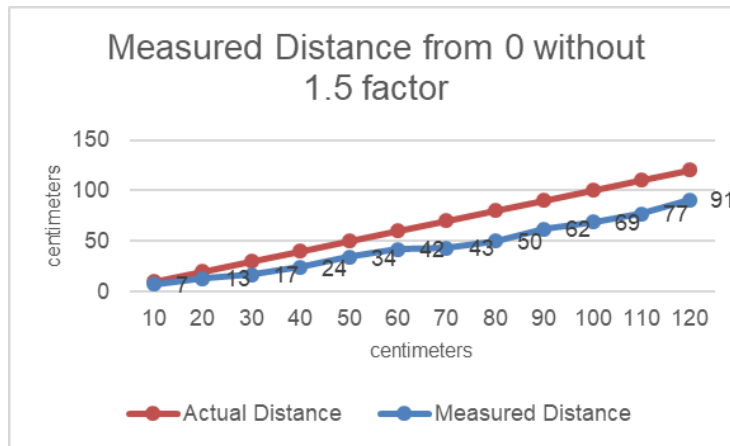


Figure 3.3.31: Test 2 with absorbers - Measured Distance from zero without 1.5 correction factor for single UE transmit antenna



Figure 3.3.32: Test 2 with absorbers - Measured Distance from zero with 1.5 correction factor for single UE transmit antenna

As illustrated in the aforementioned Figure 3.3.32, the measurement distance from the origin without considering the multiplication correction factor of 1.5 resulted in unsatisfactory outcomes. However, upon incorporating the 1.5 correction factor, a notable enhancement in the results was observed as it almost half the error for each distance traveled.

Comparing Measured Distance T1 and T2 together:

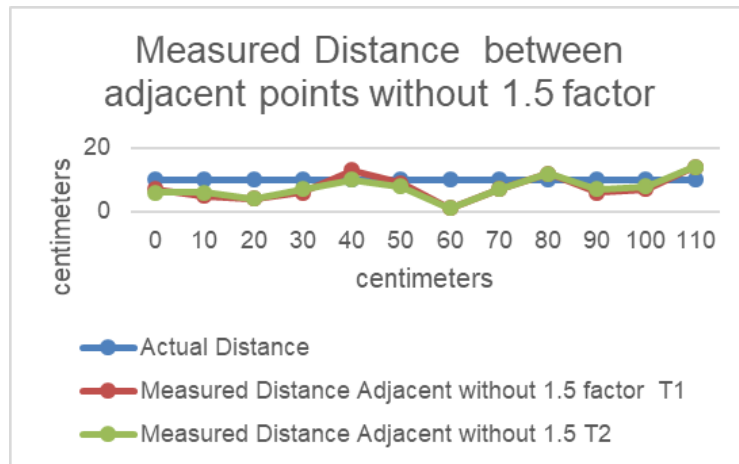


Figure 3.3.33: Test 1,2 - Measured Distance from adjacent points without 1.5 correction factor using Ref Tx and UE transmit antenna

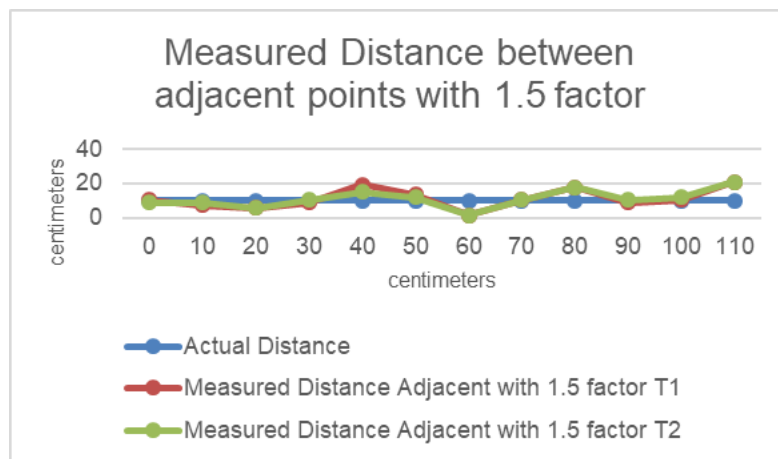


Figure 3.3.34: Test 1, 2 - Measured Distance from adjacent points with 1.5 correction factor using Ref Tx and UE transmit antenna

The Figure 3.3.33 and Figure 3.3.34 above show the measured distance between adjacent points without the 1.5 correction factor to be closer to the actual distance, however, considering the 1.5 correction factor, it can be seen that the points are further away from the actual distance. Nonetheless, it shows that the results are reproducible. Overall, comparing both set of results, it showed a slight improvement but mostly it was the same.

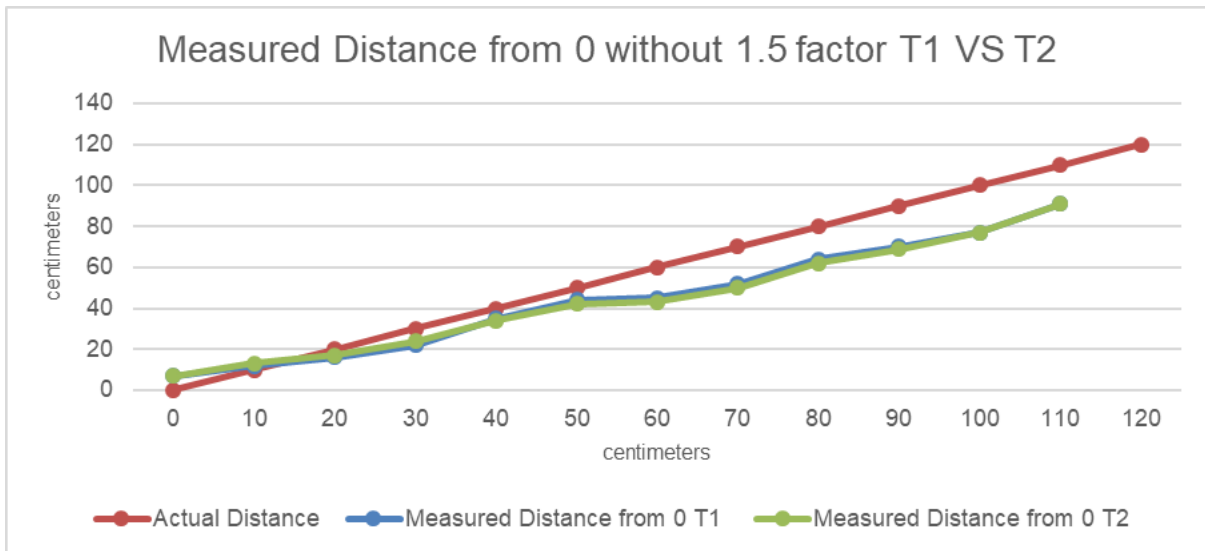


Figure 3.3.35: Test 1, 2 - Measured Distance from zero without 1.5 correction factor using Ref Tx and UE transmit antenna

Overall, comparing both set of results, it showed a slight improvement but mostly it was the same. However, considering the 1.5 correction factor always showed improvement when measuring the distance from 0.

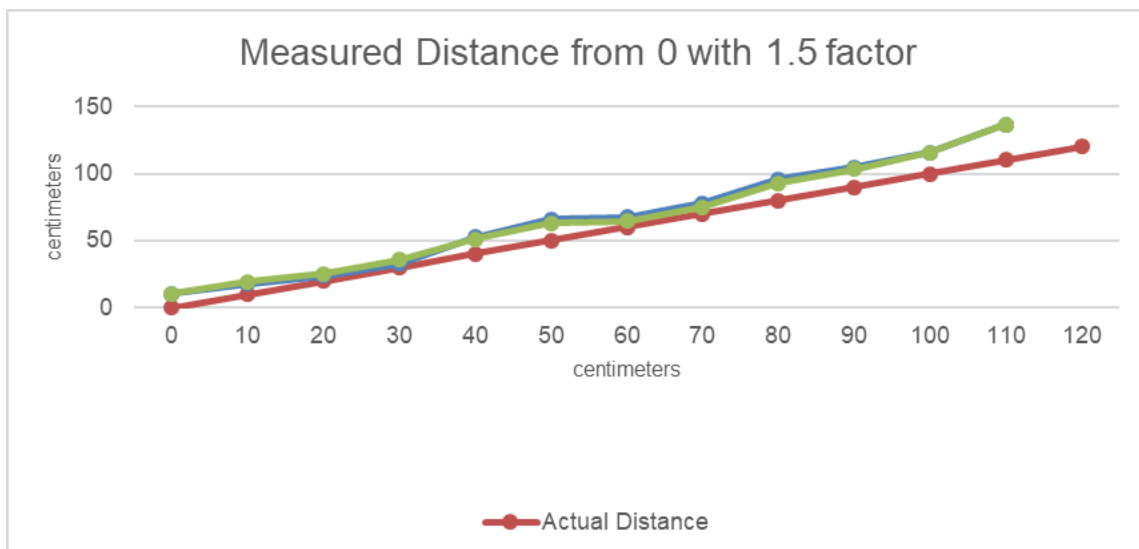


Figure 3.3.36: Test 1, 2, 3 - Measured Distance from zero with 1.5 correction factor using Ref Tx and UE transmit antenna

3.4 Detailed Measurement Data Results and Analysis

3.4.1 Capturing the Measurement Results

Measurement data is transmitted with UDP Packets using 5G functional application platform interface (FAPI) which is an initiative within the small cell industry released by Small Cell Forum (SCF) that establishes interoperability and innovation among suppliers of platform hardware, platform software and application software [3-3].

Table 3.4.1: 5G FAPI by SCF – Number and Type of messages

Type	Description
uint8_t	Number of messages included in PHY API message
uint8_t	An opaque handle (purpose not defined, however, usages may include a PHY ID, Carrier ID or Common Context) . Only P5 PARAM and CONFIG messages may be scoped at Common Context, in the current version of the specification.
Type	Description
uint16_t	Message type ID
uint32_t	Length of message body (bytes)
Message body	

In order to extract the data from the Ethernet frames the following need to be taken into account:

1. Endianness is big-endian
2. Slot numbers are between 0-19.
3. We have 16 fields for every number. These will all be in use in MIMO (4x4). The current test is SISO so only 1 value is in use.
4. We have 2 TX antenna, each one transmitting 4 sections: [24, 32, 40, 48], [64, 72, 80, 88]
5. We have 4 RX antenna each one receiving 2 sections: [24, 64], [32, 72], [40, 80], [48, 88];
6. Every packet holds information from 1 section. Therefore, we will have 8 packets of the same type in each slot.
7. Every packet begins with a general FAPI PHY API message word (64 bits):

The relevant Message type ID values for ToA measurements are:

- 7.1. CRC Indication = 0x0086 = 0d134
- 7.2. TOA Packet (Runel's proprietary) = 0x0008

Table 3.4.2: 5G FAPI by SCF – Data Fields

Field	Type	Description
SFN	uint16_t	SFN Value: 0 -> 1023
Slot	uint16_t	Slot Value: 0 ->159
NumCRCs	uint16_t	Number of CRCs (PDUs) with status indication included in this message. Range 0-> MaxULPDUsPerSlot
For each CRC {		
Handle	uint32_t	The handle passed to the PHY in an UL_TTI.request PUSCH PDU or MsgA-PUSCH PDU.
RNTI	uint16_t	The RNTI passed to the PHY in an UL_TTI.request PUSCH PDU or MsgA-PUSCH PDU. If Handle corresponds to a MsgA-PUSCH message, this field indicates the RA-RNTI associated with the received PDU, otherwise it is set to 255. Value: 1 → 65535
RAPID	uint8_t	If Handle corresponds to a MsgA-PUSCH message, this field indicates the RAPID associated with the received PDU, otherwise it is set to 255. Value 0 -> 63, or 255
HarqID	uint8_t	HARQ process ID Value: 0->15

Field	Type	Description
TbCrcStatus	uint8_t	Indicates CRC result on TB data. Value: 0 = pass 1 = fail
NumCb	uint16_t	If CB CRC status is not reported, this parameter is set to zero. Otherwise the number of CBs requested in PUSCH PDU. Note: in case of initial transmission or for non CBG Re-Tx is not used, this is the total number of CBs, otherwise (CBG Re-Tx) this is the number of CBs actually scheduled. Value: 0->65535
CbCrcStatus	uint8_t[ceil(NumCb/8)]	If NumCb=0 this field is not present. Otherwise each bit indicates CRC result on CB data. Value: 0 = pass 1 = fail Note: for the special case where the CW is composed of a single CB, PHY may indicate NumCb=1 with CbCrcStatus set to the TB CRC status
UISinrMetric	int16_t	A metric of channel quality. Up to PHY implementation whether this is Signal-to-Thermal, Signal-to-Interference+Thermal or any other reasonable equivalent interpretation. Value: - 65.534 dB ... +65.534 dB in steps of 0.002 dB (0 corresponds to 0 dB). 0xFFFF = -32768 = invalid.
Timing advance offset	uint16_t	Timing advance T_A measured for the UE in multiples of $16 * 64 * T_c / 2^{\mu}$ [3GPP TS 38.213 [4], Section 4.2] Value: 0 → 63 0xffff should be set if this field is invalid
Timing advance offset in nanoseconds	int16_t	Timing advance measured for the UE between the reference uplink time and the observed arrival time for the UE

Field	Type	Description
		Value: - 16800 ... +16800 nanoseconds. 0xffff should be set if this field is invalid
RSSI	uint16_t	RSSI. See Table 3-22 for RSSI definition. If RSSI is reported in dBFS (see PARAM and CONFIG measurement TLVs) then this value represents - 128dBFS to 0dBFS with a step size of 0.1dB If RSSI is reported in dBm (see PARAM and CONFIG measurement TLVs) then this value represents - 128dBm to 0dB, with a step size of 0.1dB Value: 0-1280 0xffff should be set if this field is invalid
RSRP	uint16_t	RSRP. See Table 3-22 for RSRP definition. If RSRP is reported in dBFS (see PARAM and CONFIG measurement TLVs) then this value represents - 128dBFS to 0dBFS with a step size of 0.1dB If RSRP is reported in dBm (see PARAM and CONFIG measurement TLVs) then this value represents - 140dBm to -12dBm, with a step size of 0.1dB Value: 0-1280 0xffff should be set if this field is invalid
}		

The frames work in pairs as shown below.

MessageID 134 in Ethernet Frame 228 produces for Slot 8, Handle 48, RNTI 220 (in red below) produces UISinrMetric: 6336, TimingAdvanceOffset: 0, TimingAdvanceOffsetNs: 205, RSSI: 423, RSRP: 423 (in blue below)

Ethernet Frame: 228

- Destination: 90:E2:BA:D9:2F:CD, Source: 00:0A:35:02:00:22, Protocol 8
- IPv4 Packet:
 - Version: 4, Header Length: 20, TTL: 64,
 - Protocol: 17, Source: 10.1.0.140, Target: 10.1.0.104
- UDP Segment:
 - Source Port: 1234, Destination Port: 1236, Length: 48
 - FAPI PHY Data:

- No of Messages: 1, Undefined: 0, MessageID: 134, length: 28
- CRC Segment
 - SFN: 518, Slot: 8, numCRC: 1
 - Handle: 48, RNTI: 220, RAPID: 255
 - HarqID: 0, TbCrcStatus: 1
 - Numcb: 1, CbCrcStatus: 1
 - UISnrMetric: 6336, TimingAdvanceOffset: 0
 - TimingAdvanceOffsetNs: 205, RSI: 423, RSRP: 423

MessageID 8 in Ethernet frame 229 produces for Slot 8 sectionID: 48, RNTI: 220 (in red below)
Time1: 205220 (in blue below)

Ethernet Frame: 229

- Destination: 90:E2:BA:D9:2F:CD, Source: 00:0A:35:02:00:22, Protocol 8
- IPv4 Packet:
 - Version: 4, Header Length: 20, TTL: 64,
 - Protocol: 17, Source: 10.1.0.140, Target: 10.1.0.104
- UDP Segment:
 - Source Port: 1234, Destination Port: 1236, Length: 152
 - FAPI PHY Data:
 - No of Messages: 0, Undefined: 0, MessageID: 8, length: 0
 - TOA Data
 - slot: 8, sectionID: 48, RNTI: 220
 - Time1: 205220, SNR1: 0, RSI1: 0

3.4.2 Capturing the Measurement Results

Since there are 2 Tx Antenna, 4 RX Antenna. The 2 Tx antenna each one has 4 sections with their corresponding results.

- [24,32,40,48]
- [64,72,80,88]

Detailed results of data collected using TDoA experimental setup in Figure 3.2.6 were extracted for distance measurement 10 cm = 0.1m and analysed in more depth.

$X1 = Y1 = 6.0\text{m}$, $X2 = 4.0\text{ m}$, $X3 = 0.5\text{m}$

Measurement Δ was at 10 cm + 58 cm offset = 68cm = 0.68 m

If it was calibrated in coaxial cable then

$20.4340\text{m} - (6.0*1.5) - (4.0*1.5) - (0.5*1.5) = 20.4340\text{m} - 15.75\text{m} = 4.6840\text{m}$

$18.3036\text{m} - (6.0*1.5) - (4.0*1.5) - (0.5*1.5) = 18.3036\text{m} - 15.75\text{m} = 2.5536\text{m}$

95% of all samples are $204340 + \text{or} - 172.45 \times 2 = 344.9$ time-tics = 3.449 cm from the mean, which is approximately the degree of accuracy obtained from the measurement campaign reported in section 3.3. On the time series Figure 3.4.2 spikes occur at slot 6, 43, 78, 115, 150, 187, 223 etc., i.e. between 35 to 37 slots apart. If it is due to interference from the flat-bed plotter stepper motor or the cooling fan motor then it is a problem because there will be

plenty of these type of motors in factories, whereas if it due to the FPGA system clock then it is also a problem. In both cases the design of EM shielding is required.

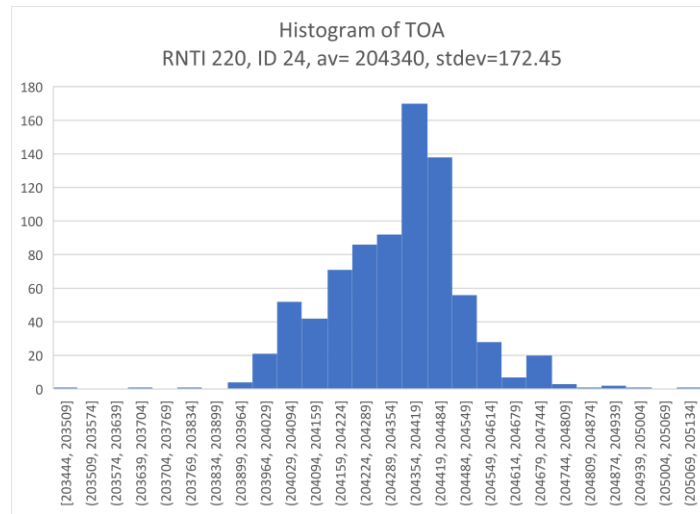


Figure 3.4.1: Frequency Distribution of measurements for RNTI=220, SECT=24

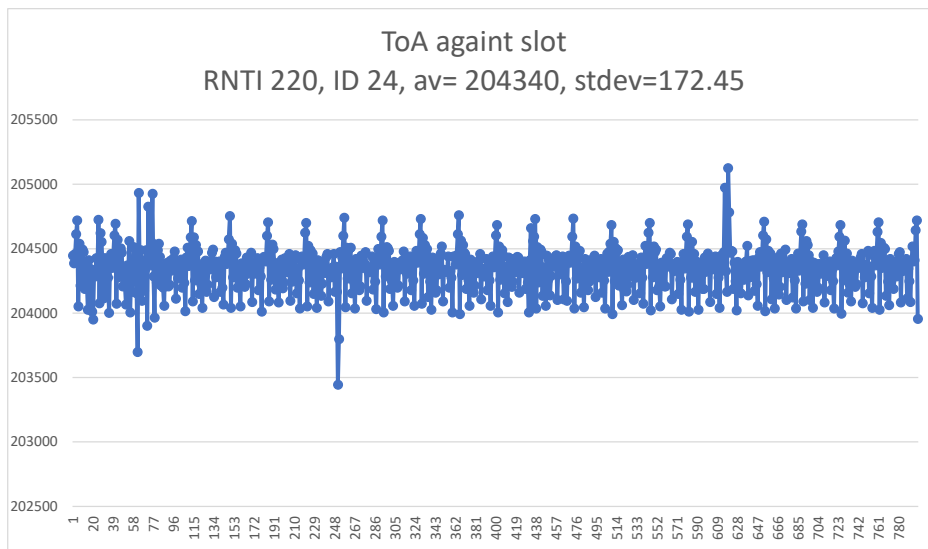


Figure 3.4.2: Time series of measurement data for RNTI=220, SECT=24

95% of all samples are $204428 \pm 176.52 \times 2 = 353.04$ time-tics = 3.5304 cm from the mean

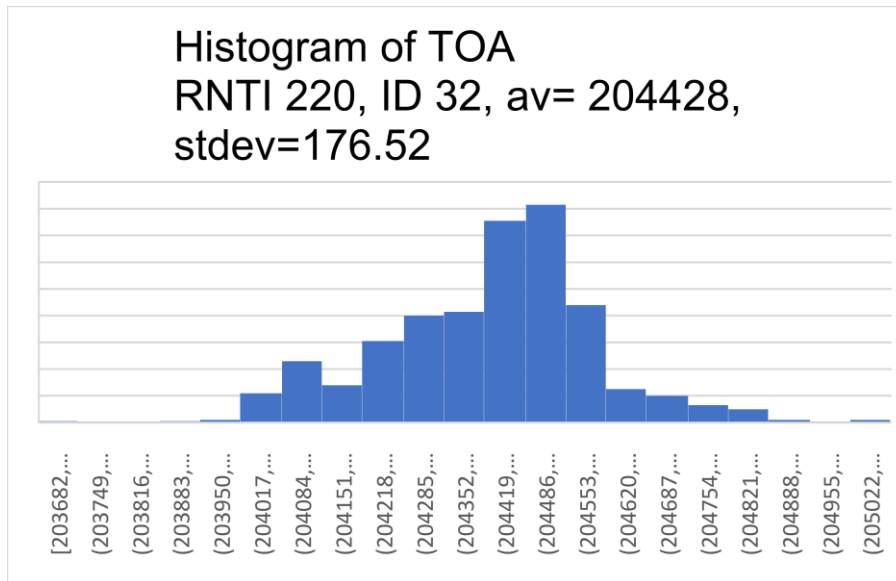


Figure 3.4.3: Frequency Distribution of measurements for RNTI=220, SECT=32

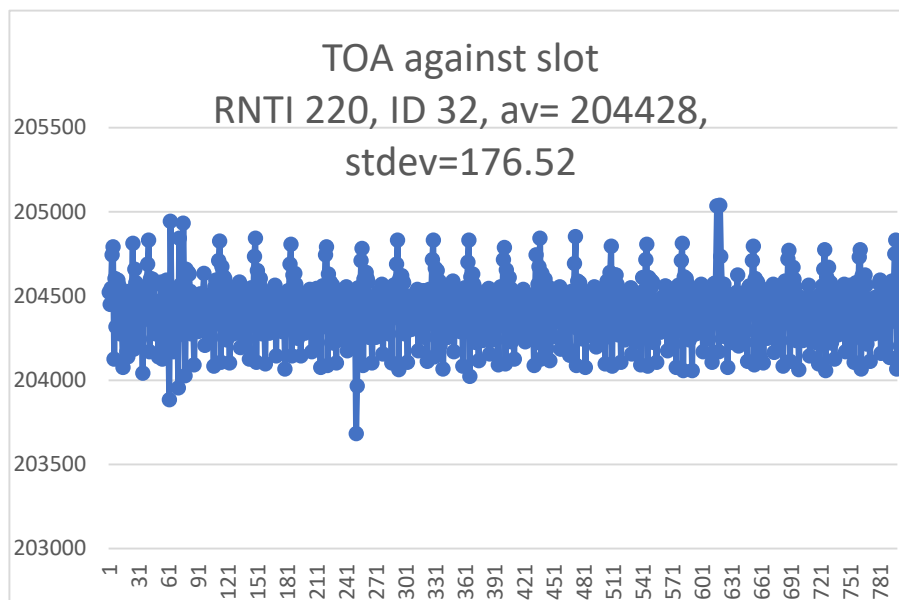


Figure 3.4.4: Time series of measurement data for RNTI=220, SECT=32

95% of all samples are $203291 + \text{or} - 175.78 \times 2 = 351.56$ time-tics = 3.5156 cm from the mean

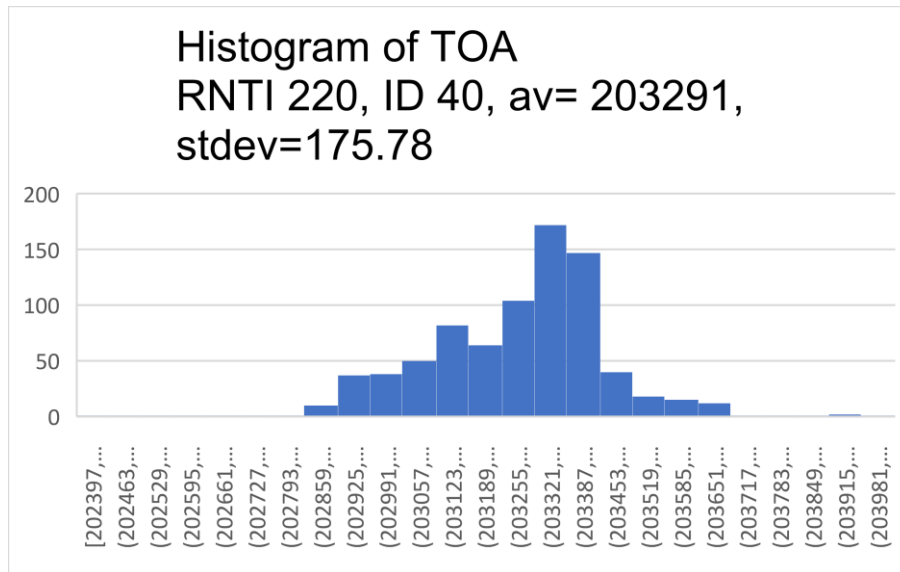


Figure 3.4.5: Frequency Distribution of measurements for RNTI=220, SECT=40

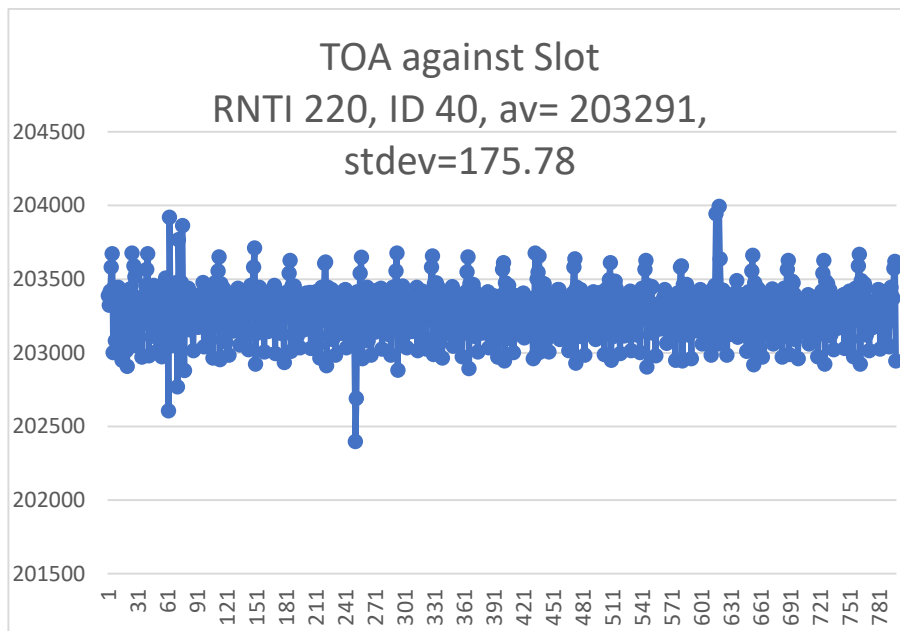


Figure 3.4.6: Time series of measurement data for RNTI=220, SECT=40

95% of all samples are $204657 + \text{or} - 159.15 \times 2 = 353.04$ time-tics = 3.183 cm from the mean

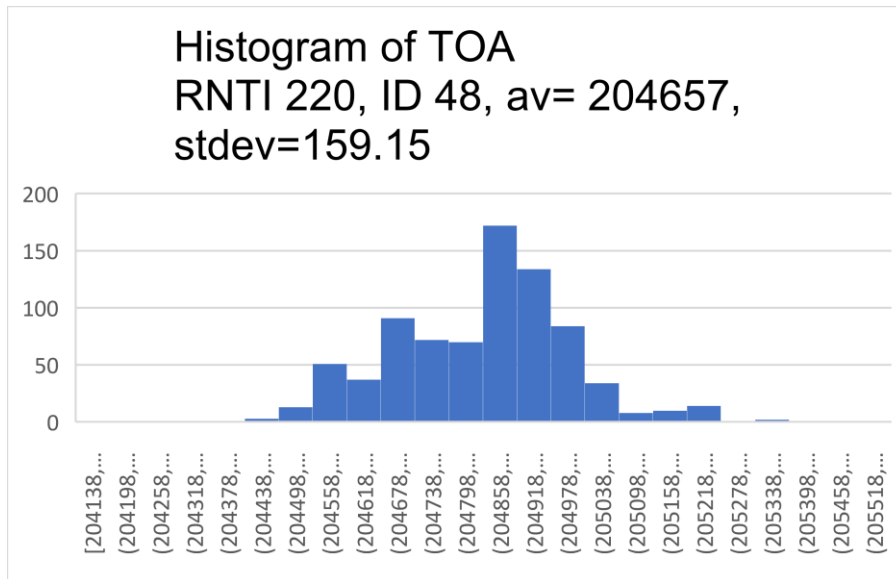


Figure 3.4.7: Frequency Distribution of measurements for RNTI=220, SECT=48

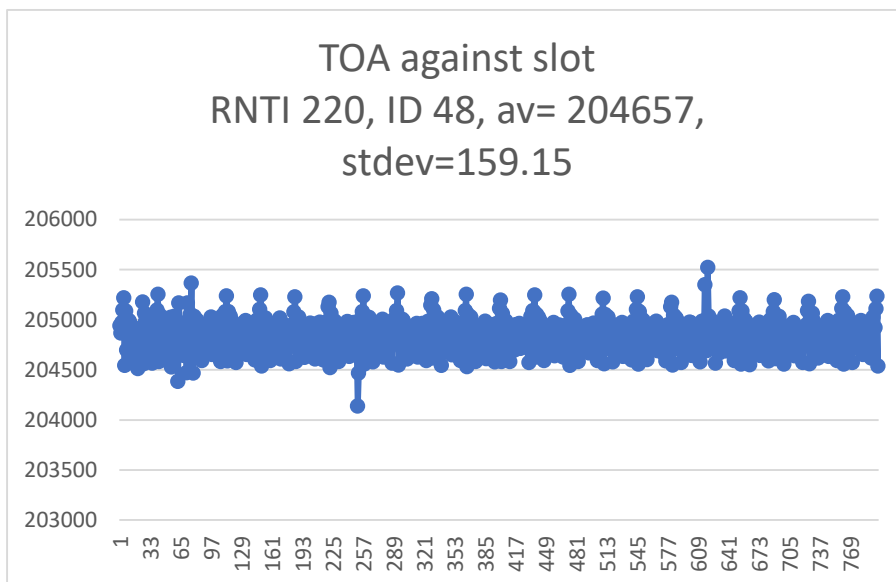


Figure 3.4.8: Time series of measurement data for RNTI=220, SECT=48

95% of all samples are $182979 + \text{or} - 218.06 \times 2 = 436.12$ time-tics = 4.3612 cm from the mean

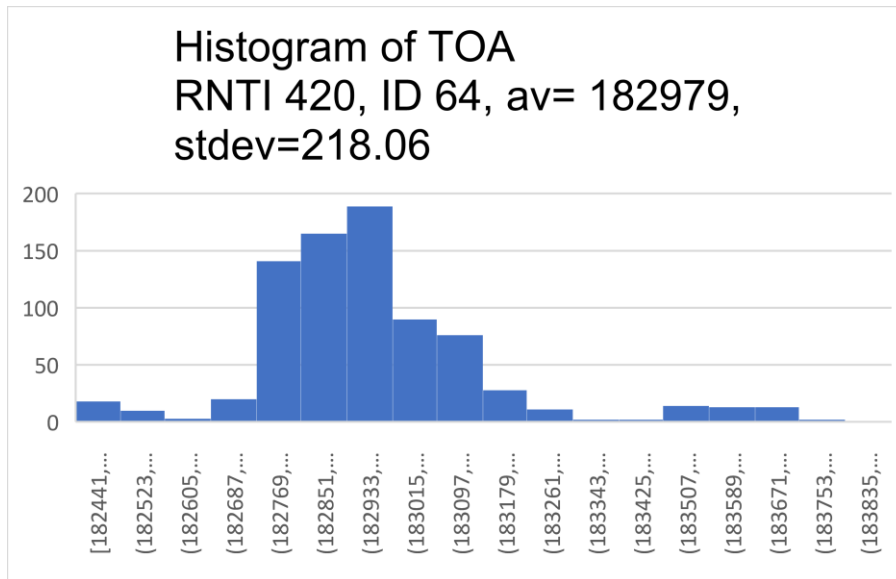


Figure 3.4.9: Frequency Distribution of measurements for RNTI=220, SECT=64

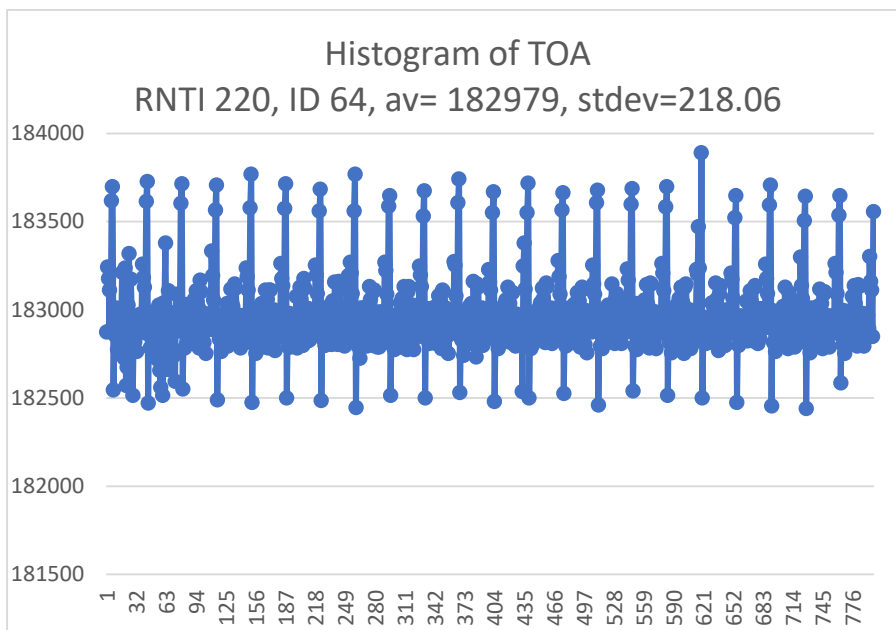


Figure 3.4.10: Time series of measurement data for RNTI=220, SECT=64

95% of all samples are $183036 \pm 222.31 \times 2 = 444.62$ time-tics = 4.4462 cm from the mean

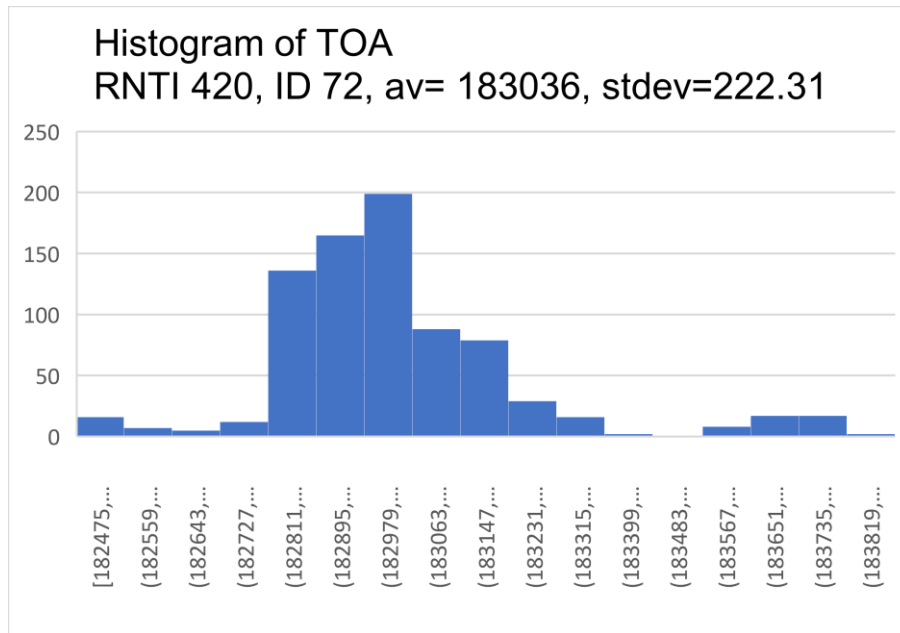


Figure 3.4.11: Frequency Distribution of measurements for RNTI=220, SECT=72

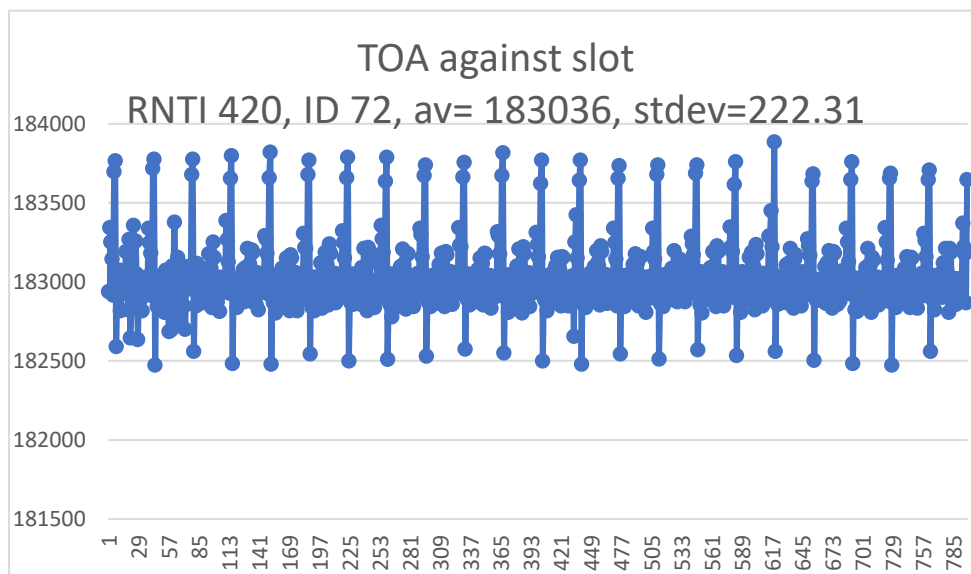


Figure 3.4.12: Time series of measurement data for RNTI=220, SECT=72

95% of all samples are $181995 + \text{or} - 225.03 \times 2 = 450.06$ time-tics = 4.5006 cm from the mean

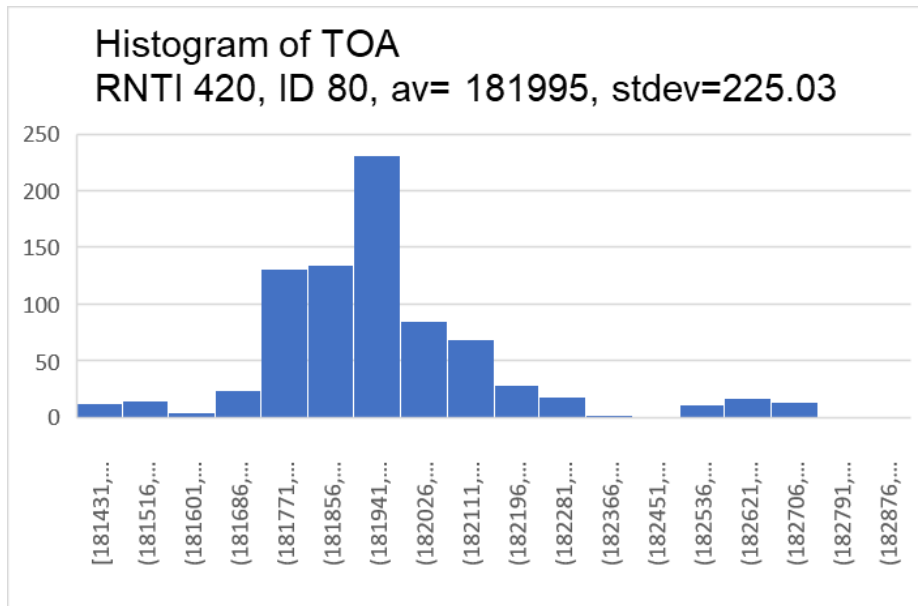


Figure 3.4.13: Frequency Distribution of measurements for RNTI=220, SECT=80

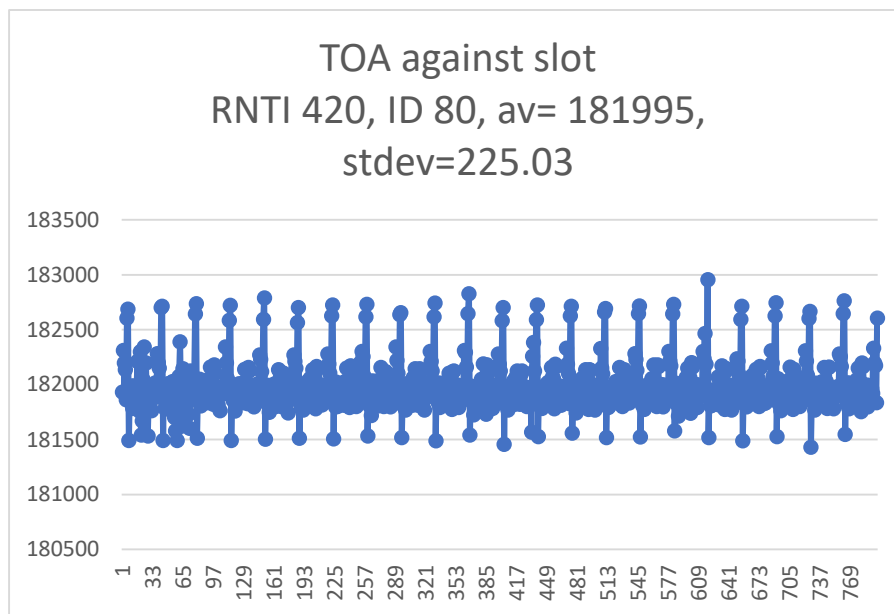


Figure 3.4.14: Time series of measurement data for RNTI=220, SECT=80

95% of all samples are $183558 \pm 212.18 \times 2 = 424.36$ time-tics = 4.2436 cm from the mean

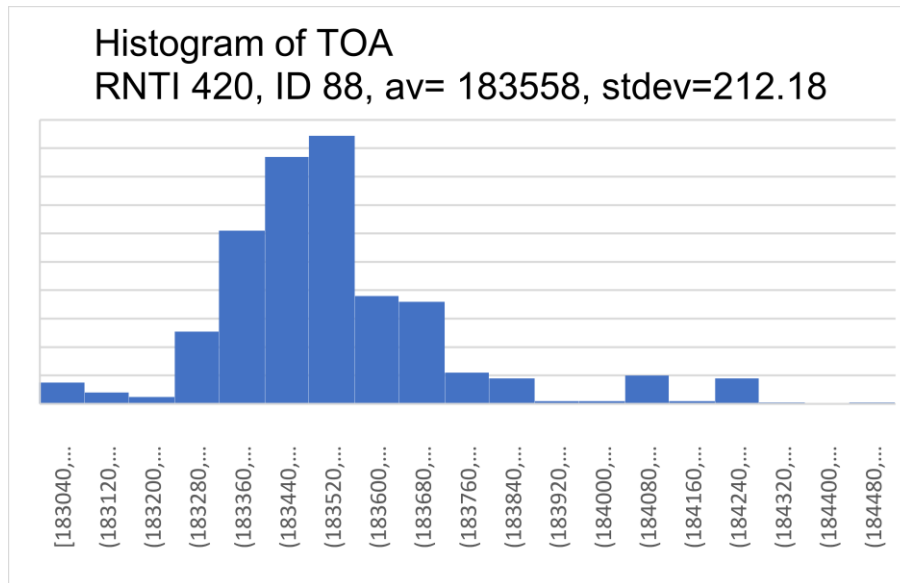


Figure 3.4.15: Frequency Distribution of measurements for RNTI=220, SECT=88

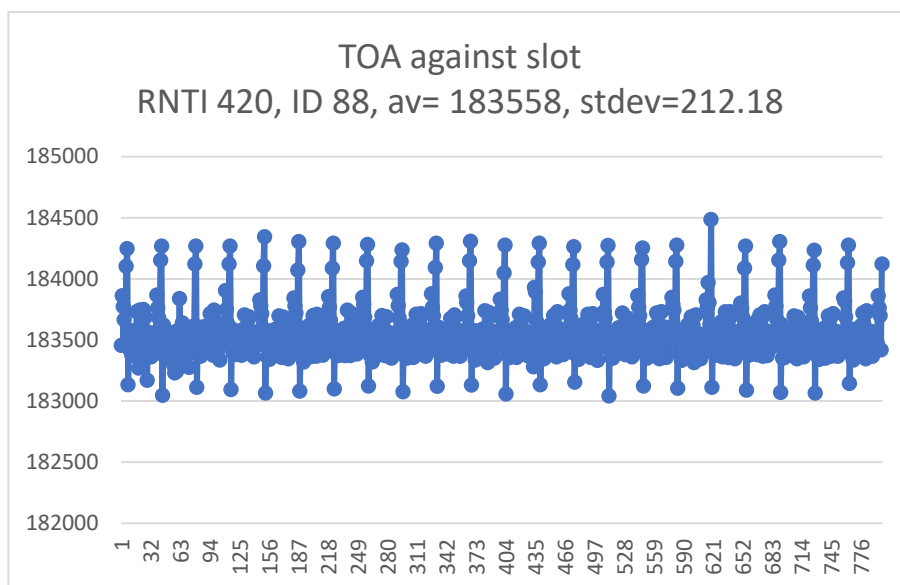


Figure 3.4.16: Time series of measurement data for RNTI=220, SECT=24

A summary of all the measurement mean, stdev and 95% CI at 68 cm are shown in Table 3.4.3. It shows that the 95% CI for the four sections 24, 32, 40, 48 is around 3.5cm and for the four sections 64, 72, 80, 88 is around 4.5cm, which provides a good indicator of the accuracy of the system.

Table 3.4.3: Summary of Measurements of mean, stdev and 95% CI at 68cm.

ID	mean (mm)	StdDev (mm)	95% CI (cm)
24	20434.0	17.245	3.449
32	20442.8	17.652	3.530
40	20329.1	17.578	3.516
48	20465.7	15.915	3.183
64	18297.9	21.806	4.361
72	18303.6	22.231	4.44
80	18199.5	22.503	4.500
88	18355.8	21.218	4.244

4 Environment Imaging and Sensing

4.1 Aim of Experiment

Within the modern-day setting, using LIDAR technologies to measure the state of an external environment is common, however with the emergence of next-generation automation within factory and large-scale warehouse environments [4-1] there is a desperate need to adapt. In a factory space, where there are a multitude of automated 'bots' conducting various tasks, there is the issue of keeping track of each bot's positioning. Localisation plays a key role in this process of keeping track of positions, and there have been multiple endeavours to find the most optimal solution to this.

The aim of this experiment is to design and develop a system where a potential device equipped with a LIDAR can be located within an environment. This environment contains multiple shapes which will be utilised as landmarks. The LIDAR then recognizes these landmarks and determines the distance from itself and the landmark. When a number of measurements have been recorded, an algorithm estimates its position within the environment. This project scope is strictly within a two-dimensional space, and as such will not determine the height of the device. The objectives are as follows:

The system should read sensor data produced by the LIDAR

The system must be able to detect landmarks through this sensor data

An algorithm must be made to retrieve distances from these landmarks

Provided that the distances have been determined, the algorithm should be able to determine the estimated position of the sensor in the environment. The motivation for this work is to determine what level of location accuracy could be obtained from a 6G Communication and Sensing system using location from landmarks

This work was performed as research informed teaching BEng student final year project by Prabhveer Mujral [4-32] under the guidance and supervision of 6G BRAINS researchers Ali Mahbas and John Cosmas.

4.2 Background Theory

4.2.1 Applications of localisation

The importance of localisation, or position estimation cannot be underestimated, as it can be applied to a multitude of applications, such as ranging from emergency systems, where having accurate location of an emergency caller is paramount [4-2] to more commercial applications, such as providing location-based services [4-3]. In more modern applications such as autonomous driving, intelligent transportation systems (ITS), and vehicular ad-hoc networks (VANETS), where Simultaneous Localisation and Mapping (SLAM) is heavily implemented [4-4], requires a more responsive and accurate position tracking in order to fully function. In addition to this, with the emergence of the Internet of Things (IoT) [4-5] and tactile internet [4-6], the application of localisation has never been more visible to the everyday consumer. Not one solution is applicable to all cases, as multiple factors can impact their efficacy.

4.2.2 Localisation Techniques

Localisation consists of determining the coordinates and direction of an object within a map. This could be interchangeable with positioning, which only determines the coordinates of an object. One of the most common methods of localisation is through satellite navigation. This generally works out a user equipment's (UE) position by estimating the intersections between two or more satellites. Although this technology is effective in outdoor environments, it is affected by multiple sources of errors, potentially causing deviations of up to 100m. If the satellite clock has an inaccuracy of one nanosecond, it results in 30cm of error. Moreover, the speeds of GPS signals are affected by both the ionosphere and troposphere, causing further deviations of up to 30m. This also requires a clear line of sight (LOS) [4-7].

From a mobile networks point of view, positioning techniques can be broken into two categories, depending on the entity which computes the position – mobile based and network based [4-2]. Mobile based is where the UE itself works out its own location whereas in network based, the network location server would calculate a UE's position. The latter is also considered to be the standard due to the control the network operator is able to have, as well as support for legacy devices. The fundamental techniques these mobile networks use is trilateration, triangulation (akin to GPS), proximity, and scene analysis.

In a general sense, mobile location estimation is possible using Received Signal Strength (RSS), Time Difference of Arrival (TDoA), and Angle of Arrival (AoA) of signals, but all suffer from the issue of LOS. In essence, to adopt these methods for localisation, multiple base stations (BS) with known locations must be utilised [4-8] where the previously mentioned geometric algorithms can be used to work out a UE's position. This issue is bettered in 5G and subsequently mmWave (30-100GHz), where machine learning models such as the data-centric fingerprint positioning [4-9] are implemented. Although machine learning models are far more effective at determining indoor position, it will require some form of offline training before putting into practice, often with extremely large datasets. In addition to this, the model would have to be retrained for each new environment.

With the emergency of 6th Generation mobile technology, a potential candidate is terahertz-band localisation [4-10]. By operating in the 0.1-10 THz range, its wavelengths range from 0.03-3mm, ultimately resulting in larger bandwidths and data rates. Although these frequencies bring a high degree of path loss and decreasing propagation distances, highly desirable localisation measurements can be derived by using high-gain antennae. Due to the infancy of this field, there are no set commercial applications however, advances in addressing system structures, and localisation algorithm research such as [4-11] suggest that the potential is present.

Another method of localisation that has been explored is through the use of sensor fusion. In this case multiple sensors are combined together to compute a more accurate result. In one report, ultra-wide band (UWB) frequency sensors were fused with vision sensors and inertial measurement units (IMUs) in an indoor environment in an attempt to create a more accurate localisation effort [4-12]. Their testing environment primarily utilised the trilateration result from the distances to three UWB beacons to work out the raw positions, which was then passed to an extended Kalman filter (EKF). This filter then combines the velocity and angle measurements from the IMUs in addition to the average position found by the vision sensor. Overall, this method produces more consistent results at a consistently high error delta.

4.2.3 LIDAR technology & localisation using LIDAR

LIDAR stands for Light Detection and Ranging and is a type of sensing technology which is heavily used in applications such as: autonomous vehicles, robotics, geographical surveys, and much more. The LIDAR emits laser pulses which bounce off objects and return back. The lasers time of arrival is then analysed to create a 3D map of the environment. The output of a LIDAR is in the form of a point cloud – a collection of millions of data points, with each point belonging to a cartesian coordinate. These sensors come in two types: airborne and terrestrial. Airborne is where a LIDAR is fixed onto an aircraft or helicopter. Terrestrial LIDAR comes in the form of mobile and static configurations. In mobile configurations, LIDARs are placed on moving entities such as cars, and are continuously capturing point clouds. Static LIDARs tend to be more portable and can be used in creating detailed point clouds of rooms such as a factory floor.

In a practical sense, there have been multiple ventures in utilising LIDAR technology in localisation. Firstly, localisation can be defined into two scenarios; when the robot is in a known environment, or when it is in a completely unknown environment. In situations in a known environment, robots can perform landmark-based localisation by detecting the distances from these markers. By parsing these values through an EKF, accurate pose data can be derived [4-13]. In completely unknown environments, in the case of LIDAR, techniques such as Simultaneous Localisation and Mapping (SLAM) are utilised. In LIDAR SLAM, a 3D LIDAR (outputs data in x, y, and z planes unlike a 2D LIDAR) takes multiple scans of an environment and determines an ego's movement by calculating the iterative closest point (ICP) differences between point clouds. By employing frameworks such as LIDAR odometry and mapping (LeGO-LOAM), highly accurate localisation results were produced [4-14]. Although these results were produced within a simulated environment, they are backed up by outdoor experiments [4-15].

LIDAR has previously been used as an assistance tool to reduce the localisation error in UWB in indoor robot localisation [4-16], rather than being the sole technology. This particular test combined a single scanning LIDAR to scan the environment alongside a UWB positioning element. The LIDAR aids localisation by measuring the distances from each UWB reference node at a particular time in the environment and is then compared against the ranges determined by the UWB. The difference between the two is considered to be the UWB error and is combined with another EKF to find the standard error. By knowing the error, adjustments can be made to position.

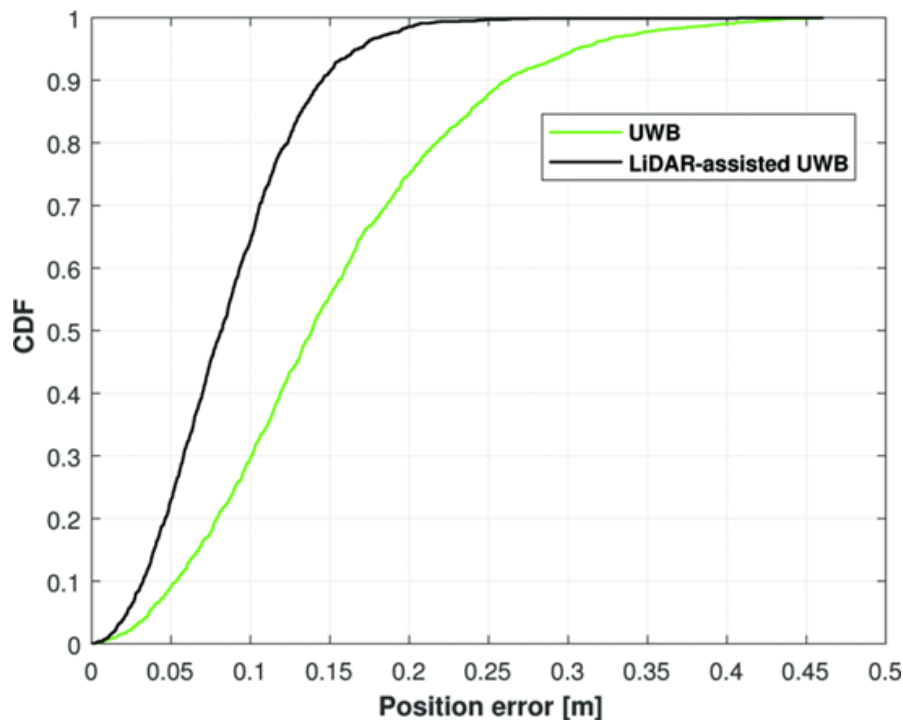


Figure 4.2.1: The CDFs of the position errors of the UWB and LIDAR-assisted UWB [4-16]

Their testing methodology consisted of a robot equipped with a LIDAR and a UWB positioning element which would be placed in a small-scale environment. Figure 4.2.1 shows that the cumulative distribution function (CDF) is smaller when utilising LIDAR, thus showing the LIDAR has improved position accuracy.

Another instance where LIDAR technology has been combined with another sensor for localisation, is with IMU sensors [4-17]. The issue identified in this report was regarding the loss in localisation accuracy, when a moving bot travelled a set distance. In this case, the IMU was the Bosch BNO055 which contained an accelerometer, a magnetometer, and gyroscopic sensors. By connecting these sensor outputs to an extension of the EKF, a more refined localisation package was created. The indoor experiment results showed that combining IMUs with a LIDAR produced localisation accuracy by 57.9-72.2%.

The key takeaway from these projects is that combining sensors with a LIDAR produces far more accurate results than on its own.

Another paper presents a framework of vehicle self-localisation within urban environments [4-18]. The idea is to classify distinct, physical objects such as light poles and stop signs as unique landmarks. Multiple segmentation techniques were applied to the LIDAR's point cloud to remove the ground floor and leave certain objects depending on the desired characteristics of the landmarks they wished to identify. Next, a type of rule-based filtering is completed on the point cloud to remove any noisy or poorly detailed objects. In this paper, they filter for cylindrical shapes, so they filtered out objects which had a larger width than height.

The stereo camera in this paper is an attempt at finding distance using its depth however, this is far less accurate than the LIDAR. Once filtered, a cascading classifier, which is a type of object detection algorithm, was run on the stereo camera to find any common traffic shapes, such as stop signs. Once detected and classified, the LIDAR's distance data is matched with the stereo camera's object detection information, by projecting the point cloud on the stereo images, to obtain a set of 2D feature points, as illustrated in Figure 4.2.2.

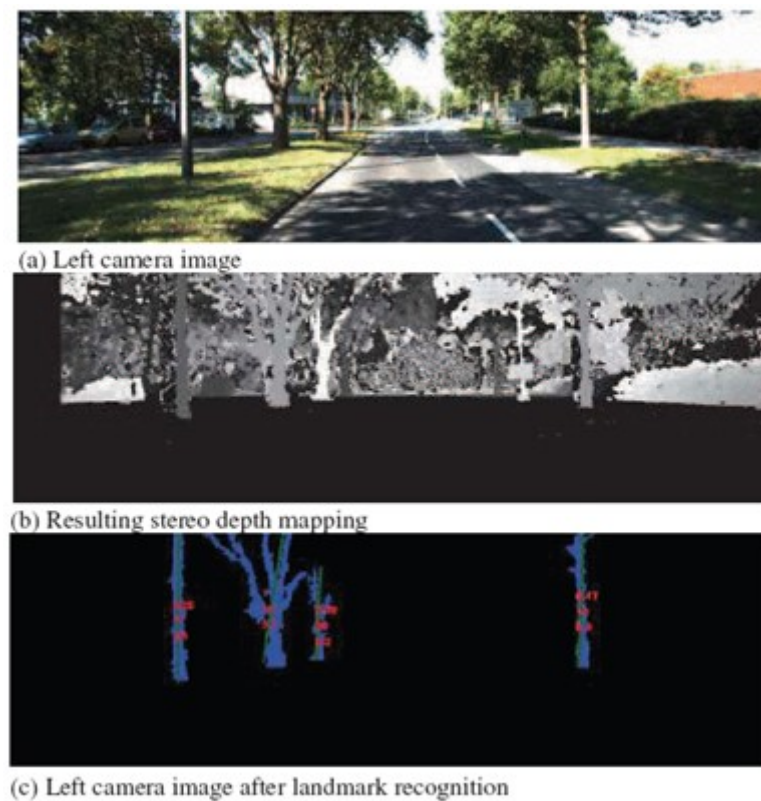


Figure 4.2.2: Illustration of the steps for the landmark detection using a stereo camera [4-18]

In terms of the results in this method, the paper tested multiple cases; only the LIDAR, just the stereo camera, and both LIDAR and camera. They ran simulations in all cases and concluded that using both sensors resulted in a position error below 0.5m. When comparing just LIDAR and camera, it was seen that the stereo camera was negligible in finding actual localisation. A major caveat is that the landmark's positions are already known in the environment, which is required to conduct the pose estimation. There is also the possibility that no landmarks would be identified.

Another paper proposed a solution of localisation with a single camera and LIDAR [4-19]. It expresses the already-mentioned disadvantages of using GPS (or lack thereof) within indoor environments and proposes an algorithm where a single moving camera takes two images. Mathematical representations of features are extracted from both images in the form of vectors and are subsequently projected onto the LIDAR's point cloud. Depth information is then found using sparse depth maps on the two images and 3D points are found on the feature pairs in both images. They then conduct a transformation matrix to find the pose. Overall, they were able to achieve an average error of less than 10cm. More importantly it did not require any prior knowledge of the environment but is reliant on both the camera and LIDAR being available. Furthermore, it could suffer if the environment is static, and is feature-less.

The majority of LIDAR-based solutions utilise SLAM as seen in [4-20]. It notes that a major strength of using LIDAR is that it is able to work in light-agnostic environments, as it can produce its own light source. This paper also presents three strategies for implementing SLAM: Gaussian filter-based, particle filter-based, and graph optimisation-based. Gaussian based assumes that the estimated state of the moving bot can be found using a probabilistic Gaussian filter. Particle filters estimate the position using a set of particles. These particles

would have individual weighting based off a number of factors. The particle with the best weighting would most likely be the following position of the bot. Finally, graph based aims to estimate position at all positions, both previous and future, by taking in all measurements and control values in a time range. It then makes a global optimisation for all poses.

A key takeaway from this paper is that the LOAM algorithm (also mentioned in [4-14]) produces localisation errors of less than 10cm in low and high-speed cases, and that a potential combination of visual and LIDAR features could increase the robustness of SLAM algorithms. SLAM does have its benefits over other methods such as its ability to conduct both system mapping and localisation but is computationally heavy.

4.2.4 Position Estimation Algorithms

To estimate position or localisation, there are a multitude of different algorithms depending on the use case. Algorithms such as the previously mentioned trilateration algorithm, could be utilised in both two- and three-dimensional environments, and appears to be the simplest to implement. This algorithm does have its drawbacks however, with one such problem being the inaccuracies in the distances measured to the landmarks. Efforts have been made to rectify this issue [4-21] by using EKFs to reduce the perceived noise in these measurements however, other issues still pose a problem such as losing LOS to the landmarks.

Another popular algorithm is fingerprinting. In this algorithm RSS values are stored in a database akin to a map [4-22]. When a UE measured their RSS, it is then matched to the radio map, and an estimated position is found. This method has a major problem - the radio map must be created which consists of measuring the RSS value of every access point (AP) in the system at every coordinate (or cell) in the map. In addition to the time taken to create this map exponentially increasing as the environment increases in size, but it will also have to be re-created should an AP be changed or shutdown.

4.2.5 Object Detection Algorithms

For this project, rather than powered landmarks such as APs, static inexpensive landmarks will be utilised. Therefore, the system must be able to identify these landmarks. One proposed method to recognise these landmarks is using object detection algorithms. Almost all object detection algorithms can be split into two methods: deep learning methods such as convolutional neural networks (CNNs), or traditional image processing techniques. Image processing such as python's OpenCV does not require any historic data for training but require near-perfect conditions to detect objects. Whereas deep learning (DL) methods usually require a form of supervised learning, in the form of manually labelling objects within a dataset, and then training the CNN over hundreds of epochs [4-23]. The benefit to DL algorithms is that they are far more robust to sub optimal conditions, thus more reliable if trained correctly.

One specific DL method is YOLO [4-24]. This form of object detection relies on the use of bounding boxes and classes. In essence, a trainer would provide a dataset of images that are pre-labelled. The objects that one wants to detect are initially given a class name and are labelled with a box around each instance in the image. One key benefit to utilising YOLO over other algorithms is that it is extremely fast in real-time detection. Other leading detection methods such as Fast R-CNN [4-25] may produce higher accuracy but struggle when run under new environments in addition to it being far slower.

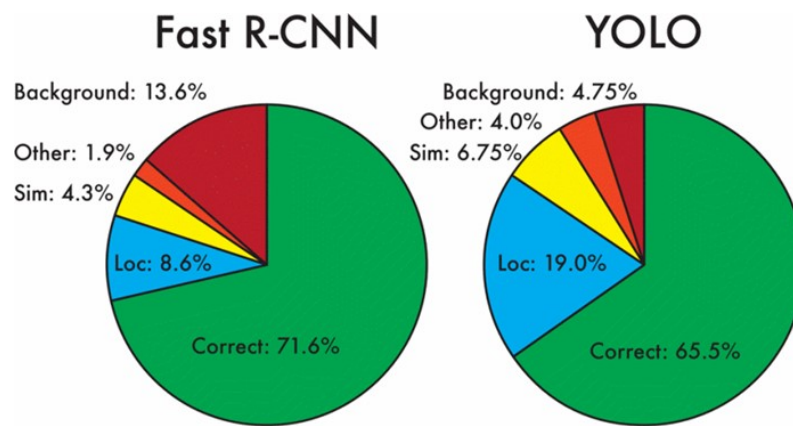


Figure 4.2.3: Source of errors in both Fast R-CNN and YOLO Detection [4-24]

Figure 4.2.3 above is a comparison between the two object detection methods, running on the same dataset looking for 20 classes. Fast R-CNN is slightly more accurate than YOLOv1, at the expense of a large increase in background errors. On the other hand, YOLO struggles at correctly localisation objects. It is possible however, to combine both detection methods by running and checking for any overlapping bounding boxes.

Overall, YOLO and its subsequent versions appear to be the optimal solution for this particular use case, mainly due to its real-time speed.

4.3 Experimental Setup

4.3.1 Introduction

This section will begin by identifying the system’s requirements and performance indicators. Then, the system environment will be deployed, which consists of creating and placing landmarks. Next, the processes behind connecting to the LIDAR, and 360-degree camera will be discussed. Furthermore, the training of the object detector will be discussed, in addition to the application of this detector. Finally, the algorithm behind taking distances, and applying them to a pose estimation algorithm will be described. This estimation algorithm will also be mapped onto a digital map, where the estimated position will be displayed within the 2D environment.

4.3.2 System Scope and Requirements

The system’s functional and technical requirements have been outlined in the Table 4.3.1 below.

Table 4.3.1: Proposed Functional and Technical Requirements

Proposed Functional Requirements	Proposed Technical Requirements
The LIDAR system should correctly detect and map the surrounding environment.	The LIDAR should be correctly configured in the system such that the testing environment is completely in view.
The system should automatically identify landmarks within the environment.	There must be a form of object detection which can identify objects within the environment.
The system must be able to accurately determine distance from landmarks.	Distance must be calculated from within the program, and as such must be able to identify the midpoint of detected landmarks.

Proposed Functional Requirements	Proposed Technical Requirements
The system must estimate position in real-time.	The program must be capable of receiving data from the object detector and LIDAR in real-time and estimate position.
The position estimation must work in the event a number of landmarks are blocked.	The object detector must be able to identify even partially blocked landmarks and must be able to determine position as long as the minimum number of landmarks are detected.
The landmarks should be standalone, and not require any continuous setup.	The landmarks should be non-electronic and should not require constant maintenance.

As the requirements have been identified, the KPIs must be outlined so that the system can be evaluated. The KPIs are as follows in Table 4.3.2:

Table 4.3.2: Proposed Key Performance Indicators

Key Performance Indicator
Distance measurements should be within 5% of actual distance.
Object detection should occur at least once per frame.
The object detector should have a false positive rate of less than 5%.
Distance measurements should occur at least once per frame.
Position estimation should be within 5% error in both x and y coordinates.

Based off these requirements, the proposed system can be outlined. The UE in this experiment will be equipped with a 360-degree LIDAR, which will continuously scan the environment. Localisation will be achieved by deploying multiple landmarks around an environment. There will be six landmarks deployed around environment, and the LIDAR will be able to identify them with the help of the object detection algorithm, YOLOv5. When the UE has detected at least three landmarks, another algorithm will measure the distance between the landmark and UE, which will then be parsed into a trilateration algorithm. This trilateration process requires the known positions of these landmarks and as such these will be known beforehand. The program will estimate the position of the UE in the environment and plot this coordinate on a 2D map. This system is outlined in the flowchart in Figure 4.3.1 below.

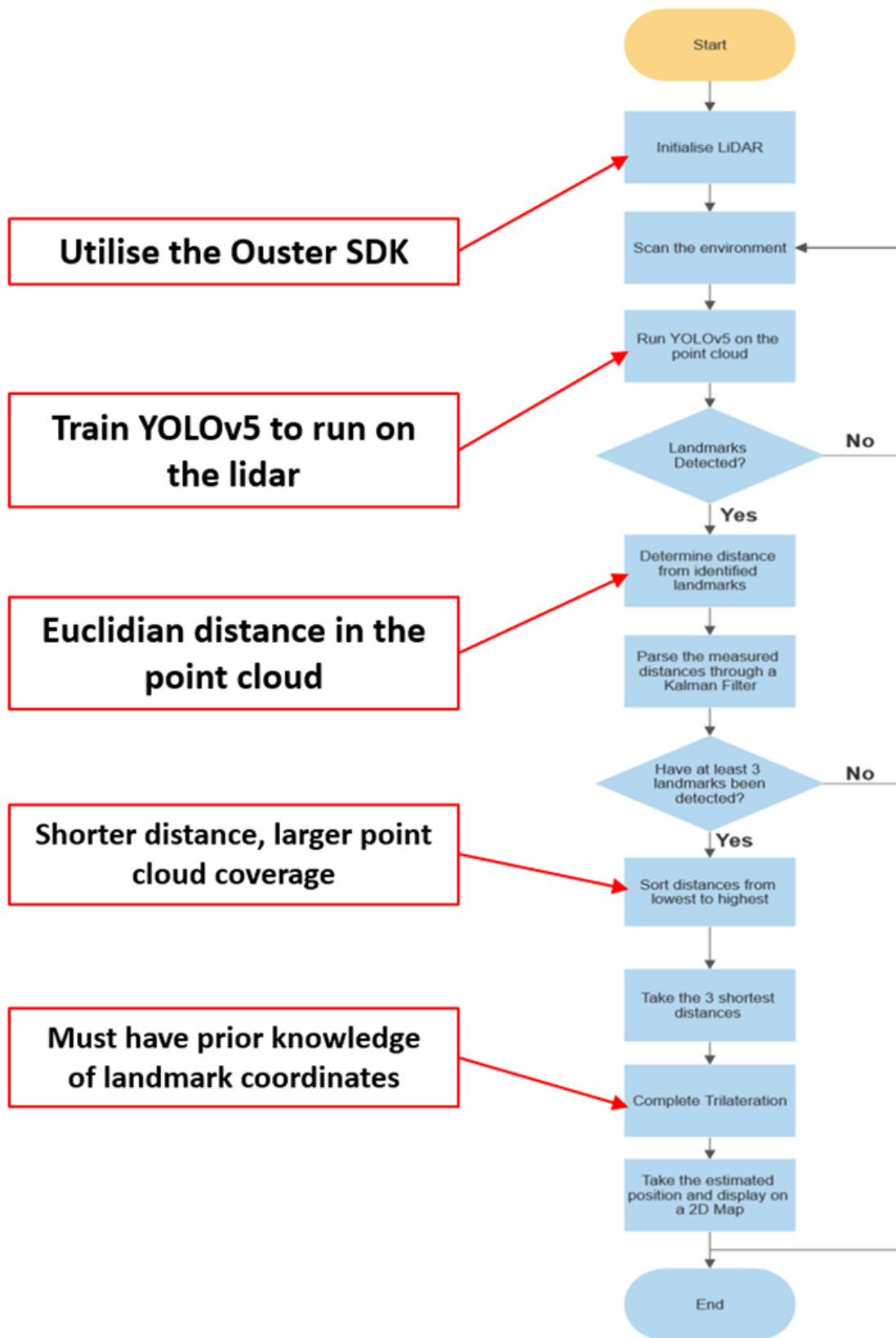


Figure 4.3.1: Complete System Flowchart

4.3.3 Setting up the System Environment

As a reference, this is the initial state of the environment, as shown in Figure 4.3.2.

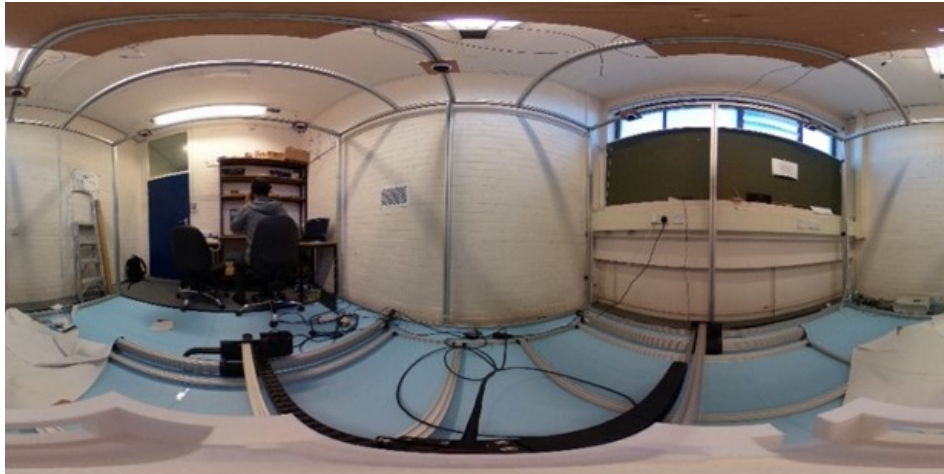


Figure 4.3.2: Initial Environment State

The current state of the environment includes a 2.50 x 2.50 x 2.16m (L x W x H) frame which will serve as the testing environment. A choice has been made to use the struts of this frame as the locations for the landmarks, as their locations should be easily measurable. Moreover, there is also a 3D plotter present in this environment. The plotter can move in all three dimensions and will be helpful in testing the accuracy of the localisation algorithm. Another important note is that both the plotter and frames are placed on top of a large millimetre graph mat. This will also be useful in measuring actual positions down to the nearest millimetre.

4.3.4 Creating Landmarks

As reported in the requirements Table 4.3.1, the landmarks should be non-electronic, and not require any continuous setup. Therefore, the ideal solution is for them to be made of a type of wood or plastic. As a LIDAR would have to be able to see these shapes, the preferred material will be a solid wood, as this will contrast best from the background. As for the landmarks themselves, since the object detector will have to identify each landmark as a unique object, these landmarks should be distinct shapes. Therefore, the distinct shapes will be a triangle, circle, octagon, trapezium, vertical rectangle, and horizontal rectangle. The shapes are also 12mm thick, which is important to note when developing the pose algorithm. The environment with the shapes attached around the frame are as shown in the Figure 4.3.3 below.

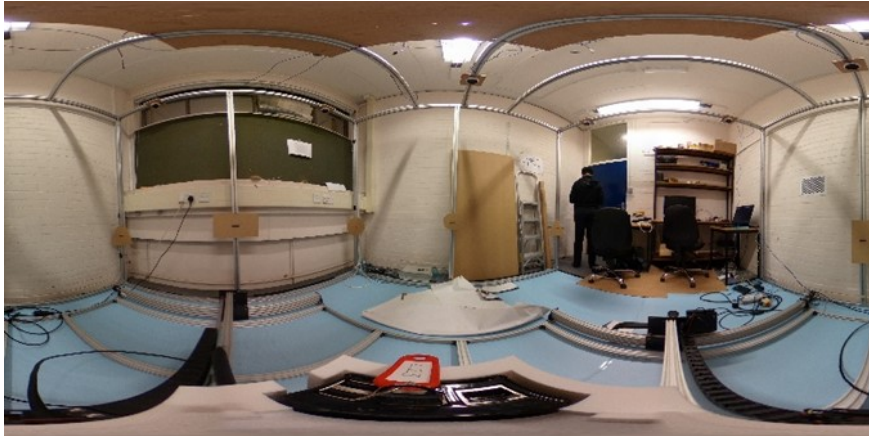


Figure 4.3.3: Deployed Testing Environment

As seen in the Figure 4.3.3, the shapes are attached to the struts of the frame and are clearly visible in the environment. As the landmarks have been set in the environment, the next step is to create a mount for the 360-degree camera.

4.3.5 Creating a 3D Camera Mount

The camera in question is a Ricoh Theta V 360-degree camera.



Figure 4.3.4: Ricoh Theta V camera [4-26]

This camera will be used in sensor fusion with the LIDAR, in particular using YOLOv5 on the images. Therefore, the camera must be closely aligned with the LIDAR as much as possible. The LIDAR has a small insert near the top, which could be used to place the camera, but a mount must be created. As seen in the Figure 4.3.4 above, the camera has a mounting hole, meaning that a model could be created which inserts into the top of the LIDAR.

As there is no prefabricated piece to insert into the LIDAR, one must be created. To create this piece, a design was made on the 3D modelling software called Fusion 360. The design is shown below.

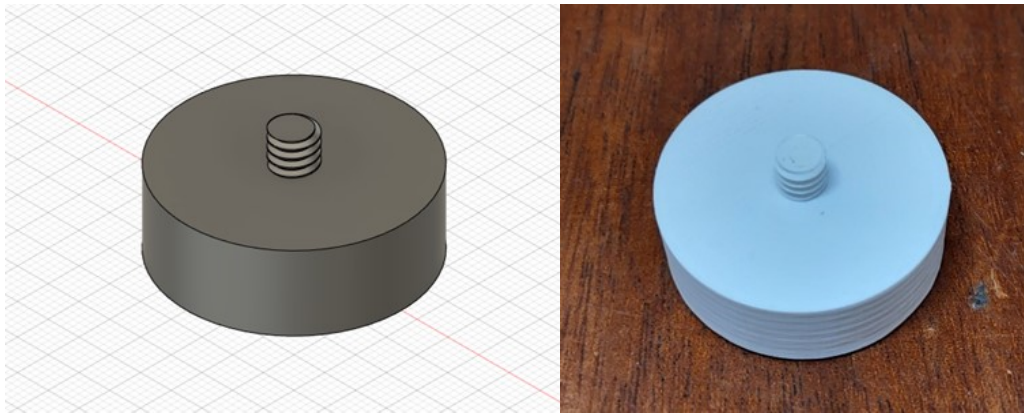


Figure 4.3.5: Designing and Creating the camera mount

The base of the mount has a diameter of 35mm and has an M6 thread with a pitch of 1.0mm, as shown in Figure 4.3.5. This model is then 3D printed and inserted onto the LIDAR.

4.3.6 Interfacing with the LIDAR using Python

The LIDAR is the Ouster OS1-32-U scanning sensor. This particular LIDAR has a vertical resolution of 32 channels, and a horizontal resolution of up to 2048 lines per rotation. The rotation rate can also go up to 20Hz and has a maximum deviation of plus or minus 10cm [4-27]. This sensor uses a gigabit ethernet output, and streams data using user datagram protocol (UDP) packets.

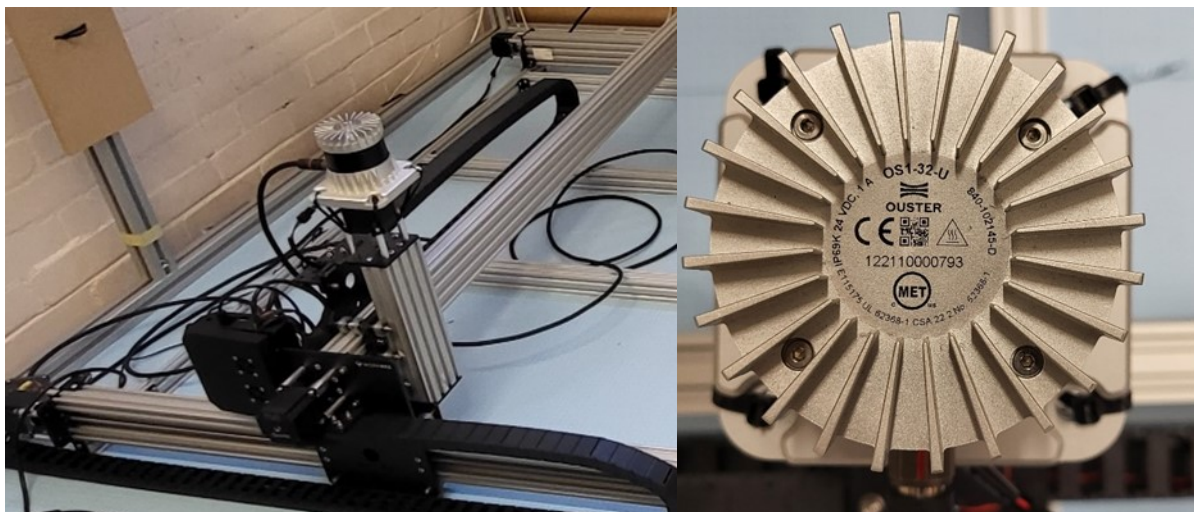






Figure 4.3.6: The Ouster OS1 LIDAR

Ouster LIDARs work by firing lasers in all direction using a rotating array of mirrors, and in this case, pulses are fired 12.6 million times per second, as shown in Figure 4.3.6. This LIDAR has four main data layers for each pixel within the point cloud, as shown in Table 4.3.3:

Table 4.3.3: Ouster Data Layers

Data Layer	Description	Example 2D Image [4-28]
Range	A representation of the distance from each pixel	
Signal	Displays the strength of the laser returned from an object. Objects further away will not be in as much detail.	
Near-IR	Displays the strength of light at the 865nm wavelength. Detail heavily depends on the amount of sunlight.	
Reflectivity	A display of how well the surface reflected the lasers.	

Each layer has its own benefits in terms of use case, with the range layer being key in all situations as this is which captures the distances.

For this project, the LIDAR is mounted upon the 3D plotter, so that it can be easily moved during testing. The LIDAR will have to complete a full 360-degree scan of the environment and produce a clear enough 2D LIDAR image such that YOLOv5 will be able to accurately identify the landmarks. Therefore, the point cloud has been configured to a resolution of 1024x10, and a vertical field of view of 45 degrees. One extremely beneficial feature of these LIDARs is their 1:1 spatial correspondence of points [4-28]. This means that each pixel within the 2D LIDAR exactly matches a single point within the 3D point cloud. This results in no unnecessary noise being introduced during processing, ultimately aiding in positional accuracy. As a result, the 2D images could be directly combined with YOLO and interpreted almost seamlessly.

Before training and running YOLO, it is integral to know how to interface with the LIDAR through Python code. Python code is the ideal method to conduct this project as there are a multitude of packages which accelerate the developmental process, in addition to having PyTorch, which is vital in using YOLO and other machine learning algorithms. As a bonus, the Ouster SDK is readily available in Python, making interfacing with Ouster LIDARs straightforward.

The goal behind interfacing with the LIDAR is to be able to correctly configure and interface with the LIDAR so that point clouds can be generated in real time. The first step in this process is to create a virtual Python environment. A virtual environment in python is simply a portable directory where the python packages can be easily installed or removed. This also instils a layer of abstraction, in the event where multiple projects are utilising hundreds of different python packages.

For this project, the PyCharm Python IDE has been used, due to its ease of use for Python programming, in addition to its ability to create virtual environments. The machine that the LIDAR will be connected to is the Dell PowerEdge R740 server. It is running on an instance of Ubuntu 22.04 LTS and is equipped with a 36-Core Xeon Gold 6140 CPU, as shown in Figure 4.3.7. This system has up to four ethernet ports, one of which will be used in directly connecting to the LIDAR.



Figure 4.3.7: Dell PowerEdge R740 [4-29]

The code required to interface with the LIDAR in real time is as follows in Figure 4.3.8: Code required to interface with the LIDAR:

```

11     # create empty config
12     config = client.SensorConfig()
13
14     hostname = 'os-122110000793.local'
15     # set the values that you need: see sensor documentation for param meanings
16     config.operating_mode = client.OperatingMode.OPERATING_NORMAL
17     config.lidar_mode = client.LidarMode.MODE_1024x10
18     config.udp_port_lidar = 7502
19     config.udp_port_imu = 7503
20
21     # set the config on sensor, using appropriate flags
22     client.set_config(hostname, config, persist=True, udp_dest_auto=True)

```

Figure 4.3.8: Code required to interface with the LIDAR

This code simply defines the LIDAR, which is connected to the host via ethernet, in terms of its network hostname, in addition to the UDP port that it should listen to. This is all after the ouster package has been imported. Once these parameters have been set, the following step is to communicate with the LIDAR, as shown in Figure 4.3.9.

```

24     # establish sensor connection
25     with closing(client.Scans.stream(hostname, config.udp_port_lidar,
26         complete=False)) as stream:
27         show = True
28         while show:
29             for scan in stream:
30                 reflectivity = client.destagger(stream.metadata,
31                     scan.field(client.ChanField.REFLECTIVITY))
32                 reflectivity = (reflectivity / np.max(reflectivity) * 255).astype(np.uint8)
33                 cv2.imshow("scaled reflectivity", reflectivity)

```

Figure 4.3.9: Code to show the Reflectivity layer in real-time

This block of code simply combines the previous parameters and runs the 'Scans.stream' function. This function attempts to connect to the LIDAR using the supplied hostname, and UDP port. If successfully connected, the program reads each point cloud produced by the LIDAR as a single scan, and outputs the reflectivity layer using 'cv2.imshow'. The output will be the 2D representation of the point cloud, in the reflectivity layer, as shown in Figure 4.3.10.

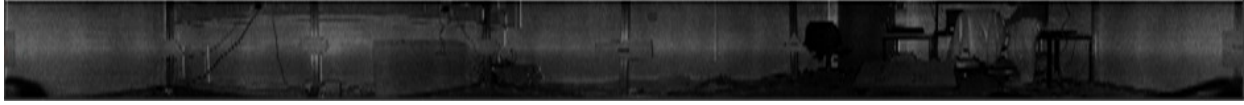


Figure 4.3.10: Output of the Reflectivity layer

Note that it is using the reflectivity layer but converted to greyscale for viewability. Now that the LIDAR has been successfully interfaced with via Python, it is now possible to begin training YOLOv5 to detect the landmarks around the environment.

4.3.7 Training YOLOv5 for object Detection

YOLO is an object detection model which is widely known for its high speed and real-time detection accuracy. This particular model can act on images, videos, live streams, and a multitude of different applications. Before training the system, it is important to understand how the YOLO process functions.

In the case of a single image input, YOLO initially resizes the image to a predetermined fixed size. This allows the model to work within set parameters, rather than a dynamic range, ultimately increasing performance. Next, the normalised image is passed into a CNN backbone, to extract features at different scales. This version of YOLO in particular uses the Cross Stage Partial (CSP) network, which was created to improve the accuracy and efficiency of CNNs by finding a solution to capturing more complex features from images, without requiring as much computational power [4-30].

Once feature maps have been created by the backbone, it is passed to a 'neck' network, which combines the feature maps to form a single representation. The head network, which is the following step, is vital as this is where the model begins to detect potential objects in the features. It applies bounding boxes, which are essentially rectangular boxes around each detection and assigns it a class probability – how likely this object belongs to a particular class. The output of this head is likely to have many overlapping boxes, so a non-maximum suppression (NMS) algorithm is applied to remove redundant boxes. It determines redundant boxes by checking their class probability values against a threshold value and acts according to whether it is above or below this value. This would not remove all overlaps, as some occlusion can occur legitimately. Finally the resulting image is produced, where the bounding boxes are surrounding any detected objects. Yolov5 flow model is shown in Figure 4.3.11

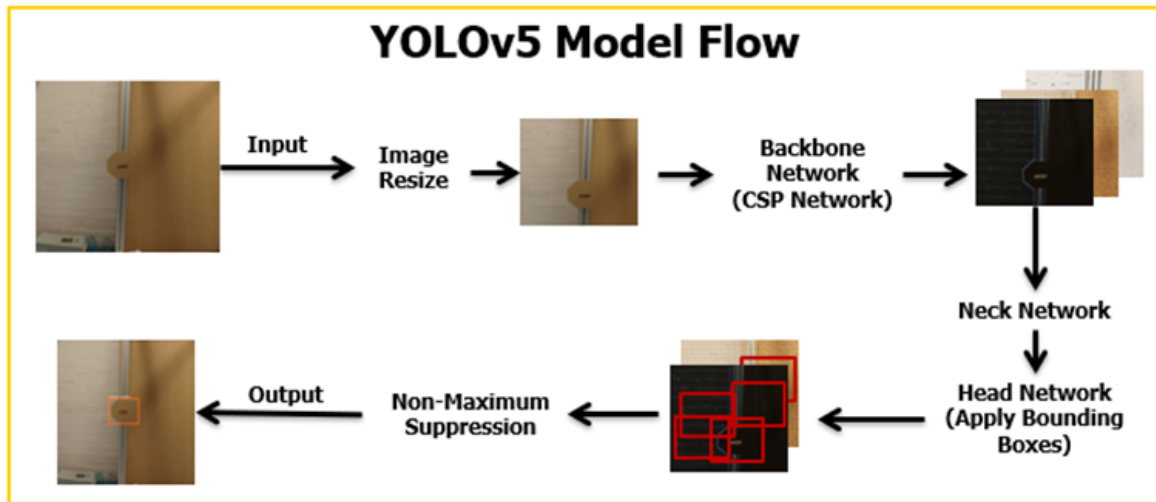


Figure 4.3.11: YOLOv5 Model Flow

In a practical sense the weights file, which is a binary file that could be considered as the source of the object detector's 'knowledge', is trained against the objects one would like to detect. This is where the 'training' term is coined when making object detectors. In order to train YOLO, a large dataset of images must be acquired. A training dataset would usually require thousands of labelled images, with the objects captured in different angles, distances, lighting, and orientations. Due to the fact that the reflectivity layer does not rely on lighting, the easiest solution to create a large dataset is by splitting each frame of a LIDAR recording into images.

In order to take a recording of the scene, the Ouster visualiser software was utilised. This software provides a simple interface to the LIDAR, and contains features such as recording scenes, playback of recordings, and measurements down to single points within the cloud. Through the program, it is possible to connect to a live LIDAR, through an available ethernet port. An interface appears with multiple parameters that can be configured, as shown in Figure 4.3.12.

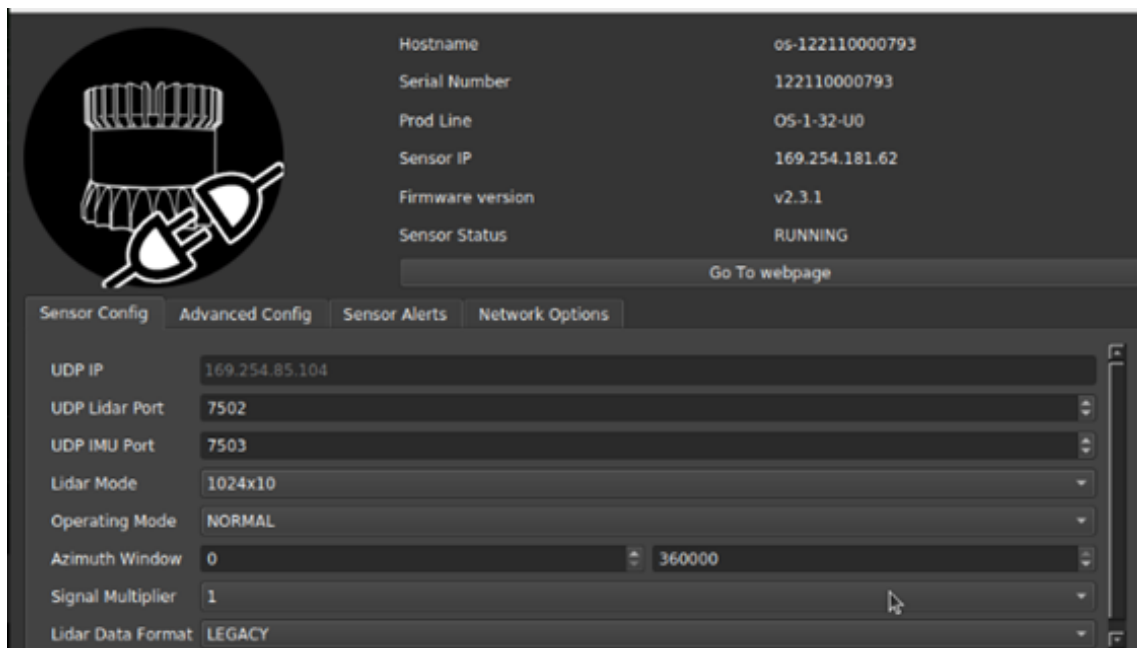


Figure 4.3.12: Configuring the LIDAR on Ouster Studio

The key settings that change the sensor's readings are the: LIDAR Mode, Operating Mode, Azimuth Window, and Signal Multiplier, as shown in Table 4.3.4.

Table 4.3.4: LIDAR Configuration Parameters

Configuration Parameter	Description
LIDAR Mode	Changes the horizontal and vertical resolution rates of the sensor. In this project it is set to 1024x10.
Operating Mode	Can be set to run normally or on standby. Standby will sleep until commands are received. It is set to normal in this project.
Azimuth Window	The angle in which the LIDAR should scan, from 0 to 360 degrees. It is set to 360 by default.
Signal Multiplier	The power in which the LIDAR lasers are fired at. Higher signal power, the further the range. As an azimuth of 360 is being used, the multiplier is set to 1.

The recording environment involved the LIDAR remaining stationary in the environment, with the shapes in the environment present in LOS. As the recording runs, one shape will be blocked to create images where not all objects are present. This process is repeated with the LIDAR in different positions within the environment. As previously mentioned, the LIDAR is mounted on a 3D plotter, which is connected to the server by a router, and is controlled by a Python program. This Python program simply connects to the 3D plotter's local Internet Protocol (IP) address and sends commands via Transmission Control Protocol (TCP), as shown in Figure 4.3.13. The plotter has a resolution of 1270 by 1270 by 50 (XYZ), and also has support for voice commands via a connected microphone. For ease of use, textual inputs were utilised instead.

```

def move_plotter(pos):
    # create a telnet link to the 3D plotter
    host = "192.168.251.2"
    tn = Telnet(host, 23)
    # wait for the link to be established
    time.sleep(1)
    # convert the position to a binary Gcode command
    command = f"G1 X{pos[0]} Y{pos[1]} Z{pos[2]} \n"
    print(command)
    binary_command = bytes(command, 'utf-8')
    tn.write(binary_command)
  
```

```

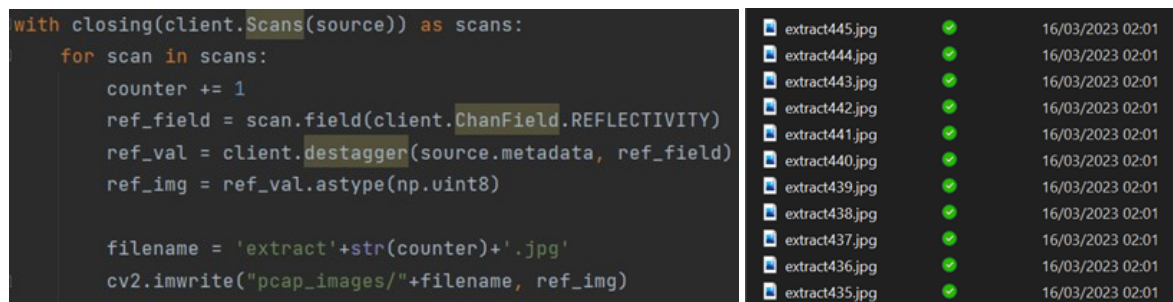
Do You Want to Use Microphone? [y/n]
Do you want to use the 3D plotter? [y/n]
Trying to connect to the plotter...
Please Wait for Blue to completely HOME

Enter 1 for New UE Location
Enter 2 for Taking Measurements
1
Please enter new X coordinate
0
Please enter new Y coordinate
0
Please enter new Z coordinate
50
G1 X0 Y0 Z50

Enter 1 for New UE Location
Enter 2 for Taking Measurements
  
```

Figure 4.3.13: Code to move the 3D Plotter

To summarise, the LIDAR is recording in multiple static positions, by moving along a 3D plotter. Once recordings were saved, some code would have to be created to split and encode each frame into an image. The following code, shown in Figure 4.3.14 simply reads each scan in the file and encodes the reflectivity layer into a readable image format. The lossy encoding of jpg was the format of choice as the marginal difference in quality against a lossless encoding was negligible.



```

with closing(client.Scans(source)) as scans:
    for scan in scans:
        counter += 1
        ref_field = scan.field(client.ChanField.REFLECTIVITY)
        ref_val = client.destagger(source.metadata, ref_field)
        ref_img = ref_val.astype(np.uint8)

        filename = 'extract'+str(counter)+' .jpg'
        cv2.imwrite("pcap_images/"+filename, ref_img)

```

extract445.jpg	✓	16/03/2023 02:01
extract444.jpg	✓	16/03/2023 02:01
extract443.jpg	✓	16/03/2023 02:01
extract442.jpg	✓	16/03/2023 02:01
extract441.jpg	✓	16/03/2023 02:01
extract440.jpg	✓	16/03/2023 02:01
extract439.jpg	✓	16/03/2023 02:01
extract438.jpg	✓	16/03/2023 02:01
extract437.jpg	✓	16/03/2023 02:01
extract436.jpg	✓	16/03/2023 02:01
extract435.jpg	✓	16/03/2023 02:01

Figure 4.3.14: Code and output in slicing LIDAR recordings to images

By running this code on a 30 second recording, a minimum of 300 images were sliced (as the LIDAR is running at 10 frames per second). In total, 5812 images were created to add to the dataset, containing the shapes from multiple different angles and sizes.

Now that a dataset has been formed, the next step in the training process is to manually describe the objects that YOLO should look detect by labelling each shape in every image. This process is extremely tedious and time consuming however, this is the core of the training process. For this project an online tool called Roboflow [4-31] was utilised to ease the labelling process. This site allows images to be directly uploaded into a new project and lets the user draw boxes around the objects they wish to detect, in addition to creating new classes. For this project, Figure 4.3.15 below shows how each image would be labelled.



Figure 4.3.15: Using Roboflow to annotate images

Notice how each shape within the environment is presented in a user-friendly manner, highlighted by the colour coordination of boxes and classes. This image-label-next image process was repeated for at least 1000 iterations, resulting in a dataset capable of training YOLOv5. Once finished, the following step is to split the dataset into three sections – train, validation, and test. Train, the largest dataset, is dedicated to training the model. Test is used to validate the trained model. Finally, the validation data is used to tune the hyperparameters of the object detector. Hyperparameters are the settings that are usually manually configured prior to training such as learning rate, the number of epochs, and batch size. The ideal split is 66%, 24%, and 11% respectively as more emphasis should be placed on training.

Roboflow provides further pre-processing and augmentation features to stretch out the usefulness of the dataset for example, it can automatically stretch or rotate the labelled objects. For this dataset these settings have been left as-is to maintain the integrity of the real environment. Once the images have been split and any further processing has been done, the data must be generated into an image-annotation pair. This is where each image's labelled classes are written into a text file, with coordinates mapping exactly where the boxes were

drawn. As a final procedure, a configuration file must also be created, to list where the dataset is stored, as shown in Figure 4.3.16.

```
coco.yml
1 # train and val data as 1) directory: path/images/, 2) file: path/images.txt, or 3) list: [path/ima
2 train: ./data/train
3 val: ./data/valid
4 test: ./data/test
5
6
7 # number of classes
8 nc: 6
9
10
11 # class names
12 names: ['circle', 'horizontal_rectangle', 'octagon', 'trapezium', 'triangle', 'vertical_rectangle']
13
```

Figure 4.3.16: Configuring the training file

Once these image-label files have been moved into the relevant directories, it is now possible to begin running the trainer. The trainer comes in the form of a Python file, which heavily utilises the PyTorch package. In this file, the process shown in Figure 4.3.17 was completed.

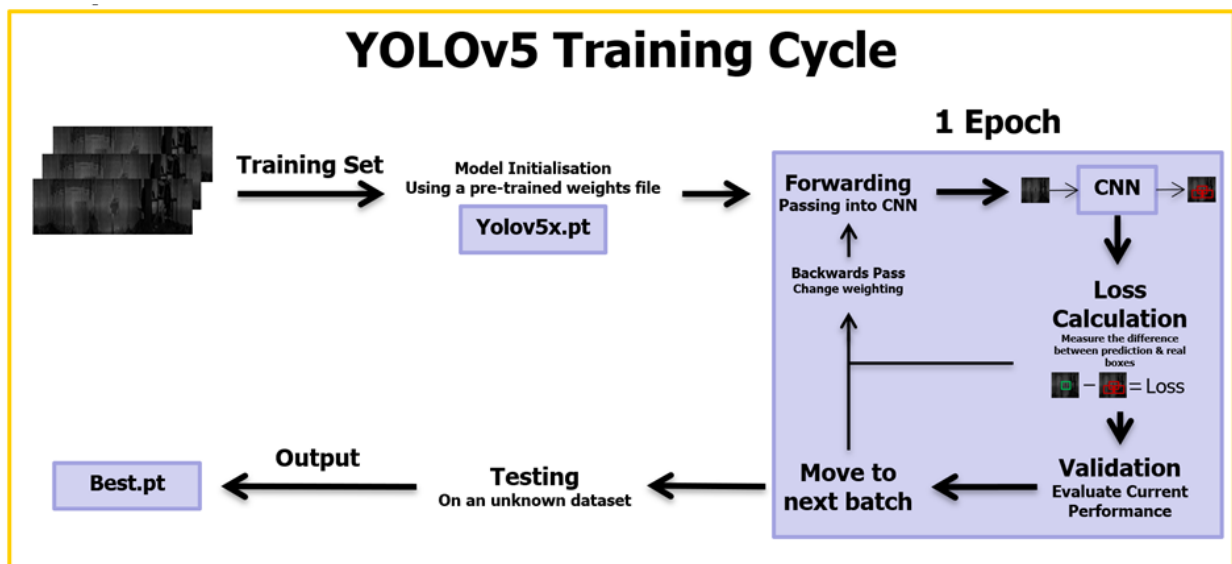


Figure 4.3.17: YOLOv5 Training Cycle

The training process begins by loading the hyperparameter values that were set beforehand. Then, it begins by initialising a pre-existing weights file, provided there is one. Using a pre-trained weights file speeds up the training process and is recommended. The following step is to forward the training images into the neural network which in turn generates predictions for the location class of the objects within the image. It is extremely unlikely that it is near 100% accuracy in the first pass, so a loss calculation is conducted to numerically compare the difference between the predicted and actual labels. Using this calculated loss, the algorithm adjusts its weighting, and backpropagates the error through the neural network, in an attempt to reduce the loss. This occurs multiple times on each image within a batch. Once a single batch is completed, this is considered to be one epoch. A training model will normally require at least 100 epochs before a reliable model is created. Upon the completion of the desired number of epochs, the trainer then runs on a training set, which was previously unknown to the model.

For this project, the trainer was initialised to run for 250 epochs, however the program has a special feature where it will stop training should there be no significant gain in accuracy. As seen in the Figure 4.3.18 below, the model was on its 9th epoch, and was producing box loss, object loss, and classification losses of 7%, 2%, and 0.6% respectively. Generally speaking, these losses should be below 0.01 for optimum reliability. In this project, a dedicated graphics processing unit (GPU) was tasked to run the object trainer due to its capability of performing vast parallel processes, in addition to having dedicated video memory.

```

Epoch      GPU_mem  box_loss  obj_loss  cls_loss  Instances  Size
8/499      5.66    0.07425  0.02628  0.006406  79         416: 100% |██████████| 43/43 [00:13<00:00, 3.23it/s]
Class      Images  Instances  P         R         mAP50  mAP50-95: 100% |██████████| 9/9 [00:02<00:00, 4.39it/s]
all        210     730       0.94     0.968    0.974  0.569
    
```

Figure 4.3.18: Running the YOLOv5 Trainer

As previously mentioned, the algorithm is able to test for performance after every epoch. In this process, the program conducts multiple tests and produces multiple graphs. A confusion matrix shows the probability distribution between the predicted classes against the true classes, Figure 4.3.19.

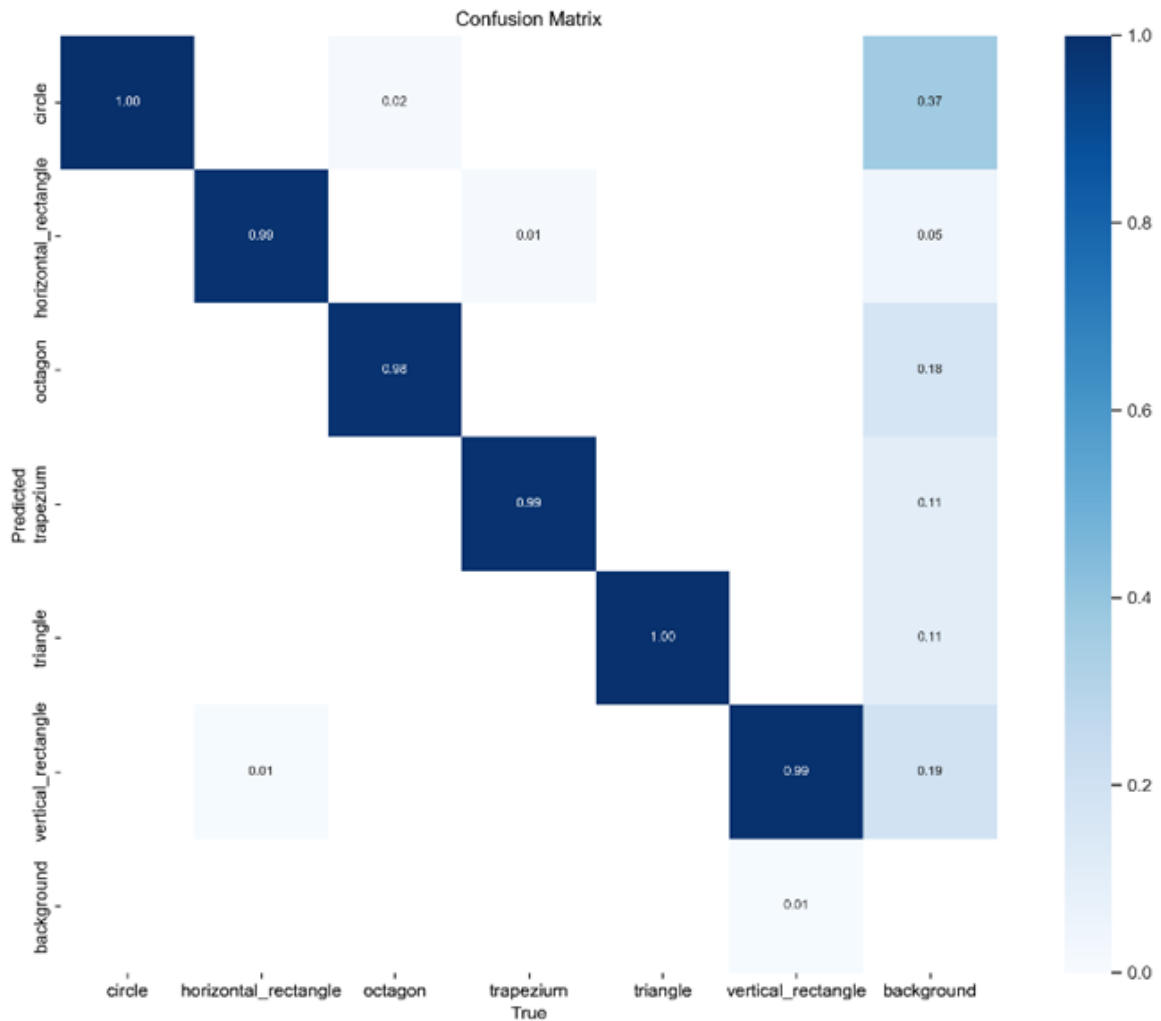


Figure 4.3.19: Confusion Matrix for the newly trained weights file

Ideally the matrix should show 1 in the centre diagonal, meaning that each class in the dataset was detected correctly. In this particular training, the model did very well in correctly

detecting classes, but not as much in detecting the background. By using these statistics, an F1 score can be calculated. This score combines the model's precision and recall into a single value, which represents the overall accuracy. Precision is a measure of the fraction of true positive detections among all detections, while recall measures the fraction of true positive detections among all actual objects in the image. This score ranges from 0 to 1, with 1 being the best score. A high precision suggests that the model produces very little false positives, while a high recall shows that the model is detecting most of the true positive objects.

$$F1 \text{ Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

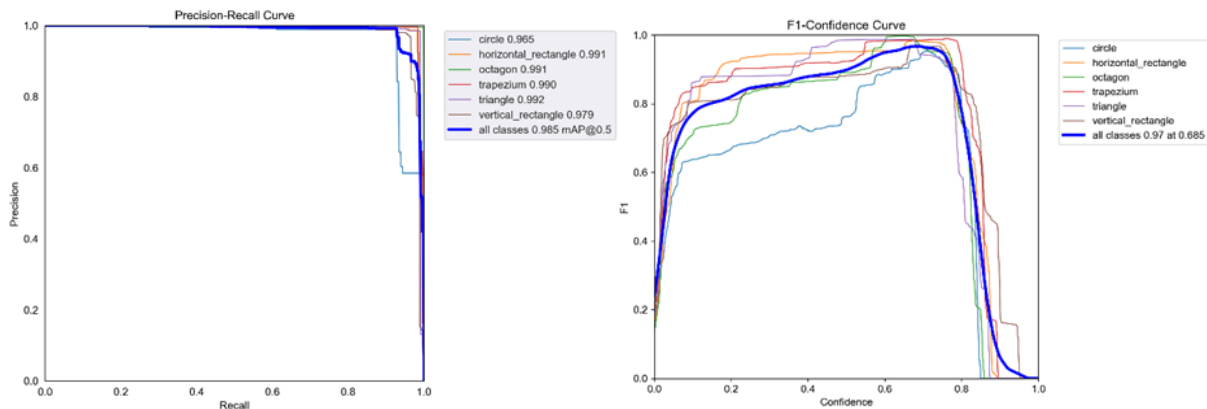


Figure 4.3.20: Precision Recall and the F1-Confidence

The above Figure 4.3.20 precision-recall curve shows that the model has most classes achieving at least 0.99 apart from the circle and vertical rectangle. The following F1-Confidence Curve in Figure 4.3.20 shows that the model achieves the highest F1 score of 0.97 when it has a minimum confidence score of 0.685. This means that the minimum confidence level when running this object detector should be set to around this value. There is a trade-off to be had here however, where the confidence threshold could be set higher, but at the risk of losing true positive detections. Similarly, a lower confidence could be set to the detriment of detecting false positives.

Overall, a new weights file was created as a result of training YOLOv5 against the LIDAR dataset. The new file is designed to detect the shapes around the environment, through the LIDAR's 2D reflectivity output. This dataset was created by slicing multiple thirty second LIDAR recordings into individual images and labelling the objects through Roboflow. The optimal confidence level of this model is 0.685, based off the resulting F1 score. The next step is to use this newly trained object detector in combination with another algorithm to detect the position of the LIDAR within the environment.

4.3.8 Using Strictly LIDAR for Pose Estimation

Now that the object detector has been trained to work with the reflectivity layer of the LIDAR, it is now possible to begin creating the algorithm which will estimate the sensor's position within the environment. The Figure 4.3.21 below shows the designed flow in this algorithm.

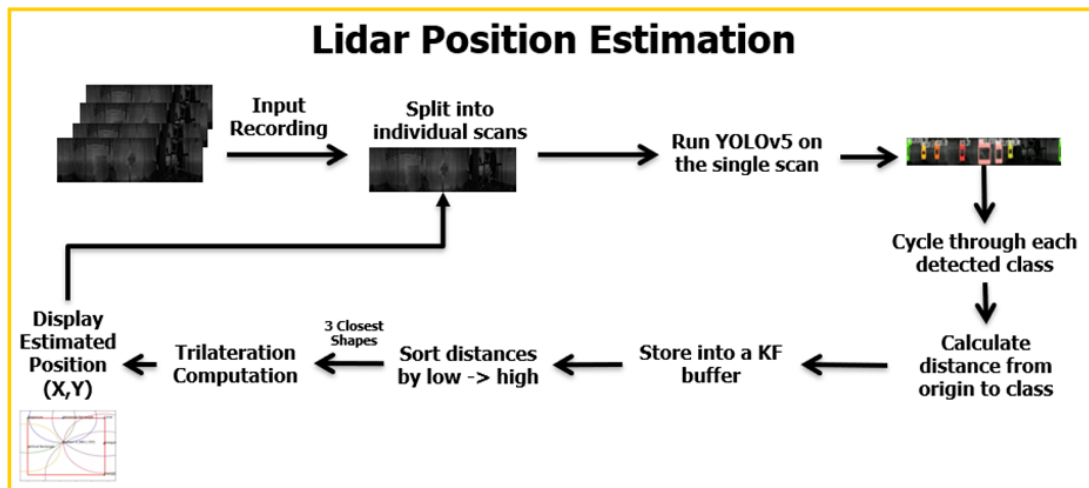


Figure 4.3.21: LIDAR Position Estimation

The initial step is to apply YOLO onto the LIDAR's output, detect shapes on each scan, and calculate the distances from them within the environment. Having accurate distance measurements will be paramount as this could heavily change the estimated position.

4.3.9 Processing YOLOv5 on Point Clouds

The first script is to run YOLOv5 on the point cloud. From a setup perspective, aside from the weights file, the Python code which reads into a LIDAR's recording only requires the LIDAR metadata, which contains the LIDAR resolution, frequency, number of channels, and other configuration data. This is so that the recording can be played back correctly by the algorithm. In addition to this, the minimum confidence threshold must also be declared, and as found in the testing results, this should be set to around 0.685, as shown in Figure 4.3.22. There is also another parameter, which allows the algorithm to display an OpenCV window of the recording and bounding boxes in real time.

```
( '--weights', nargs='+', type=str, default=ROOT / 'weights/best.pt', help='model path(s)')
( '--source', type=str, default='./lidar_recordings/423_0.pcap', help='file/dir/URL/glob, 0 for webcam')
( '--metadata-path', type=str, default='./lidar_recordings/1024x10.json', help='metadata path')
( '--view-img', default=True, action='store_true', help='show results')
( '--conf-thres', type=float, default=0.685, help='confidence threshold')
```

Figure 4.3.22: Configuring parameters for detection

When running this code with the correct source and metadata, the following output is produced, as shown in Figure 4.3.23.

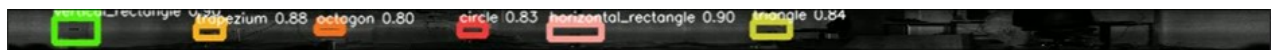


Figure 4.3.23: YOLO output on LIDAR recording

In this output, each shape has been detected alongside their confidence levels, as shown in Figure 4.3.24. Each shape that has been detected has a confidence of at least 80%, and it can also be seen that the bounding boxes have been correctly placed.



Figure 4.3.24: YOLO output with a blocked shape

In another recording however, an entity is blocking the circle in the scene, but the object detector has reacted correctly as there is no longer a box in that position. This shows that the

training of the model has been fruitful. As a result, measuring accurate distances should be possible.

4.3.10 Extracting Distances from Point Cloud

The theory of extracting distances from the point cloud recording depends on the ability of the LIDAR’s 1:1 spatial correspondence of points. In theory, the coordinates that YOLO has mapped the bounding boxes onto should also be mapped to a set of points within the 3D point cloud. Therefore, by listing the points within the box as a region of interest, the algorithm should be able to work out a particular coordinate which best represents the shape. This coordinate can then be mapped onto the point cloud, and the appropriate x, y, and z coordinates point can be read. Then, the Euclidian distance can be found from that particular point to the sensor (which is set as the origin).

$$Euclidian\ distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

The Figure 4.3.25 below shows this flow.

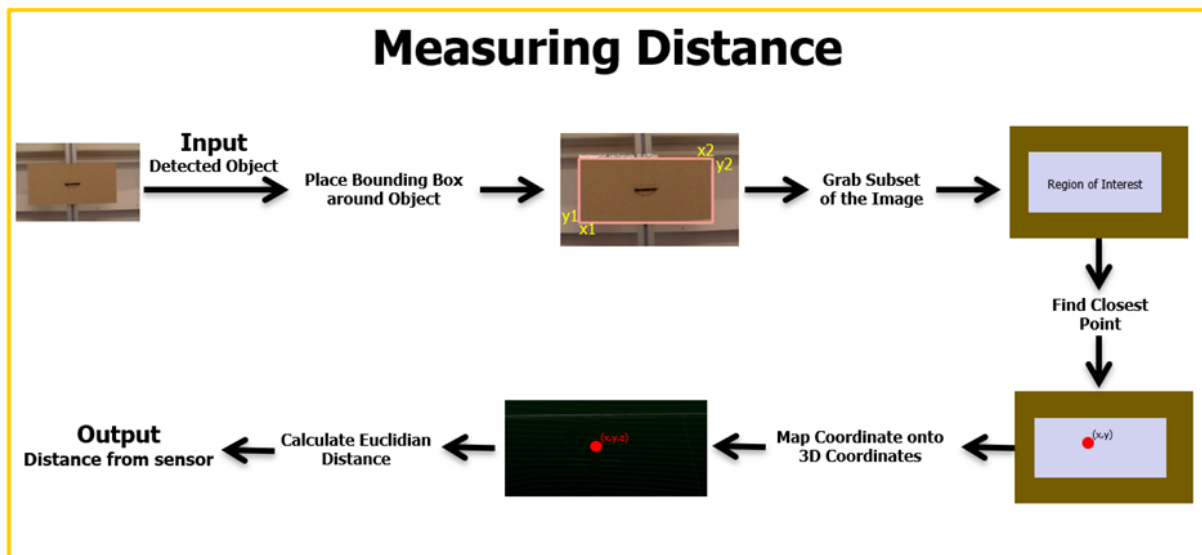


Figure 4.3.25: Measuring Distance

This diagram shows how each detected object is processed to calculate the distance from itself and the sensor. For each bounding box, the algorithm notes its coordinates, and produces a subset of the overall image. In this subset, a subarray is created which covers the area in which the object is detected within the image. With this region of interest, a function is run to find the nearest point in the array. This point represents the coordinate of the object which is closest to the sensor. Using this special coordinate, it is then reflected upon the range layer of the point cloud using a pre-made look up table (LUT), to retrieve the coordinates. Finally, the distance calculation is conducted to get the distance from the shape’s closest point and the sensor.

In practice, the code which conducts this specific process is shown in Figure 4.3.26 below:

```

poi_roi = np.unravel_index(range_roi.argmax(), range_roi.shape) #(y,x) in roi
poi_x = poi_roi[1] + x1 # Map coordinate onto the region of interest
poi_y = poi_roi[0] + y1
poi = (poi_y, poi_x) #(y,x) in global
poi_list.append(poi)

xyz_val = xyz_destaggered[poi] # Map coordinate using the LUT
xyz_list.append(xyz_val)
xyz_1 = xyz_list[0]

# Calculate Distance using LUT values
dist = math.sqrt((xyz_val[0])**2 + (xyz_val[1])**2 + (xyz_val[2])**2)

# Annotate the object with the distance value
annotator.box_label(xyxy, label, dist, color=colors(c, True))

```

Figure 4.3.26: Code to calculate distance

The final line produces an annotation on the playback which shows the class name and distance measured. By running this code procedure on the playback, the following output shown in Figure 4.3.27 is produced:



Figure 4.3.27: LIDAR output with measured distance

Notice how each shape has differing distances, showing that the function is working as intended. Now that distances can be retrieved; it must now be filtered using a Kalman filter.

4.3.11 Implementation of Kalman Filters

In essence, a Kalman filter (KF) is a mathematical algorithm which uses a series of measurements over a period of time to estimate the state of a system. It is a closed loop function which follows a recursive formula. It takes an input measurement and uses a number of historical inputs to update the output value. In this use case, the system would input a number of distances, one per detection of a particular object, and the filter would output an estimation. This is used to remove any extreme outliers which could be sent from the object detection algorithm. The Figure 4.3.28 below shows the intended use case.

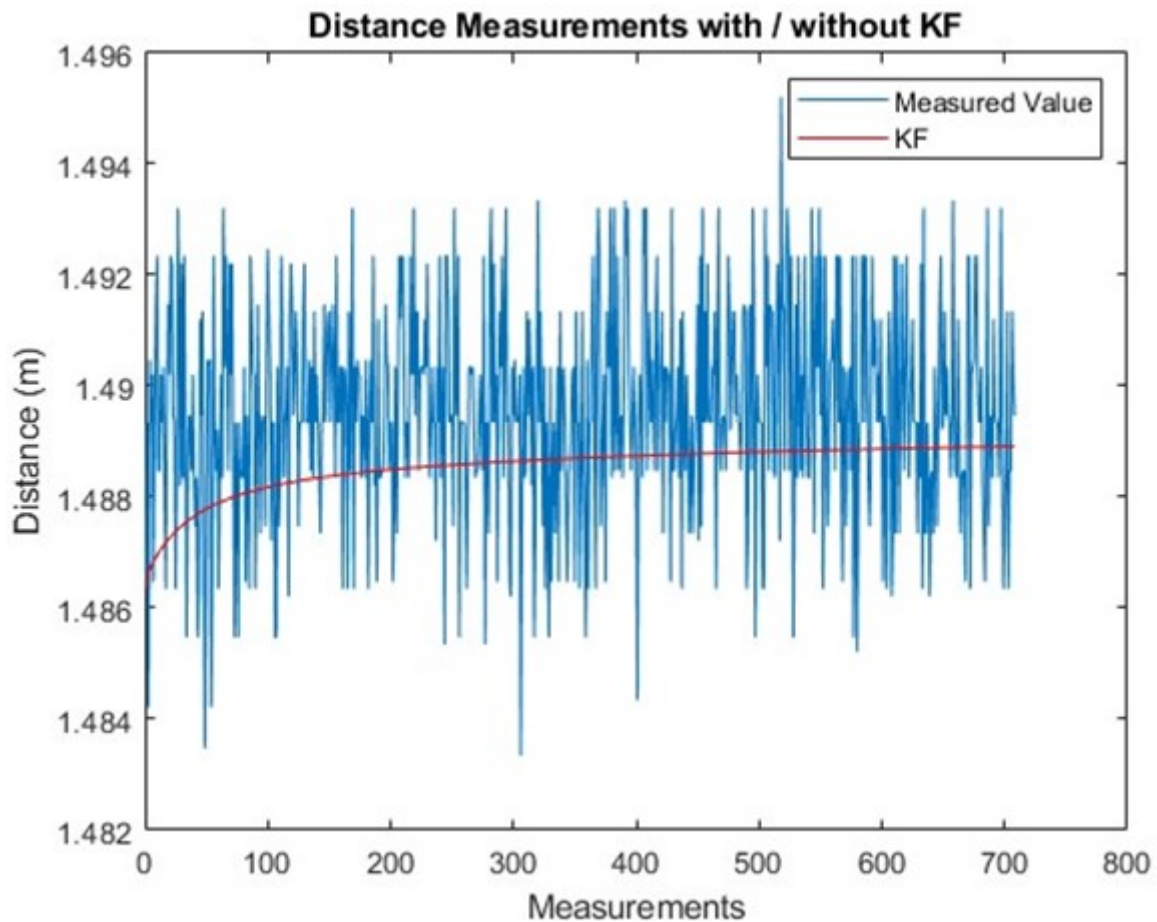


Figure 4.3.28: Applying the KF on distances

Theoretically speaking, the KF should eliminate the noise when distances are measured and produce a more stable output. In order to implement this, the code filter should be receiving measurements whenever the object detector measures a distance. As there are multiple classes, the code should also be able to recognise which shape the distance value belongs to. Moreover, these distances must be held in a buffer so that the KF has its recursive data.

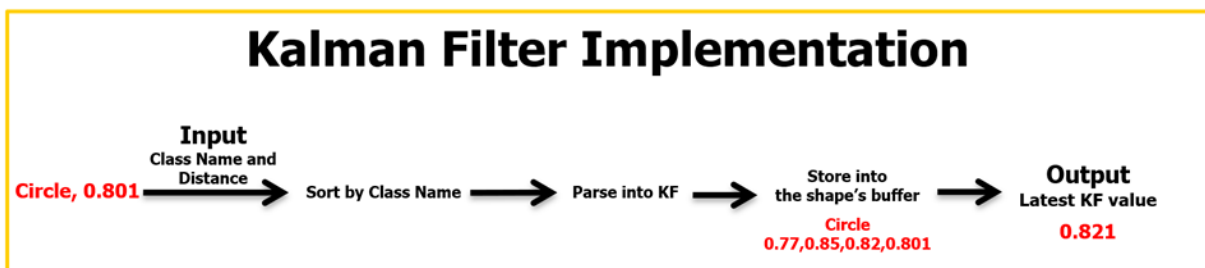


Figure 4.3.29: Kalman filter implementation

Simply put, the code should call a function to update the KF, with the class name, and distance value as parameters, as shown in Figure 4.3.29. The resulting function will read the class name and subsequently call the KF subroutine. The filter would then run the length of the buffer and output the most recent filtered value. The code implementation is shown in Figure 4.3.30 below:

```

def extract_data(shape_name, dist):
    if shape_name == "vertical_rectangle":
        vert_rect.appendleft(dist) # Update Buffers
    elif shape_name == "horizontal_rectangle":
        hori_rect.appendleft(dist)

# Kalman Filter
def kalman(buffer):
    max_len = int(buffer.maxlen)

    if len(buffer) == 10:
        base_val = float(buffer[0])

    for m in range(0, max_len):
        kalman_filterbuf.appendleft(base_val + (1 / (1 if m == 0 else m)) * ((float(buffer[m])) - base_val))

        base_val = kalman_filterbuf[m]
    return base_val

```

Figure 4.3.30: Code implementing KF use

Now that the KF has been implemented. It is possible to begin estimating the position of the sensor within the environment.

4.3.12 Trilateration Estimation using Distances

Using the calculated distances, it is now possible to estimate the position of the UE within the environment. As this project is only focusing on the two-dimensional position of the sensor, the most appropriate technique is trilateration. This is the most optimal as it only requires the coordinates of the landmarks (in this case the shapes) in the environment, and the distances from those landmarks. In addition to this it only requires three distances so there is flexibility in the event that one or more shapes are blocked. The flow that the code should follow is shown in Figure 4.3.31 below:

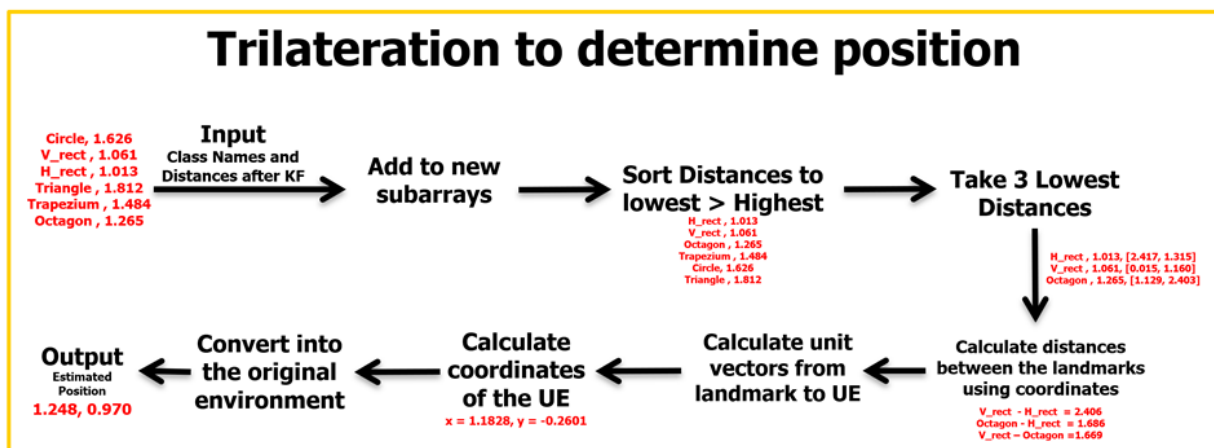


Figure 4.3.31: Process to calculate position using trilateration

This figure represents the steps that the algorithm should take to determine position. As the distances for each shape come from the KF, the trilateration algorithm can read them, and map them onto an array. The coordinate for each shape is vital in this process, and as such must be carefully measured. To measure each shape's coordinates within the environment, a reference point must be chosen. For accurate measurements, a laser distance measuring tool, as shown Figure 4.3.32, has been utilised for distances with a tolerance of one millimetre.



Figure 4.3.32: Leica Disto D2 Laser Measuring tool

Using this tool, it is possible to map the environment. From the origin, the shapes around the perimeter of the site, as shown in Figure 4.3.33, are able to be measured.

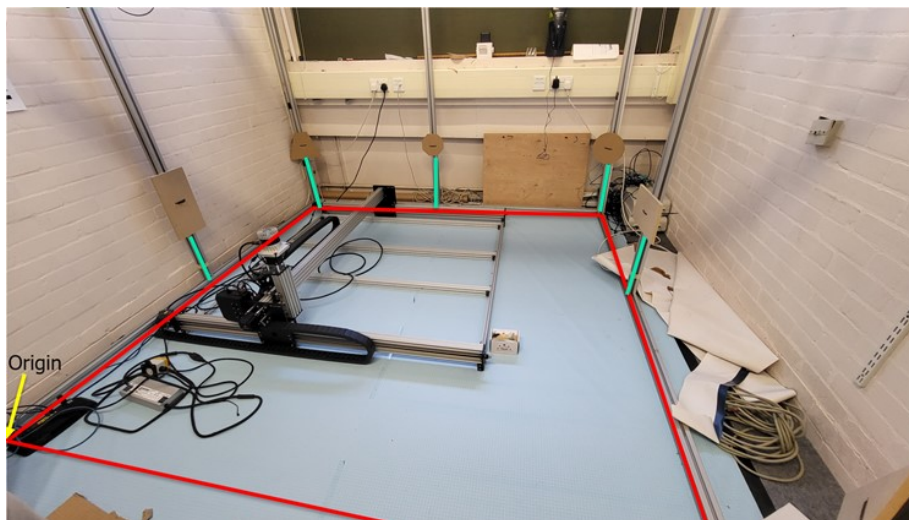


Figure 4.3.33: Mapping the system environment

By measuring from the x and y axis, the following coordinates for each shape were derived, as shown in Table 4.3.5:

Table 4.3.5: Measured Shape Coordinates

Shape Name	X-Axis	Y-Axis
Vertical Rectangle	0.027	1.160
Trapezium	0.033	2.403
Octagon	1.129	2.391
Circle	2.414	2.374
Horizontal Rectangle	2.405	1.315
Triangle	2.388	-0.003

Notice that coordinates are not linear – this is due to the fact that the exterior frame surrounding the environment is not uniform, making this process difficult to accurately measure. The coordinates are including the 12mm thickness of the material used. Using these coordinates, it is now possible to code this trilateration algorithm.

By sorting the distances from lowest to highest, the top three shortest distances can be selected. The shortest three have been chosen as from an error point of view, objects further away are more likely to produce incorrect results. This is due to the fact that the object

themselves would have a smaller region of interest on the LIDAR output, meaning that the bounding boxes are more likely to be out of place. After sorting, the distances between the landmarks must be found. This calculation utilises the coordinates of those shapes and the Euclidian distance formula as shown in equation 4.6.2.2. This calculation occurs three times to cover all combinations between the landmarks and is then used to calculate unit vectors. These unit vectors are used to represent the direction in space. Using these unit vectors, it is possible to work out the direction in which the sensor is at. Then, it is possible to work out the estimated position of the sensor in vector form (i and j directions). Then, by reusing the measured distances, it is possible to find the x and y coordinates of the sensor. From a code perspective, the key lines are shown in Figure 4.3.34 below:

```
# Calculate distances between landmarks
a = np.linalg.norm(C - B)
b = np.linalg.norm(C - A)
c = np.linalg.norm(B - A)

# Calculate unit vectors from landmarks to object
ex = (B - A) / c
i = ex.dot(C - A)
ey = (C - A - i * ex) / np.linalg.norm(C - A - i * ex)
ez = np.cross(ex, ey)
d = np.linalg.norm(B - A)

# Calculate coordinates of object
j = ey.dot(C - A)
x = (dA**2 - dB**2 + d**2) / (2 * d)
y = (dA**2 - dC**2 + (i**2 + j**2)) / (2 * j) - (i / j) * x

# Convert coordinates back to original coordinate system
position = A + x * ex + y * ey
```

Figure 4.3.34: Coding in the trilateration calculation

The resulting position variable is simply a list with two values, the x and y coordinates. This is the estimated position in the environment.

4.3.13 Displaying position on a 2D Map

A human readable interface where the real-time position can be shown as each frame of the LIDAR is processed must be made. As it is a two-dimensional estimation, it would be best to show the map on a coordinate grid. Python has a package with the name of 'matplotlib' which can easily create a grid, in addition to the ability to plot shapes and specific points. Moreover, it can be constantly updated, which is required as the estimated position will update as new scans are interpreted. The coordinates for the shapes are already known, so the environment can already be built. The below Figure 4.3.35 shows how this map can be coded.

```
graph.plot(hori_rect[0], hori_rect[1], 'o')
graph.text(hori_rect[0], hori_rect[1], 'Horizontal Rectangle')
```

Figure 4.3.35: Displaying a shape

This code is repeated for each shape. In addition to displaying the shape's positions, the exterior frame should also be displayed. As previously mentioned, this is not a uniform shape, so must be coded in as a polygon, rather than a square or rectangle.

```
frame_coords = [(0, 0), (2.401, -0.027), (2.436, 2.401), (-0.01, 2.414)]
frame = plt.Polygon(frame_coords, linewidth=2, edgecolor='r', facecolor='none')
plotter = plt.Rectangle((0.321, 0.719), 1.50, 1.50, linewidth=2, edgecolor='purple', facecolor='none')
```

Figure 4.3.36: Displaying the exterior frame and plotter

The four corner coordinates of this polygon were declared and parsed into the 'plt.Polygon' function. This produces the shape on the map with a red edge colour, with no fill. Also, it would also be beneficial to show the plotter on the map. Figure 4.3.36 also represents this.

As the system relies on distance measurements from the landmarks, it would be another good addition to represent these distance measurements in the form of circles, as shown in Figure 4.3.37. This could show the user the state of the measurements as each frame is read. If there were sudden changes in the distance measurements, the user will be able to see them.

```
# Estimate position and update the map
def estimate_position(vert_rect_dist=0, hori_rect_dist=0, trapezium_dist=0, octagon_dist=0, circle_dist=0,
                    triangle_dist=0):
    print("Estimating Position")

    hr_circle = plt.Circle((hori_rect[0], hori_rect[1]), hori_rect_dist, edgecolor='b', facecolor='none')
    graph.add_patch(hr_circle)
```

Figure 4.3.37: Displaying circles around landmarks

To show the estimated positions of the sensor within the environment on the map, the values of the trilateration function must be used. The code simply calls this function and uses 'graph.plot' to show this on the map, as shown in Figure 4.3.38.

```
graph.plot(position[0], position[1], 'o')
graph.text(position[0], position[1], 'Position ({} , {})'.format(round(position[0], 3), round(position[1], 3)))
```

Figure 4.3.38: Displaying the estimated position

Notice how the value is rounded to three decimal places. This is to keep in line with the tolerance of the tools used in measuring the coordinates of the landmarks. The final output of this map is shown in Figure 4.3.39 below:

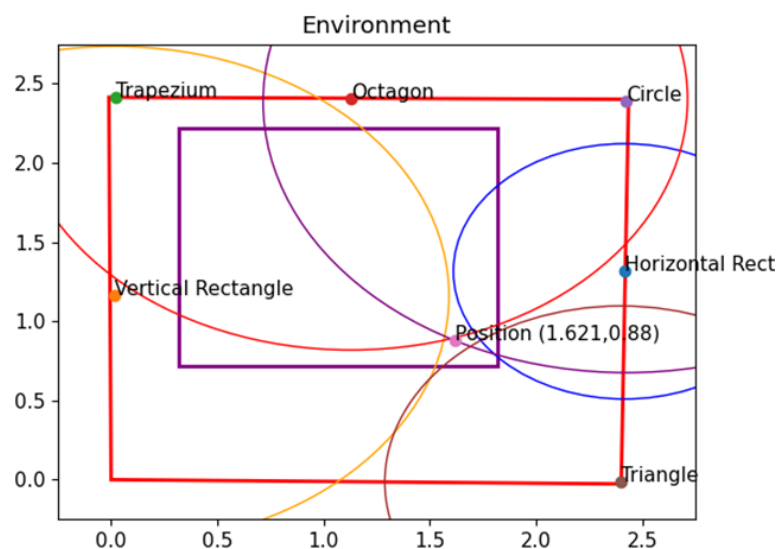


Figure 4.3.39: Real time mapping of the estimated position

As previously mentioned, this map updates when a new frame of the recording is processed, meaning it is updating in almost real time.

This concludes using only the LIDAR to estimate positions within the environment. The LIDAR recordings were run frame by frame through YOLOv5, and the resulting objects had the distances measured from them. Using the distances retrieved in each frame, the shortest

three were utilised in a trilateration algorithm to find an estimated position. This position was then shown on an updating map.

4.3.14 Position Tracking with Live LIDAR Streams

At present, the system is built to take a recording of the LIDAR and estimate position. As a bonus, it is possible to have this algorithm work in real-time. By simply replacing the code in the main algorithm with the block in Figure 4.3.40, it is possible to conduct YOLO and trilateration on a live stream.

Detecting via Recordings	Detecting in Real-Time
<pre>pcap_file = pcap.Pcap(source, metadata)_# Open the .json file with closing(client.Scans(pcap_file)) as scans:_# Open each scan bs = 1_# batch_size for scan in scans:_# Iterate in each scan</pre>	<pre>with closing(client.Scans.stream('os-122110000793.local', 7502, complete=False)) as stream: bs = 1_# batch_size for scan in stream:</pre>

Figure 4.3.40: Changing the code to run in real-time

By running this new algorithm, the same results are produced, with a slight stutter. Due to this stutter, this method will not be tested however it is clearly possible to conduct position estimation in fully real-time through the use of YOLOv5 and a LIDAR sensor.

4.3.15 Utilising Both 360 Camera and LIDAR for Position Estimation

In this section, rather than using just the LIDAR for position estimation, a 360-degree camera can be fused. The LIDAR's output is of a very low resolution, meaning that the object detector is prone to erroneous results in both training and final use. The trainer greatly struggles due to the lack of features in these images; therefore, the camera's higher detail should theoretically provide better results. The plan behind using the camera is that its images will have YOLOv5 running upon it. It has greater resolution, colour depth and granularity than the LIDAR, making it the logical choice for the object detector to run on.

The proposed process is shown in the flow below:

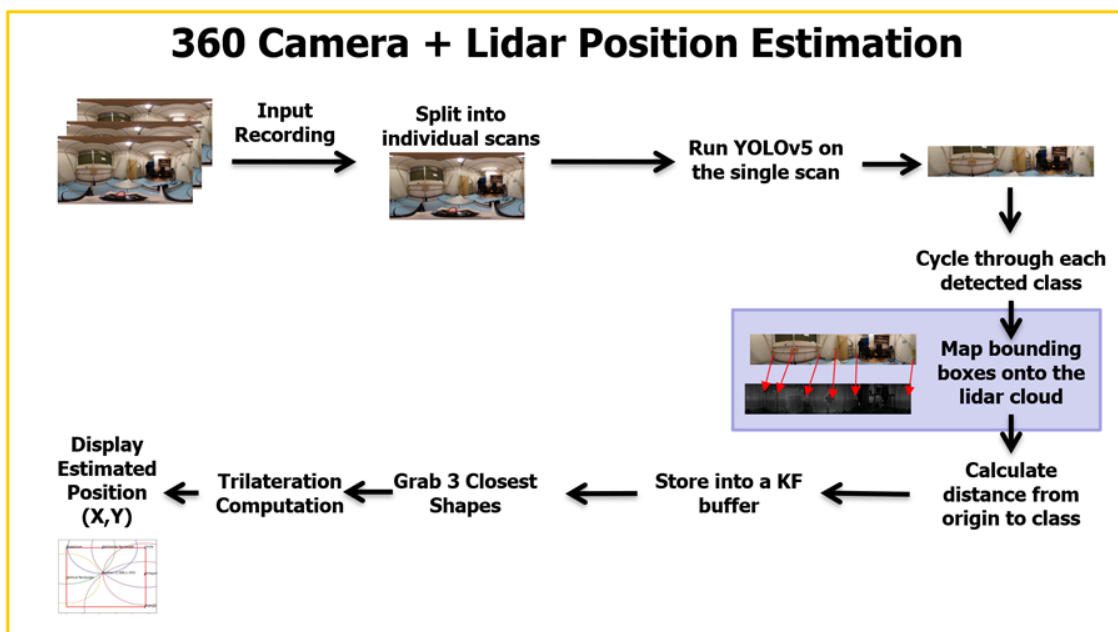


Figure 4.3.41: Camera and LIDAR sensor fusion flow

As described in the Figure 4.3.41, the camera's role is to allow the object detector to have finer detail in identifying objects. This in essence should reduce the number of false detections, and ultimately produce a more accurate position estimation.

4.3.16 Processing YOLOv5 on 360-degree Images

In order to process YOLOv5 on the camera images, it must be retrained again. This results in repeating the process described in section 4.5. This time, the dataset is built off a multitude of images from the camera. The dataset was generated from wirelessly taking images from the camera, using a Python script. By connecting the camera to a wireless network, it is able to be accessed using its IP. The following Figure 4.3.42 shows the key code to wirelessly take images.

```

THETA_ID = 'THETAYL00104791'
THETA_PASSWORD = 'password' # default password. may have been changed
THETA_IP = "http://192.168.50.203/"
THETA_URL = THETA_IP + 'osc/'

def takePictureTester():
    print("taking 10 images")
    for pictureNumber in range(10):
        cameraCommand("takePicture")
        print("processing image " + str(pictureNumber + 1))
        print("please wait 5 seconds")
        time.sleep(5)

```

Figure 4.3.42: Wirelessly capturing images

Using this code, multiple images were taken of the environment from multiple different positions within the environment. After running the trainer until no noticeable gains in accuracy were had, the following recall and precision graphs were generated.

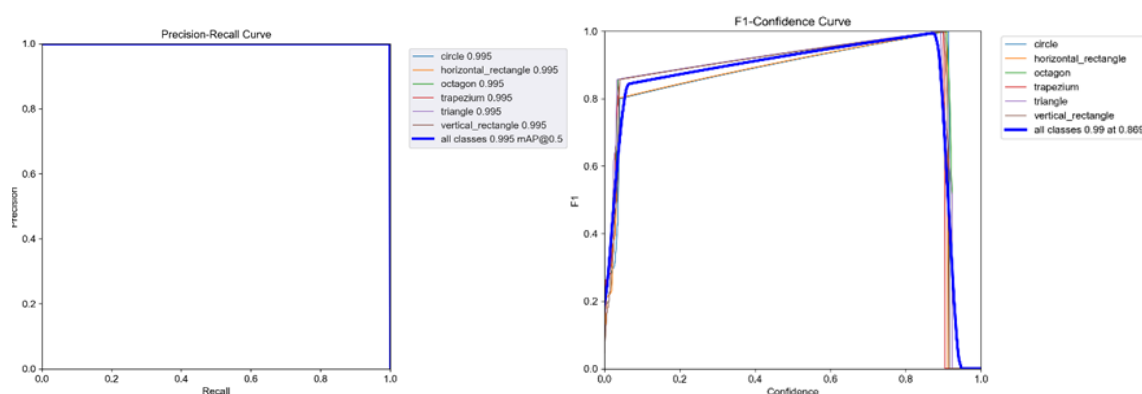


Figure 4.3.43: 360-degree camera P-R and F1-Confidence Curves

As seen by the graphs in Figure 4.3.43, the scores are far higher than on the LIDAR training (0.869), potentially meaning that YOLOv5 is able to better identify the shapes in this medium. By running the object detector on one of the images, the following Figure 4.3.44 is shown:

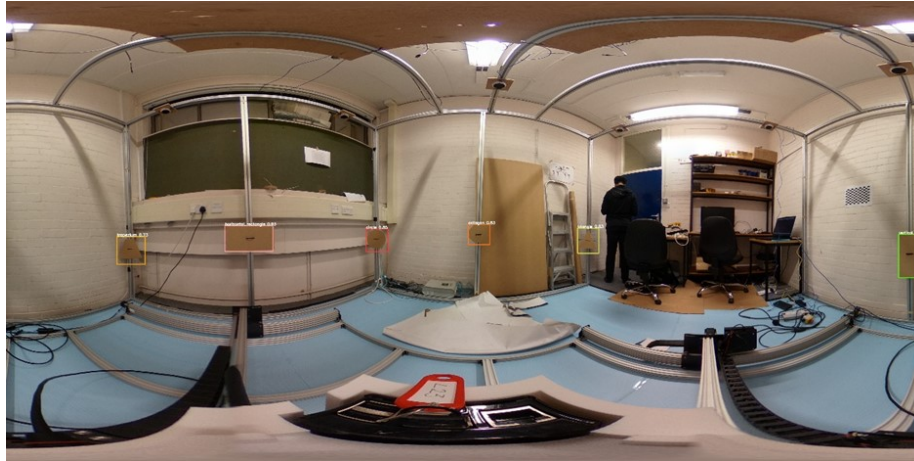


Figure 4.3.44: Running YOLO through the 360-degree camera

It is clear to see that the bounding boxes are firmly in line with the shapes, and their confidence levels are firmly beyond 90%. This backs the notion that the object detector will run far better on higher quality images. As the training is now complete, the proceeding step is to find a way to map the regions of interest onto the point cloud.

4.3.17 Fusing Camera Images with Point Clouds

Since the Ricoh camera that is used to capture the images utilises a dual fish-eye lens, advanced processing would be required in order to directly map it onto the LIDAR image. Furthermore, the camera's aspect ratio does not match with the LIDAR's image. In order to rectify this, the camera image must first be converted into an equirectangular projection. Then it must be cropped and resized to match the resolution of the image. Simply resizing the image will cause an undesirable loss in detail, meaning another method must be utilised. Image processing techniques such as warping or morphing could be a potential method to align the images together.

In essence, fusing the two images is beneficial for greater precision in detecting distances and overall accuracy, at the expense of processing the two images together. This should be further explored in the future.

4.4 Experimental Results and Analysis

This section is dedicated to testing the LIDAR algorithm against the described requirements. The system's performance will also be tested by comparing its position estimations against the actual positions within the environment.

4.4.1 Comparing Position Accuracy

Position accuracy will be tested by comparing the actual x, and y coordinates against the estimated coordinates.

By moving the 3D plotter, the LIDAR will be moved to 16 different positions. Using a single trained weights file, 30 second recordings will be taken at each position. The recording is then parsed through the algorithm and the resulting estimated positions are saved to a file. The actual positions of these 16 positions will be manually measured, as shown in Figure 4.4.1. As the algorithm updates position on every object detection, this length of recording should produce at least 1000 measurements.

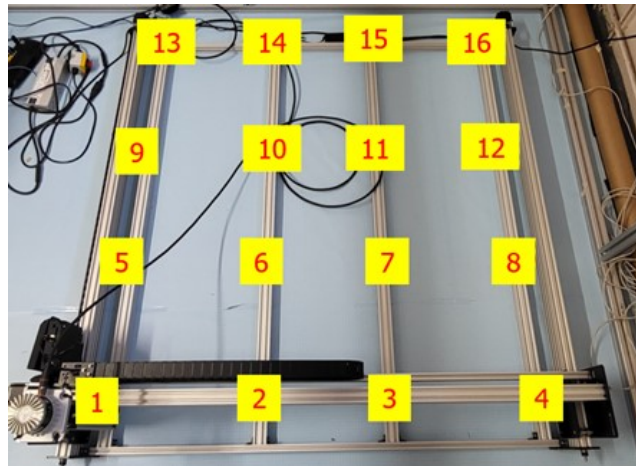


Figure 4.4.1: Recording locations on the 3D plotter

The recorded x and y coordinates from the file are then fed into MATLAB. The code calculates a mean average error between the estimated and actual values, is shown in Figure 4.4.2. The output is then shown in multiple graphs. A snippet of the MATLAB code is shown below.

```
% Calculating Error with real measurements
abs_diff_x (m)= (abs(x_actual - x(m)));

% Calculating Error with mean values
abs_mean_diff_x (m)= (abs(x_actual - x_mean(m)));
avg_perc_mean_diff_x(m) = mean(abs_mean_diff_x);

% Calculating Error with KF values
abs_diff_KF_x (m)= (abs(x_actual - xn1_0));
avg_diff_KF_x(m) = mean(abs_diff_KF_x);
```

Figure 4.4.2: Code calculates a mean average error between estimated & actual values

This code reads the coordinate values and passes them through a KF. It also calculates the mean average of all points. For each position on the plotter, the following actual positions were found. Note that the origin is kept the same as Figure 4.3.33. Actual distances for each position on the 3D Plotter is shown in Table 4.4.1.

Table 4.4.1: Actual distances for each position on the 3D Plotter

Plotter Position	Coordinates (x,y)
1	(1.608,0.814)
2	(1.608,1.240)
3	(1.608,1.662)
4	(1.608,2.084)
5	(1.176,0.813)
6	(1.196,1.242)
7	(1.196,1.666)
8	(1.196,2.090)
9	(0.784,0.814)
10	(0.784,1.242)
11	(0.784,1.666)
12	(0.784,2.090)
13	(0.337,0.814)
14	(0.345,1.242)
15	(0.337,1.666)
16	(0.337,2.090)

The estimated x and y graphs as well as their errors for positions 1, 11, and 16 are show in appendix IV-1, IV-2, and IV-3 respectively. To allow for a fair comparison, the number of data points have been capped at 1000.

Each position’s mean error against the actual position will be shown on the following error map in Figure 4.4.3.

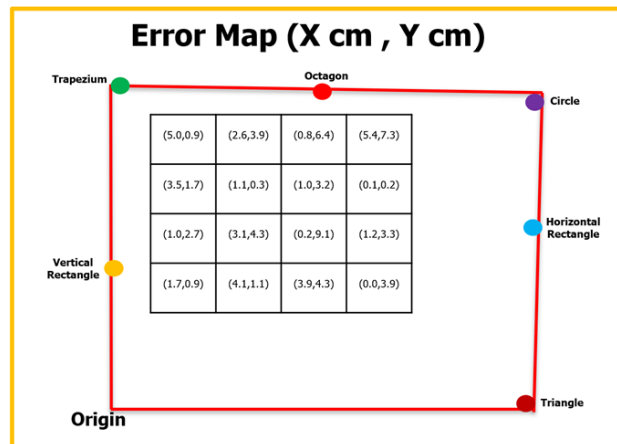


Figure 4.4.3: Estimated Position Error map

By examining each sector in the error map, the ranges heavily vary. The x coordinate varies by up to 5.4cm and y by up to 9.1cm. This large discrepancy could be due to multiple reasons such as poor object tracking, or poor actual measurements. A percentage error map is present in appendix E. This will be further discussed in the limitations section.

4.5 Comparing against the Requirements

As described in section 4.3.2, the functional requirements are at the core of this project. Therefore, the developed system must be assessed in accordance with it. The Table 4.5.1 below tests whether the system has met these specifications. The tests were performed against the recordings taken in section 4.4.1

Table 4.5.1: Testing the developed system against the Functional Requirements

Functional Requirements	Testing Methodology	Evidence	Outcome
The LIDAR system should correctly detect and map the surrounding environment.	Each recording was checked on whether it displayed correctly.	The LIDAR is able to take a point cloud of the complete environment.	Pass
The system should automatically identify landmarks within the environment.	Each recording must have had at least 3 detected landmarks.	The system was producing at least three detections in each recording.	Pass
The system must be able to determine distance from landmarks.	The system was tasked to produce distances for position estimation.	Figure 4.6.2.3 Shows the object detector correctly calculating and displaying distance.	Pass

Functional Requirements	Testing Methodology	Evidence	Outcome
The system must estimate position in real-time.	Each recording was required to produce estimated positions.	The system is able to estimate position in real-time. It was able to do so with both recordings and live data.	Pass
The position estimation must work in the event a number of landmarks are blocked.	A new recording was created, where three positions were blocked.	Figure 4.6.1.4 shows that YOLO is able to run even with certain shapes blocked. As the algorithm only requires 3 shapes, position estimate can still occur.	Pass
The landmarks should be standalone, and not require any continuous setup.	Landmarks were checked on whether they needed extra setup once deployed.	The landmarks in Figure 4.2.1.1 were static objects, with no electronic maintenance required.	Pass

Although the system meets the functional requirements, it must also be assessed against the KPIs described in Table 4.3.2.

4.6 Testing Against the KPIs

In order to test the system against the KPIs, the system must be heavily utilised in the processes described in section 4.3.2. Tests were conducted against the recordings developed in section 4.4.1.

Table 4.6.1: Testing the developed system against the KPIs

Key Performance Indicator	Testing Methodology	Evidence
Distance measurements should be within 5% of actual distance.	The estimated distances from Position 6 were compared against actual distances.	Testing shows that the measured distances to shapes were within 5%.
Object detection should occur at least once per frame.	The software is capable to write detections to file.	For each recording, the detections file listed at least 300 frames.
The object detector should have a false positive rate of less than 5%.	Each recording was visually checked for false positives against total detections.	The percentage for false positives came to 11% out of all detections across all recordings.
Distance measurements should occur at least once per frame.	The software is capable to write distances to file. The number of rows should equal the number of frames.	For each recording, the output file detected files exceeding 300 rows. It equals the number of frames.

Key Performance Indicator	Testing Methodology	Evidence
Position estimation should be within 5% error in both x and y coordinates.	Comparing the error map produced in appendix E.	Appendix E shows that out of 32 estimations (both x and y), the percentage error exceeds 5% on 5 occasions (15.63%).

The testing of KPIs show that the system is able meet most indicators although not to the requested standards.

4.7 System Limitations

Overall, the system is able to complete its task of finding the position of the sensor within the environment. However, there are multiple limitations restricted this project as a whole:

- The high variance between actual and estimated positions as seen on the error map.
- The LIDAR's low-resolution output is not detailed enough for YOLOv5 to run perfectly on.
- It is unsure where on the LIDAR distance measurements are being from.
- The exterior frame's non-uniformity could cause errors in determining the coordinates of the landmarks.
- The processing power required for YOLO may be too high for day-to-day use.
- The test environment is an almost ideal situation for landmarks as they are quite close.
- The trilateration algorithm is only for a two-dimensional space, which has limited applications.

The varying error in estimated positions could be due to multiple factors. The measuring of actual position may have been done with an inaccurate method, in addition to not knowing where the LIDAR is taking measurements from. It could be assumed that the LIDAR is taking measurements from the edge, which this project has assumed. Or it could be from the centre of the LIDAR, meaning that a four-centimetre offset must be included to all distance measurements. Moreover, the fact that frame on which the landmarks are attached to is not uniform, meant that the process of measuring the coordinates of the landmarks was far from straightforward. In addition to the difficulty measuring, it could have also led to erroneous coordinates, ultimately resulting to poor estimation results.

Speaking in regard to YOLO itself, the object detector is accelerated using a graphics card in this project. In situations where there is no such card, processing of position may be far too delayed to be practical. Furthermore, the LIDAR's output already struggles within this testing environment; it is beginning to struggle in distinguishing detailed shapes such as the octagon from distances greater than 0.5m, causing a loss in detection or even a false detection as the circle. In events where multiples of the same class are detected, the localisation algorithm has the potential to produce erroneous results. This is where the camera would immediately improve the system due to its greater image quality and resolution.

5 Data combination and data selection for SLAM improvement

5.1 Aim of experiment

Utilizing the optical spectrum is considered a promising strategy for advancing future networks, enabling themselves to accommodate high-density and high-capacity connectivity requirements. However, there is also an elevated risk of localization failures or degraded performance when User Equipments (UEs) to localize are in motion in the considered area. This is due to the possibility of not receiving certain signals, leading to data missing.

To address the issue of data missing in the localization field, solutions can be categorized into three groups: 1) Map-based localization which is limited by the time-consuming and labour-intensive calibration. It can be applied only when the map of the indoor environment and the locations of Access Points (APs) are known in advance. 2) Data completion for localization can be proposed aiming to complete missing data based on the noisy partially known received data. However, estimating unknown signal parameters can cause a degradation of the localization accuracy if the generation is based on a small amount of noisy known received parameters. 3) Data combination for localization which is based on combining data from different sources or different technologies. Three levels of combination exist (see Figure 5.1.1) including: i) Parameters fusion which cannot be a good solution for the data missing problem. Since, once the signal is not received, different parameters are unknown. ii) Sensors' fusion which is applied only when we have data from different sensors with various natures. iii) Technologies fusion integrating measurements from different communication technologies. To boost Optical Wireles Communication (OWC) based localization performance, WiFi is proposed in this work due to its low cost without need to add extra infrastructure. If we apply the classical combination method for localization considering OWC data and WiFi data, we can obtain higher localization error compared to the classical OWC localization system. This can be done to the fact that WiFi signals can be heavily affected by noise and multi-path.

In this work, we propose a two-step localization system to improve the performance of the data fusion method. Our approach is distinctive in its dual phase execution: an offline phase and a subsequent online phase. The main contributions of this study are:

- A selection process is integrated in the localization system based on classification using a Deep Neural Network (DNN) model, trained offline. Such model applied online is going to take the Received Signal Strength (RSS) received from both considered technologies and the output will be class 1 or class 2. If class 1, we will apply localization based on OWC only and if class 2, we will apply localization based on technologies combination. For the authors' best knowledge, it is the first time that a hybrid localization model based on technology-based data selection is conducted.
- The proposed system is validated based on simulated data generated using realistic propagation models considering two environmental configurations and varying simulation parameters. Obtained performance are compared to OWC-based system and fusion-based system.

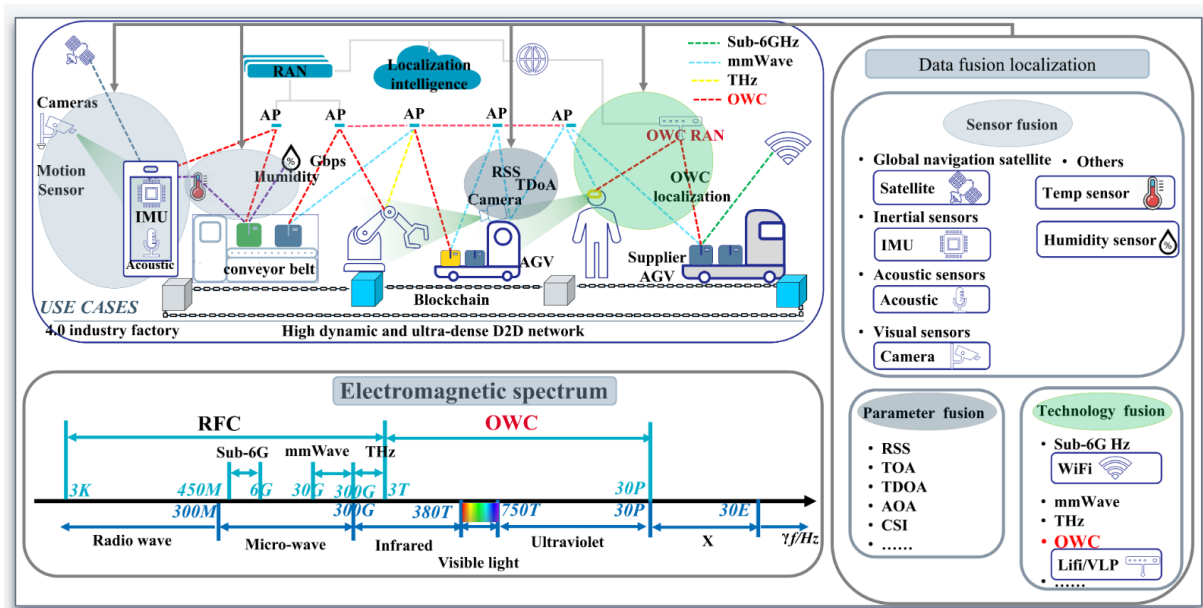


Figure 5.1.1: Data combination for localization.

5.2 Hybrid localization system using WiFi (2.4GHz) and OWC

An innovative hybrid localization system is necessary to improve the fusion localization accuracy. The key to the hybrid system is to choose which Neural Network (NN) model to use for location estimation depending on the considered technology whether we apply OWC - NN or fusion - NN. Therefore, we rise the selection process using NN-based classifier. Different steps of the proposed hybrid localization system are shown in Figure 5.2.1. During the offline phase, RSS data is collected at T UEs received from N OWC APs and M WiFi APs. These data associated to the corresponding coordinates are organized to construct the training database of size $(T, N + M + 2)$. Each fingerprint is composed of $(N + M)$ RSS values and the 2D corresponding coordinates. During the offline phase, three NNs (i.e., Selector - NN, OWC - NN and fusion - NN) are built and optimized to be used online.

- OWC - NN:** This model, used for localization based on OWC signals, is trained, and validated based on OWC data. To build it, we consider a database of size $(T_r, N + 2)$ and for validation $(T_v, N + 2)$, where $T = T_r + T_v$. T is the total number of UEs positions, T_r is the number of training positions and T_v is the number of validation positions. The inputs are the RSS vectors, and the outputs are the 2D corresponding coordinates.
- Fusion - NN:** This NN is used for localization based on combined data. During this model training, we consider $(T_r, M + N)$ RSS as inputs and its corresponding $(T_r, 2)$ coordinates as outputs. For validation, the remaining T_v measurements are considered.
- Selector - NN:** Once both NN models used for localization are built, a labeling step is going to be performed. In fact, for each UE, we compare the obtained localization errors using OWC only and combining it with WiFi. This comparison is performed calculating the difference between the real coordinates and the estimated coordinates based on the used localization NN. If the OWC - NN is better, a label = 1 is assigned to the corresponding fingerprint. Else, a label = 2 is assigned. At the end of this step, two

classes are formed: a first class with label 1 which contains the sub-set of data having better localization results with OWC data and a second class with label 2 which results are better when combining OWC and WiFi data. Then, the selection - NN, which is formed as a binary classification problem, is trained based on OWC - RSS combined with WiFi - RSS as inputs and the class label (1 or 2) as output. In the online localization process, when a UE receives raw data that contains OWC - RSS and WiFi - RSS, the selection process is applied at first using the trained Selector NN in order to predict the class label. If the estimated label is equal to 1, OWC - NN is applied and otherwise, fusion - NN is performed. Eventually, the estimated localization is obtained.

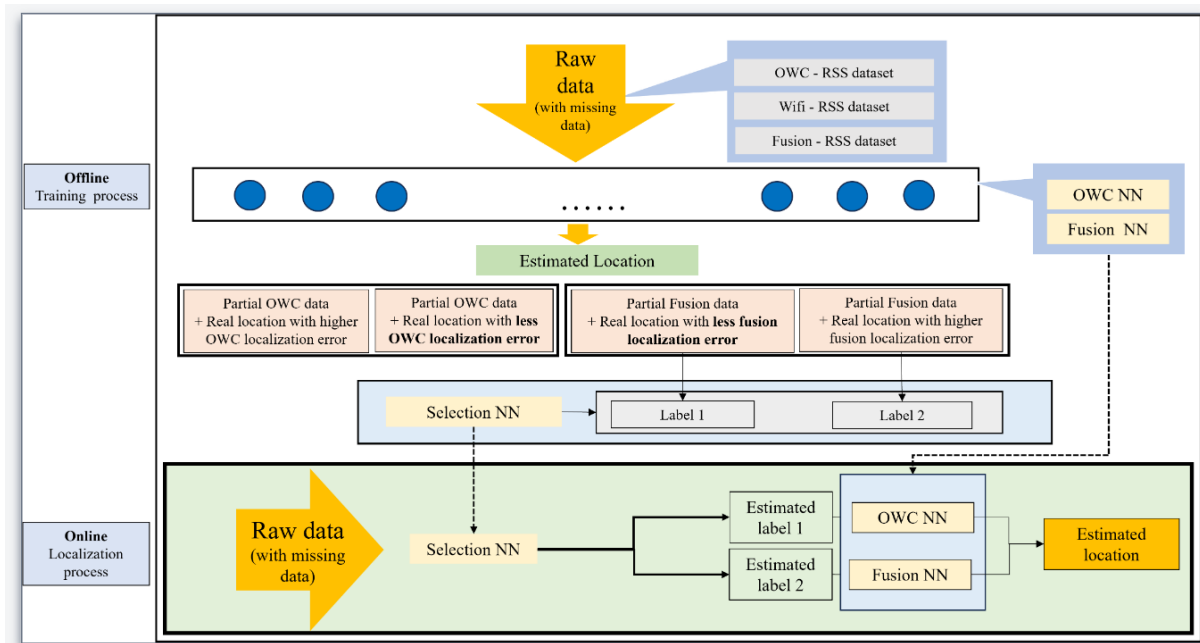


Figure 5.2.1: Architecture of the proposed hybrid localization system.

5.3 Simulation results and analysis

Two environmental configurations ($L \times W \times H$) are settled to evaluate the localization performance of the proposed system. The first one is the small room with the size of $L = 5\text{ m}$, $W = 5\text{ m}$ and $H = 5\text{ m}$. The second one is the big room with the size of $L = 20\text{ m}$, $W = 20\text{ m}$ and $H = 10\text{ m}$. We consider $N = 4$ OWC APs and $M = 10$ WiFi APs for both configurations.

To simulate the data missing phenomena, the weakest signals in a certain proportion (eg. 20% or 50%) are set to 0. Besides, considering that different noise cases can exhibit different localization accuracy, we simulate noise scenarios including a noise-free case and Signal-to Noise Ratio (SNR) of 40/50 in OWC transmission, with sigma shadowing of 2 dBm and 5 dBm in WiFi transmission cases as well. In summary, three parameters (room configuration, data missing and noise proportion) are varied in this work to verify the fusion localization performance in different use cases under different parameters values to generalize the obtained results. Localization tasks are assisted by NNs. To generate the offline database, 1156 UEs locations are considered, and 5 measurements are taken at each location to minimize the effect of RSS variations. Therefore, $T = 5780$ fingerprints are constructed and each one is constructed by 16 fields (4 OWC - RSS, 10 WiFi - RSS and 2D coordinates). 70% data are considered as T_r training data and the T_v rest is used for validation or test. Hyperparameters

and architectures of different used DNNs have been optimized based on corresponding input data via an exhaustive simulation process.

At first, we compare the localization performance of the OWC-based localization (i.e., a DNN is applied to localize targets based on OWC - RSS only) and Fusion-based Localization (i.e., a DNN is applied to localize targets based on fusion - RSS). Localization results obtained with both environmental configurations varying the noise value and the percentage of missing data are shown in Table 5.3.1: Obtained Localization Errors Considering Both Environmental Configurations.. We can easily notice that no matter the room configuration, data missing proportion and noise value, mean localization error using fusion-based localization method is always less than the one using OWC signals only. This means that fusion method does generally make effective. However, when WiFi signals become noisier (σ shadowing = 5), performance associated to fusion-based method are heavily affected reaching only 1 % and 0.51 % of improvements compared to OWC-based method when considering the first room. This is since WiFi signals are noisy and cannot serve efficiently for localization. Also, if we check the percentage of UE positions where location estimation is better when using only OWC signals, we notice that we have a significant percentage where the signals' fusion increases the localization error. Thus, the fusion methodology is insufficient, and a hybrid localization system based on the selection is proposed to choose OWC- based localization or fusion-based localization.

Table 5.3.1: Obtained Localization Errors Considering Both Environmental Configurations.

Room configuration	Room configuration I						Room configuration II					
	20%			50%			20%			50%		
Data missing proportion												
Shadowing value	no	2	5	no	2	5	no	2	5	no	2	5
SNR value	no	40	50	no	40	50	no	40	50	no	40	50
OWC-based localization mean error (m)	0.012	1.562	0.906	0.201	1.550	0.914	0.139	5.878	4.649	0.661	5.906	4.668
Fusion-based localization mean error (m)	0.010	0.608	0.894	0.182	0.711	0.910	0.089	1.882	4.577	0.541	1.908	4.512
Fusion mean error decrease Vs OWC mean error (%)	25%	61%	1%	9%	54%	0.51%	35%	68%	2%	18%	68%	3%
Percentage of locations where OWC localization is better (%)	40%	15%	4%	66%	21%	70%	40%	13%	69%	75%	13%	68%

To evaluate the performance of our hybrid proposed localization system, simulations are conducted with the same OWC transmission and WiFi transmission parameters. We present results when considering the second environmental configuration with SNR equal to 50 in OWC transmission and σ shadowing equal to 5 dBm in WiFi transmission. We present only this scenario to simplify the presentation, however, interpretations and conclusions still available for other scenarios. In Table 5.3.2, we compare the localization performance for four schemes:

- **OWC-based localization system** using OWC - NN.
- **Fusion-based localization system** using fusion – NN.
- **Classical selection-based scheme:** this is the ideal selection process. Using this selection, for each UE to localize, OWC-based localization and Fusion-based localization are applied. Then, both predicted positions are compared to real coordinates and the best estimation is held. This method has two disadvantages: i) the two DNNs are applied, and the comparison process is held online which increases the online computation complexity. ii) to proceed the comparison, we need to know the

real coordinates. However, in reality, these coordinates are not known. Thus, this scheme is used as unrealistic benchmark which we want to reach using the proposed selection.

- **Proposed hybrid system based on the selection process:** Such system is based on the use of three different NNs: OWC NN, Fusion NN and Selector NN.

Table 5.3.2: Comparison of localization errors using different approaches considering the second configuration with SNR = 50 and sigma shadowing = 5.

Data missing proportion		20%			50%		
Evaluation parameters	Mean localization error (m)	Localization mean error decrease Vs OWC (%)	Percentage of locations with higher error Vs OWC (%)	Mean localization error (m)	Localization mean error decrease Vs OWC (%)	Percentage of locations with higher error Vs OWC (%)	
OWC-based localization	4.65	–	–	4.67	–	–	
Fusion-based localization	4.58	1.54%	69.11%	4.51	3.34%	68.31%	
Classical selection based localization	3.92	15.81%	0%	3.91	16.22%	0%	
Proposed hybrid localization system	4.29	7.68%	51.46%	4.29	8.02%	53.21%	

Data missing proportion		20%			50%		
Evaluation parameters	Mean localization error (m)	Localization mean error decrease Vs OWC (%)	Percentage of locations with higher error Vs OWC (%)	Mean localization error (m)	Localization mean error decrease Vs OWC (%)	Percentage of locations with higher error Vs OWC (%)	
OWC-based localization	4.65	–	–	4.67	–	–	
Fusion-based localization	4.58	1.54%	69.11%	4.51	3.34%	68.31%	
Classical selection based localization	3.92	15.81%	0%	3.91	16.22%	0%	
Proposed hybrid localization system	4.29	7.68%	51.46%	4.29	8.02%	53.21%	

To verify that the classical selection-based system is working correctly, we check the percentage of UE locations where OWC-based localization is better than this selection scheme in Table 5.3.2: Comparison of localization errors using different approaches considering the second configuration with SNR = 50 and sigma shadowing = 5.. An ideal selection is conducted as a control group without the classification error. Using the NN selector, we aim to improve fusion-based localization performance to minimize the percentage of targets where OWC only outperforms the fusion estimations. We focus on the mean localization error of different localization methods. It turns out that when data missing proportion is 20%, the mean localization error decreases by 7.68% and 1.54% compared to the one of OWC localization concerning the proposed hybrid scheme and the fusion scheme, respectively. This improvement can also be observed in the percentage of locations where OWC localization outperforms others. This percentage decreases from 69.11% when using fusion to 51.46% when using the NN-based selection. This verifies the out-performance of the proposed localization system. Results obtained with 50% of missing data confirm the ones obtained with 20%. Consequently, our proposed model improves the localization accuracy of the OWC-based localization and the fusion-based localization by integrating the decision step which does not add significant extra online complexity since we just apply a model totally trained offline.

6 Federated Learning (FL)-based localization

6.1 Aim of experiments

VLC and FL together offer a powerful and innovative approach for indoor localization. VLC utilizes visible light as a communication medium, leveraging existing lighting infrastructure to transmit data at high speeds, while also providing illumination. This unique dual-purpose capability makes VLC an attractive solution for indoor positioning. On the other hand, Federated Learning ensures privacy and data security by enabling on-device model training and updates, without sharing raw data externally. The combination of VLC and FL addresses the challenges of indoor localization, offering high-speed data transmission, improved positioning accuracy, and enhanced privacy protection.

Advantages of VLC over other wireless communication technologies

- **Utilization of Unregulated Visible Light Spectrum:** One of the major advantages of VLC lies in its utilization of the visible light spectrum, which is unregulated and free to use. Unlike radio frequencies, which are often subject to interference and congestion due to the proliferation of wireless devices, VLC leverages light waves to transmit data. As a result, it operates on a relatively unused portion of the electromagnetic spectrum, minimizing the risk of interference and providing a reliable means of communication.
- **Enhanced Security and Privacy:** VLC offers an inherent advantage in terms of security and privacy compared to traditional wireless communication technologies. The nature of visible light propagation restricts its coverage to the line of sight, making it challenging for signals to penetrate through walls or other solid objects. Consequently, VLC transmissions are less susceptible to unauthorized interception, eavesdropping, or signal snooping, enhancing data security and ensuring private communication channels.
- **Impregnable Physical Layer Security:** Incorporating VLC in wireless communication systems adds an extra layer of security, as eavesdroppers attempting to intercept the transmission from outside the communication area are rendered ineffective. The limited range of visible light ensures that unauthorized individuals cannot access the data being transmitted without physically entering the illuminated zone, making VLC an attractive option for secure communication in sensitive environments.
- **Energy-Efficiency Through Low Power Consumption:** VLC's reliance on Light Emitting Diodes (LEDs) as the light source contributes to its remarkable energy efficiency. LEDs require significantly lower power to operate compared to traditional wireless communication systems. This characteristic is particularly advantageous in scenarios where energy conservation is crucial, such as in battery-powered devices or energy-constrained environments. VLC's low energy consumption not only reduces operational costs but also aligns with the growing demand for sustainable and eco-friendly technological solutions.

Challenges of VLC and its Mitigation through Machine Learning Techniques

It is important to note that VLC-based localization in dynamic indoor environments can be challenging.

One of the primary hurdles is the fluctuating position of the transmitters, which emit the visible light signals to establish communication and determine the location of the receiving device. In dynamic indoor settings, the transmitters may be subject to relocation, alterations,

or reorientation to accommodate changes in the layout or functions of the indoor space. Consequently, this variability in transmitter positions introduces uncertainties and complexities, leading to suboptimal localization outcomes.

Moreover, the presence of obstacles within the indoor environment worsens the localization challenges. As visible light signals traverse through the space, they encounter various obstacles, such as walls, furniture, and other objects. These obstacles create multipath effects, wherein the signals reflect, refract, or scatter, giving rise to multiple signal paths to reach the receiver. The multipath phenomenon can lead to signal interference and propagation abnormalities, contributing to inaccuracies in the localization process.

The presence of these challenges lead to high localization errors in terms of accuracy, reliability, and scalability. Accuracy refers to how closely the estimated location aligns with the actual position of the target device, while reliability relates to the consistency and dependability of localization results across different instances. Additionally, the scalability of the localization system is crucial for its effectiveness in large or complex highly dynamic indoor environments. To overcome these limitations and enhance the efficiency of VLC-based localization in dynamic indoor environments, we need machine learning techniques for data-based localization to mitigate this problem. By adopting machine learning techniques, we can significantly improve the performance of VLC-based localization.

Adapting Federated Learning approach

With the rapid advancements in communication technologies and integration of digital services and industries, the volume of data generated within indoor environments has surged exponentially. Sending such vast amounts of data to a central server poses several challenges and concerns. Firstly, the centralized approach raises data security and privacy risks, as sensitive information may be vulnerable to breaches or unauthorized access during transmission and storage. Additionally, the sheer volume of data being transmitted over the network imposes significant strain on available bandwidth, potentially leading to network congestion and degraded performance.

To address these critical issues, the adoption of the Federated Learning approach becomes increasingly appealing. By leveraging Federated Learning, data remains localized and confined within each device, thereby significantly reducing the exposure of sensitive information to potential threats. This local processing and model training also enable reduced data transmission, resulting in lower latency and alleviating the bandwidth burden on the communication infrastructure.

Moreover, the implementation of Federated Learning aligns well with the emerging emphasis on energy efficiency and resource optimization. Centralized Learning necessitates continuous data transmission to the central server, consuming substantial energy resources in the process. On the other hand, Federated Learning's decentralized nature reduces the need for frequent data transmissions, leading to energy savings and improved sustainability. These advantages make FL a compelling and promising approach to address the challenges posed by dynamic indoor environments and pave the way for more robust and resilient indoor positioning solutions in our increasingly connected digital world.

6.2 Centralized learning and Federated learning background

Machine learning techniques play a vital role in improving the performance of indoor localization models. Among them, two distinct approaches stand out: **Centralized Learning** and **Federated Learning**. These techniques offer unique advantages and cater to different privacy and scalability requirements in the context of indoor localization.

- In **Centralized learning**, all the data is collected and stored in a central server, and the model is trained on this server. So basically, this central server is responsible for aggregating the data from different sources, performing the training, and then send the trained model to the participating devices. Centralized learning is best suited for situations where the data is centralized and the computational resources are available on the central server. However, localization methods based on centralized learning require collecting data from IoT devices into a central server/unit, resulting in a lot of data exchange with the server (time-consuming, resource-intensive), privacy concerns, and a high reliance on the server. This requires high bandwidth and assumes the server to be trustworthy. ED is a short form of End Device.
- **Federated Learning** addresses the privacy and communication challenges of centralized approaches. It enables model training directly on the devices themselves without sharing data with a central server. Each device performs local model training using its own data, and only model updates (weights) are sent to the central server. The central server aggregates these updates to create a global model, which is then redistributed to the devices for further refinement in an iterative process. A system model of a typical federated learning network is shown in Figure 6.2.1. In the context of indoor localization, the choice between these techniques depends on the specific requirements of the indoor localization system and the level of data privacy and communication overhead allowed by the deployment scenario.

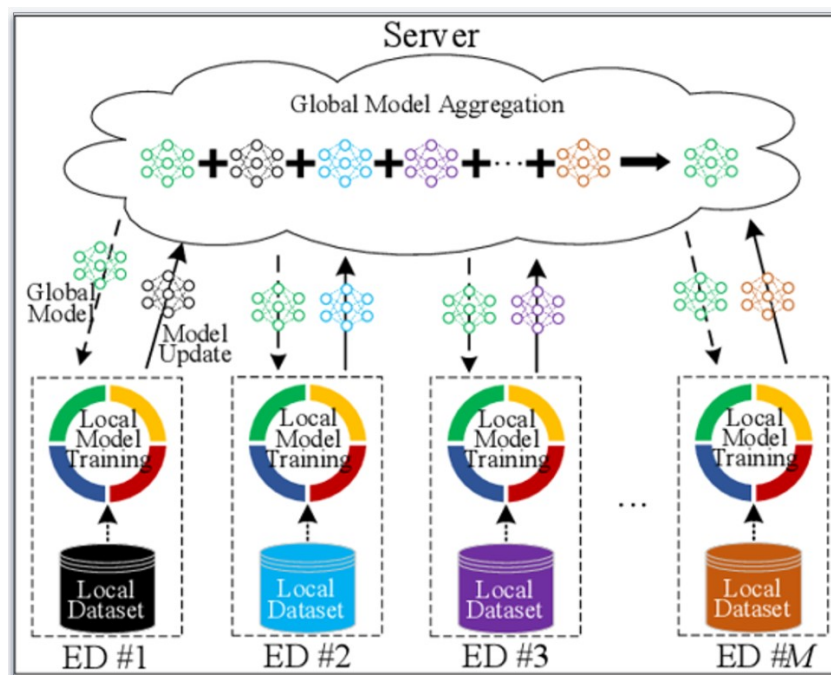


Figure 6.2.1: Illustrations of system model of a typical federated learning network.

6.3 Use case and description of FL framework

Main contributions

The main focus of this work is to first design a hierarchical learning scheme for the joint building, floor, and precise 2D coordinates prediction in multi-building and multi-floor indoor environments. Then, regarding the resource constraints of IoT devices, we propose a federated learning framework to train the proposed hierarchical model, yielding a communication efficient collaborative and privacy-preserving indoor localization solution. The main contributions of this work can be summarized as follows:

- We develop a novel 3D indoor positioning system with a new DNN architecture incorporating the hierarchical nature of indoor localization tasks within multi-building and multi-floor indoor environments. Our approach involves a single hierarchical DNN model that accurately predicts the buildings, floors, and precise 2D coordinates of users simultaneously. To validate our solution and ensure its practicality for indoor IoT applications, we rigorously tested it using a publicly available experimental indoor localization dataset.
- We propose a federated training of the proposed architecture to preserve IoT devices location data privacy and save the bandwidth of the wireless infrastructure. Consequently, we provide a collaborative, bandwidth optimization, and privacy-preserving indoor localization solution for IoT applications.
- Being aware of the exponential growth of IoT networks, we investigate the scalability of the proposed FL framework and we provide an analysis of different wireless transmission

Different steps of the proposed framework

In a multi-building and multi-floor indoor environment, the position of the target can be recursively obtained in a hierarchical manner starting with the building identification followed by the floor identification, and finally the fine-grained location of the target. Thus, we can expect to determine the position of the target with more precision. Let's consider a smart city which have 3 buildings (B0, B1, B2). Each building has 4 floors (F0, F1, F2, F3). Each building has its own set of Wi-Fi access points (APs).

The goal is to implement Federated Learning to train a central model for predicting the longitude and latitude coordinates inside buildings based on Received Signal Strength Indicator (RSSI) data from multiple buildings. **Please be aware that, in order to maintain clarity, due to the absence of readily accessible VLC public data, we will employ Wi-Fi RSSI data for the training of our model.** Once measurements collected at Bosch Germany will be pre-processed, we will proceed to test it. Subsequently, we will apply our model to the VLC data to assess its effectiveness, identifying both its strengths and limitations. Figure 6.3.1 illustrates a hierarchical representation of our federated learning process that outlines the process.

- **Data Collection:** Data will be collected from multiple buildings in a smart city. Each building (B0, B1, B2) has four floors (F0, F1, F2, F3). Each floor is treated as a user in the FL setup, and their data will be used to train floor models.
- **Training Local Models (Floor Models):** Each floor from each building trains its own floor model using its local Wi-Fi APs and the associated RSSI data. This is done independently on each floor without sharing raw data with the central server. Each

floor model aims to predict the longitude and latitude coordinates of locations inside its corresponding building.

- **Aggregating Floor Models to Regional Models (Building Models):** After the floor models are trained, the weights of these models are aggregated to create building-level models, referred to as regional models. For each building (B0, B1, B2), the floor models' weights are combined to create a regional model that represents the collective knowledge of that building.
- **FL Central Model (Global Model):** The regional models from the three buildings (B0, B1, B2) are further aggregated to create the FL Central Model. This central model will be the main model that predicts the longitude and latitude coordinates of locations inside all the buildings in the smart city.
- **Monitoring and Validation:** Key Performance Indicators (KPIs) are monitored and validated using the training results of the FL Central Model. This step is essential to ensure the model's performance is up to the desired standards.
- **Hyperparameter Tuning:** If needed, the model hyperparameters like learning rate, the number of layers, and floor selection (data quality, computation, etc.) can be adjusted to optimize the FL Central Model's performance.
- **Sharing Weights with Regional and Local Floor Models:** Once the FL Central Model is trained and evaluated, its updated weights are shared back with the regional models. This allows each building to benefit from the knowledge learned across the smart city without sharing sensitive data centrally. Additionally, the local floor models can also receive updates from the FL Central Model, enhancing their prediction capabilities.
- **Iterations:** The process of aggregating and sharing weights between floor models, regional models, and the FL Central Model will go through multiple rounds of training and updating to continually improve their performance.

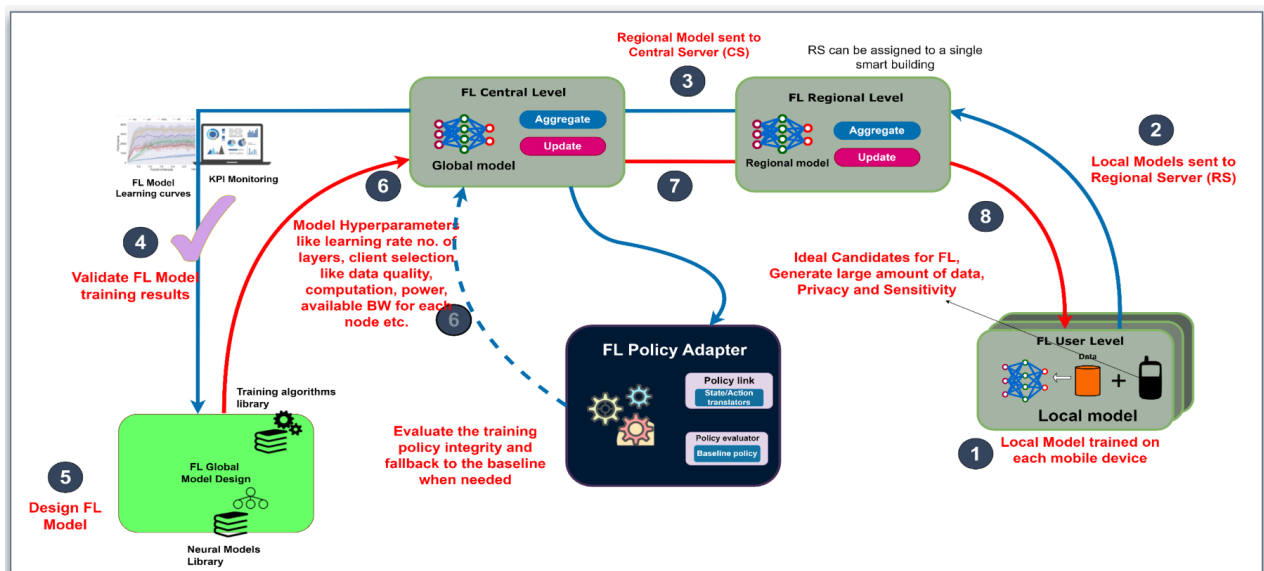


Figure 6.3.1: Hierarchy of federated learning infrastructure for location Estimation.

6.4 Validation of the proposed framework based on a publicly available database

For performance evaluation, we will train our model proposed FL model based on the publicly available UJIndoorLoc database which covers almost 110 000 m². This database is multi-floor and multi-building with 3 buildings and 4/5 floor per building.

To optimize different used DNN models during training, we employ a customized error function that aligns with the learning objectives and dataset characteristics. The error function used in our federated learning setup is the Mean Absolute Error (MAE). Different models' configuration which will be applied is depicted in Table 6.4.1. Note that for fair comparison, we will use the same configuration for our FL as well as for the Centralised Learning (CL) model. And Simulation settings for our Federated Learning approach are provided in Table 6.4.2.

Table 6.4.1: DNN Models' Configuration

Floor model		Building model		Central model	
Hidden layers	256 – 64	Hidden layers	256 – 64	Hidden layers	256 – 64
Dropout layer	0.25 – 0.1	Dropout layer	0.25 – 0.1	Dropout layer	0.25 – 0.1
Input activation	ReLU	Input activation	ReLU	Input activation	ReLU
Output activation	Linear	Output activation	Linear	Output activation	Linear

Table 6.4.2: FL simulation settings

Parameter	Value
Model optimizer	Adam
Learning rate	0.0005
Exponential decay rates	0.1, 0.99
Number of Users (Floors)	12
Batch size	32
Number of epochs per local iteration	10
Number of epochs per central iteration	1000
Communication rounds	100

In Figure 6.4.1, we show the training and validation performance of the FL learning and the CL. In fact, we present the variation of the localization error in meters as a function of the number of epochs. The training localization error is equal to 5.32 m for the CL model and 5.45 m for the FL. The obtained validation error reached after convergence is almost equal to 10.8 m for both models. In Table 6.4.3, we compare the mean localization error of three different localization schemes (FL-based localization, CL-based localization and K-Nearest Neighbors (KNN) for localization). Based on this table, we can approve results obtained in Figure 6.4.1 and we can also mention the outperformance of the NN-based localization models compared to the conventional KNN model.

We easily notice that using the FL approach, we can maintain the same localization performance as the CL approach improving the aspects of data privacy, scalability and the adaptability of the localization model. This can cause some increase on the training complexity, however, this process is offline and does not impact the required real-time response for localization systems even in large sensor networks.

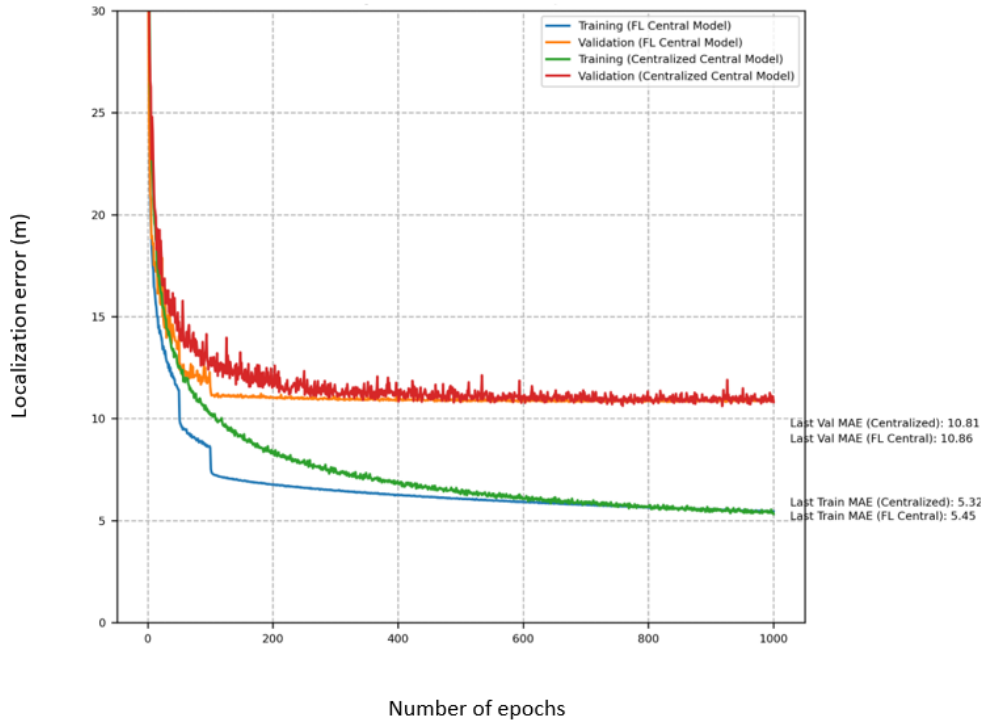


Figure 6.4.1: Training performance Vs Validation performance.

Table 6.4.3: Localization error comparison

Approach	Localization Error (Meters)
KNN	12.18
Centralized Model	10.81
FL Central Model	10.86

7 Novel Beacon Positioning Signal Design To Unlock the Strict Time Synchronization

7.1 System Model and Positioning Beacon

With the widespread application of light-emitting diode (LED)-based illuminating and the rapid development of visible light communication systems, visible light positioning (VLP) technology has become a promising solution for indoor positioning systems [7-1] and [7-2]. Different from the traditional indoor positioning technologies, such as wireless fidelity (Wi-Fi), ultrasonic, Bluetooth, and ultra-wideband (UWB), the VLP system can obtain high accuracy positioning with densely distributed LEDs and license-free spectrum resource.

Numerous studies have been devoted to enhancing positioning performance based on advanced modulation, synchronization, and positioning schemes. These works can efficiently enhance positioning accuracy in the case of rich hardware resources. Specifically, a high sampling rate is usually required to sample the input signal to avoid the aliasing effect, which leads to increased implementation complexity and power consumption. In other words, the performance of the VLP system depends heavily on the wide bandwidth of the LED-based transmitter and a high sampling rate of the photodetector (PD)-based receiver [7-3] and [7-4]. However, there are many resource-limited scenarios, especially in the Internet-of-Things (IoT) networks, the VLP system in these scenarios should meet the requirements of low hardware cost and power consumption. The authors in [7-5] and [7-6] pointed out that the direct use of low-cost PD-based receivers at low sampling rates results in distortion effects. Moreover, due to the large junction capacitance and package inductance of LED, the bandwidth of LED is generally less than 10 MHz. Thus, it is meaningful to investigate a suitable positioning scheme to alleviate both the bandwidth and sampling rate requirements in the resource-limited VLP systems.

To solve this issue, an indoor beacon construction and low sampling rate positioning scheme is proposed based on on-off keying (OOK) modulation pulse pairs. Different from the traditional time of arrival (TOA)-based positioning scheme, this positioning scheme does not require stringent time synchronization. Moreover, the proposed scheme achieves high positioning performance with a low sampling rate and low bandwidth. Specifically, inspired by the ease of implementing both communication and illumination services with small modulation bandwidth, the OOK modulation is exploited in the considered VLP systems. Then, the OOK-based pulse pairs are proposed to design the positioning beacon signal without the synchronization requirement. Moreover, considering the fact that the sampling rate is usually low in the practical resource-limited scenario, a high-precision pulse reconstruction method is introduced to eliminate the distortion effect resulting from the low real sampling rate and to alleviate the positioning errors caused by the time measurement errors. Note that this work has been published in International Conference on Indoor Positioning and Indoor Navigation (IPIN) 2023 and achieved 4th best paper award [7-7].

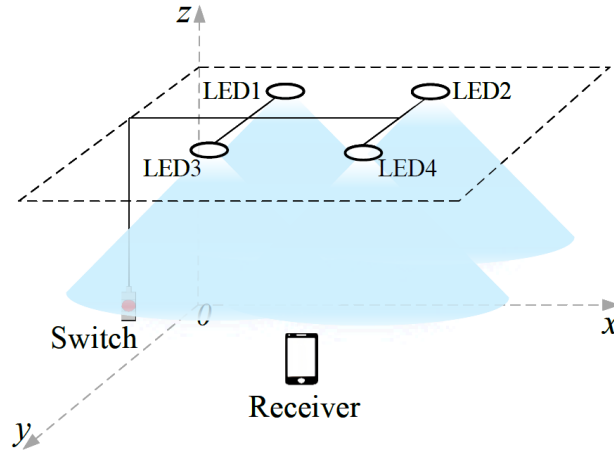


Figure 7.1.1: The VLP system model.

In a typical indoor scenario, even number of lamps are commonly installed to fulfil uniform illumination. Thus, as shown in Figure 7.1.1, four LEDs are uniformly deployed and are powered and controlled by a unified power grid in the VLP system. In order to facilitate deployment with lower costs, a general-purpose single bipolar timer is installed in each LED to control the bias current and then to achieve the OOK modulation. Except for the unified power wiring, there are no other connections between the four LEDs, and there are no synchronization requirements among these timers. Benefiting from these settings, the proposed positioning scheme can be easily applied to any VLP system with more than four LEDs.

The proposed positioning scheme first performs OOK modulation on the bias current of the i -th LED and forms a positioning beacon signal $x_i[t]$. Then, the received signal from k -th pulse pair of the i -th LED to location S is defined as

$$y_{i,s}(k) = \sqrt{P_i} \times h_{i,s} \times x_i[\phi_i(k) + \tau_{i,s}] + n_0,$$

where P_i is the radiation power of the i -th LED, $h_{i,s}$ is the channel gain between the i -th LED and the s -th PD. Moreover, $\phi_i(k)$ is the radiation time of the k -th pulse pair at the i -th LED, $\tau_{i,s}$ is the transmission time of the coded pulse pair between the i -th LED and the s -th PD. Channel noise n_0 obeys the Gaussian distribution.

Since the channel gain is inversely proportional to the square of the distance, the non-line-of-sight (NLoS) gain is much weaker than the line-of-sight (LoS) gain. Without loss of generality, it is safe to assume that the LoS path dominates the channel gain. Moreover, it is assumed that the LED radiation model follows the Lambertian model. Therefore, the channel gain $h_{i,s}$ is expressed as follow.

$$h_{i,s} = \frac{(m+1)A_{\text{PD}}}{2\pi d_{i,s}^2} \cos^m(\phi) \cos(\varphi) G(\varphi) T(\varphi),$$

where $d_{i,s}$ is the distance between the i -th LED (x_i, y_i, z_i) and the s -th PD (x_s, y_s, z_s) . m is the Lambertian radiation order, A_{PD} is the detection area of PD. $G(\varphi)$ and $T(\varphi)$ are the

gains of optical filters (OF) and optical concentrators (OC), respectively. Moreover, ϕ and φ are the radiation angle and incident angle between the LED and PD. It is assumed that the normals of LED and PD are parallel to each other. Thus, given the channel gain $h_{i,0}$ and distance $d_{i,0}$ from the i -th LED to the reference position (x_0, y_0, z_0) , the relationship between the path channel gain $h_{i,s}$ and distance $d_{i,s}$ from the i -th LED signal to s -th PD is expressed as

$$h_{i,s} = h_{i,0} \left(\frac{d_{i,0}}{d_{i,s}} \right)^{m+3}.$$

Based on above equation, the signal $y_{i,s}(k)$ received from the emission pulse at time $\phi_i(k)$ follows the Gaussian distribution, and it is given by

$$p(y_{i,s}(k) | \phi_i(k)) = N\left(h_{i,s} \sqrt{P_i w_{s,\phi_i}(k) T_w}, \sigma^2\right),$$

where the Gaussian distribution with mean μ and variance σ^2 is denoted by $N(\mu, \sigma^2)$. For the mean item in the above equation, $w_{s,\phi_i}(k)$ quantifies the ratio of the overlap time $\Delta t_{i,s}(k)$ between the expected pulse observation window and the actual observation window to the pulse width T_w , and it is calculated as

$$w_{s,\phi_i}(k) = \begin{cases} \frac{\Delta t_{i,s}(k)}{T_w}, & \Delta t_{i,s}(k) > 0 \\ 0 & \Delta t_{i,s}(k) \leq 0, \end{cases}$$

$$\Delta t_{i,s}(k) = T_w - \left| \phi_i(k) + \frac{d_{i,s}}{c} - t_{i,s}(k) \right|,$$

where c is the light speed, $t_{i,s}(k)$ is the arrival time of the k -th pulse pair, $|a|$ represents the absolute value of a .

In the low-cost or resource-constrained VLP system, the bandwidth of LED is limited. Thus, the OOK is chosen to cope with this challenge. However, the low switching rate and response time result in a slow rise speed of the OOK modulation pulse. In other words, the traditional OOK modulation-based positioning scheme is difficult to obtain high-precision performance. Therefore, a novel OOK-based coded pulse pair is proposed to inherit the bandwidth-cost advantage of OOK and alleviate the limitation of OOK. The main idea behind this beacon is to modulate the bias current of each LED according to the predefined requirements, and then control the intensity of each LED to obtain the pulse pair between LEDs.

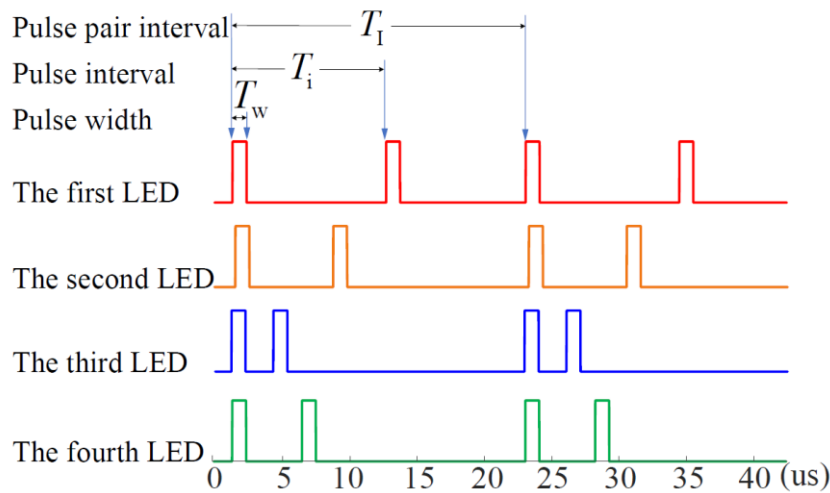


Figure 7.1.2: Example of positioning beacon design with the four LEDs.

The example of the positioning beacon design profile with four LEDs is shown in Figure 7.1.2. It can be seen that each LED has an equal pulse width T_w and a pulse pair interval T_I ($T_I = NT_w$, N is any positive integer). The guideline of this positioning beacon and practical choice of the interval parameters are given as follows. Firstly, in order to facilitate the receiver to distinguish the beacons of different LED, the pulse interval T_i of each LED is required to be different from that of other LEDs, and any two pulse intervals T_i and T_j are mutually prime. Moreover, the pulse interval T_i is much larger than $(\tau_{t,\max} + \tau_{e,\max})$ to avoid overlap between pulses, where $\tau_{t,\max}$ and $\tau_{e,\max}$ denote the maximum transmission delay in the considered indoor space and the upper limit of time error, respectively. Furthermore, the pulse pair interval T_I is greater than any pulse interval T_i to ensure that all beacon signals can be transmitted in a pulse pair interval T_I .

7.2 Pulse Reconstruction and Position Estimation

Due to the strong dependence between the positioning performance and the sampling rate, it is important to provide a high sampling rate for achieving high accuracy positioning. However, a high sampling rate in the receiver significantly increases the cost and power consumption of the receiver, while the hardware resources and energy in the receiver are usually limited. Therefore, in order to maintain the low cost and low power consumption in resource-limited VLP systems, a high-precision pulse reconstruction method is proposed. Specifically, based on the sampling rate conversion method, the approximation of a high sampling rate pulse is obtained from a low sampling rate pulse sequence, and then a maximum a posteriori (MAP)-based position estimation method is proposed.

This subsection introduces the high-precision pulse reconstruction method based on the rational factor sampling rate conversion, which can reconstruct discrete-time signals under ideal conditions and resample continuous-time signals at different sampling rates. For any continuous-time signal $x(t)$, a discrete-time signal $x(nT)$ can be obtained by the sampling

rate of $f_x = 1/T_x$. Then, based on the interpolation idea, $x(nT)$ can be converted to a continuous-time signal $y(t)$, and $y(t)$ is defined as

$$y(t) = \sum_{n=-\infty}^{\infty} x(nT_x)g(t - nT_x),$$

where $g(t - nT_x)$ is the reconstruction function. However, $y(t)$ is difficult to reconstruct the original $x(t)$ in a practical scenario. In practice, the reconstruction signal $y(mT_y)$ with a high sampling rate of $f_y = 1/T_y$ is given as

$$y(mT_y) = \sum_{n=-\infty}^{\infty} x(nT_x)g(mT_y - nT_x),$$

where nT_x and mT_y correspond to the input and output time of the signal, respectively. The above equation represents a linear time-invariant system in the case of the sampling rate $f_x = f_y$. On the contrary, when $f_x \neq f_y$, it is re-formulated by changing the sum index variable from n to $k = k_m - n$ and is given by

$$y(mT_y) = \sum_{k=-\infty}^{\infty} g(kT_x + \Delta_m T_x)x((k_m - k)T_x),$$

where $k_m = \lfloor mT_y / T_x \rfloor$ and $\Delta_m = mT_y / T_x - k_m$, the $\lfloor \cdot \rfloor$ denotes floor function.

The key in the reconstruction process is to determine the corresponding value of $g(n)$. Thus, in order to facilitate this reconstruction process, the ratio of $T_y / T_x = D / I$ is limited as a rational number. Then, Δ_m has only I unique values. The corresponding $g_m(nT_x)$ has I different value sets, and it has a period of m , namely,

$$g_m(nT_x) = g_{m+rI}(nT_x), \quad r = 0, \pm 1, \pm 2, \dots$$

In order to preserve the spectral characteristics of the original signal as much as possible, the proposed high-precision pulse reconstruction is conducted by first interpolating and then sampling. The details of the proposed high-precision pulse reconstruction structure are shown in Figure 7.2.1. Specifically, $I - 1$ zeros are firstly inserted between adjacent sample points of $x(n)$ to achieve the upsampler with factor I . Then, the filter is utilized to remove the unnecessary frequency components in the $X(w)$. Moreover, the signal $y(m)$ is extracted from the $w(l)$ using the downsampler with factor D . Thus, for the input signal $x(n)$ with sampling rate f_x , the sampling rate of the reconstructed signal $y(m)$ is $(If_x) / D$.

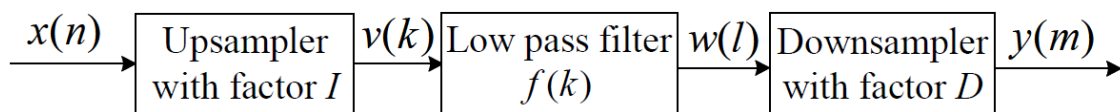


Figure 7.2.1: The proposed high-precision pulse reconstruction structure.

Moreover, the frequency response $F(\omega_v)$ of the low-pass filter must balance the effects of up-sampling and down-sampling. Thus, $F(\omega_v)$ is given by

$$F(\omega_v) = \begin{cases} I, & 0 \leq |\omega_v| \leq \min(\pi / D, \pi / I) \\ 0, & \text{otherwise.} \end{cases}$$

Finally, the spectrum of the output sequence $y(m)$ can be obtained and it is written as

$$Y(\omega_y) = \begin{cases} \frac{I}{D} X\left(\frac{\omega_y}{D}\right), & 0 \leq |\omega_y| \leq \min\left(\pi, \frac{\pi D}{I}\right) \\ 0, & \text{otherwise.} \end{cases}$$

Where $X(\omega)$ is the spectrum response of the input signal $x(n)$. Then, the MAP-based position estimation is presented as follows.

MAP is a probability estimation method based on the Bayesian theorem, which is used to estimate the most likely value of an unknown random variable under the condition of given observations. The goal of MAP estimation is to find the parameter value with the maximum posterior probability, where the posterior probability refers to the conditional probability of the parameter value in the case of a given observation [7-8]. Based on the MAP detector, the optimal location \hat{s}_{MAP} for the s -th PD estimated by the proposed scheme is given by

$$\begin{aligned} \hat{s}_{\text{MAP}} &= \arg \max_s p(s | y_{\text{re}}) \\ &\stackrel{(a)}{=} \arg \max_s p(s | y_{\text{re},1}, y_{\text{re},2}, y_{\text{re},3}, y_{\text{re},4}) \\ &= \arg \max_s \prod_{i=1}^4 \left[\int_{\Phi_i} \frac{p(y_{\text{re},i} | \phi_i) p(\phi_i, s)}{p(y_{\text{re},i})} d\phi_i \right], \end{aligned}$$

where y_{re} is the reconstructed high sample rate signal and $y_{\text{re},i}$ is the reconstructed signal corresponding to the i -th LED. The process (a) holds since the y_{re} is composed of $y_{\text{re},i}$. Moreover, Φ_i is the set of all possible radiation times ϕ_i of the visible light pulse emitted by the i -th LED. Since the measured time error and the terminal position distribution are independent and the random variables of both the measured time error and terminal position obey the uniform distribution, the joint probability $p(\phi_i, s)$ in the above equation is rewritten as

$$p(\phi_i, s) = p(\phi_i) p(s) = \frac{1}{|\Phi_i|} \times \frac{1}{|\mathbf{S}|},$$

where $|\Phi_i|$ and $|\mathbf{S}|$ are the potentials of the measurement time value set and the terminal position value set, respectively.

In the time domain, the positioning method described in the equation of \hat{s}_{MAP} mainly includes three types of time errors. Firstly, the time error among the emission time of the pulse pair at

each LED is caused by the difference in the arrival time of the pulse source to the LED. Note that this time error remains unchanged when the cable layout is determined. The second type of time error results from the difference in the response of each LED. Fortunately, the property difference of LEDs in the same scenario is small to achieve better illumination quality, so its time error can be ignored. Finally, there is a random error in the arrival time of the pulse estimated by the terminal, it is caused by the link noise.

The first two errors determine the centre value offset of the radiation time set Φ_i in each LED. The third error determines the divergence degree of radiation time Φ_i . Moreover, since the pulse pairs from each LED to the same receiver are affected by the same link noise environment, it is reasonable to assume that the radiation time set Φ_i of the four LEDs estimated by the receiver has the same characteristic in the proposed scheme. Thus, the joint probability of time error of each LED and s -th PD in the above equation are same. Then, substituting the above equation into the equation of \hat{s}_{MAP} , it can be further simplified as

$$\hat{s}_{\text{MAP}} = \arg \max_s \prod_{i=1}^4 \left[\int_{\Phi_i} p(y_{\text{re},i} | \phi_i) d\phi_i \right].$$

In order to reduce the computational complexity, the logarithmic operation is applied to the above equation, and the mean value is calculated. Then, based on the equations of channel gain and ratio of the overlap time, the final positioning optimization problem is reformulated as

$$\begin{aligned} P_0 : \min_s \sum_{k=1}^K \sum_{i=1}^4 & \left| y_{\text{re},i}(k) - h_{i,s} \sqrt{P_i w_{\phi_i,s}(k) T_w} \right|, \\ \text{s.t. } C1 : c & \left| t_{i,0}(k) - \phi_i(k) \right| - d_{i,0} < \varepsilon, \end{aligned}$$

where tiny constant ε is the preset positioning error threshold. Moreover, $t_{i,0}(k)$ is the arrival time of the k -th pulse pair observed by the receiver at the reference position. Then, the solving process for problem P_0 is introduced in the following.

Given the position error threshold ε , pulse width T_w , pulse interval T_I , LED coordinate (x_i, y_i, z_i) , LED radiation power P_i , reference position coordinate (x_0, y_0, z_0) , and time domain search progress factor α is given in the considering VLP system. The distance $d_{i,0}$ can be obtained. Meantime, for the fixed terminal at the reference position, the pulse signal is detected and the corresponding number of pulse pairs K is counted. Then, the distinguishable pulse $y_{i,0}(k)$ and pulse arrival time $t_{i,0}(k)$ of the i -th LED in the k -th pulse pair interval are measured. Thus, the reconstructed signal $y_{\text{re},i,0}(k)$ is obtained by using the proposed pulse reconstruction. Moreover, the channel gain $h_{i,0}$ and the radiation time $\phi_i(0)$ of 0-th pulse pair are calculated as:

$$h_{i,0} = \frac{1}{K \times \sqrt{P_i}} \sum_{k=0}^{K-1} y_{re,i,0}(k),$$

$$\phi_i(0) = \frac{1}{K} \sum_{k=0}^{K-1} (t_{i,0}(k) - \frac{d_{i,0}}{c} - kT_1).$$

Secondly, the time error is calculated as

$$\delta^2 = \frac{1}{K} \sum_{k=0}^{K-1} [t_{i,0}(k) - (\frac{d_{i,0}}{c} + \phi_i(k))]^2.$$

The radiation time $\phi_i(k)$ of k -th pulse pair is calculated as

$$\phi_i(k) = \phi_i(0) + k \times T_1.$$

Considering the fact that the probability that the value is distributed in $(\mu - 3\sigma, \mu + 3\sigma)$ is 99.74%, the i -th LED possible radiation time set Φ_i is calculated as

$$\Phi_i = \left\{ \phi_i(k) - 3\delta : \frac{\alpha\mathcal{E}}{c} : \phi_i(k) + 3\delta \right\}.$$

Based on the given Φ_i , the likelihood ratio \mathcal{Y} is calculated as

$$\mathcal{Y} = \sum_{k=1}^K \sum_{i=1}^4 \left| y_{re,i}(k) - \sqrt{P \times w_{\phi_i}^s(k) \times T_w} \times h_i^s \right|.$$

Finally, when the \mathcal{Y} is found, the optimal estimates of the radiation time $\hat{\phi}_i(k)$ and position \hat{s} of \mathcal{Y} are obtained.

7.3 Laboratory Experimental Testbed

An experimental testbed is constructed to verify the effectiveness of the proposed positioning scheme in the typical $5 \times 5 \times 3 \text{ m}^3$ scenario. Based on the mutually prime principle and the maximum transmission delay, the pulse intervals T_i of LED1-LED4 are set as $11 \mu\text{s}$, $7 \mu\text{s}$, $3 \mu\text{s}$, and $5 \mu\text{s}$, respectively. Figure 7.3.1 depicts the structure of the semi-physical experimental testbed and the detail of the transceiver front end. Specifically, the proposed OOK-based beacon generation is realized by the field programmable gate arrays (FPGA) board (ZYNQ7000-XC7Z035). An oscilloscope with a low sampling rate of 100 MHz is exploited to measure the signal received by PD. Then, the measured data is processed with MATLAB.

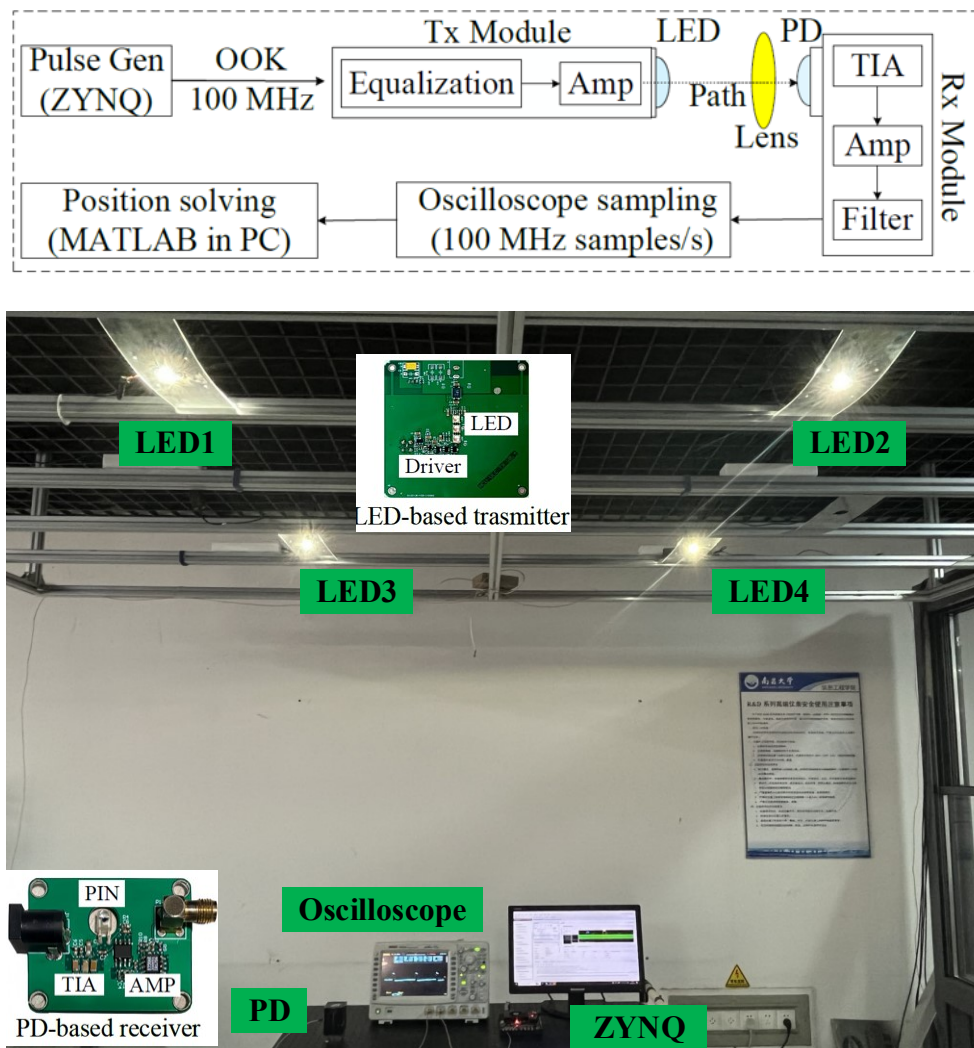


Figure 7.3.1: The structure of the semi-physical experimental testbed.

7.4 Experimental results and analysis

Firstly, the characteristic of positioning beacon is analyzed. Figure 7.4.1 shows the received waveform at the reference point. It can be seen that the highest pulse consists of the first pulse of the four LED pulse pairs. In other words, although the propagation time error of different LEDs results in the width of the first received pulse being wider than the other pulses, the first pulse of the four LED pulse pairs arrives at the reference point almost simultaneously. Thus, it is beneficial for the receiver to extract the arrival time of beacon. Moreover, the high level of each pulse has numerous and non-negligible glitches. This reveals that high-precision pulse reconstruction is very urgent in the resource-limited scenario with low sampling rate.

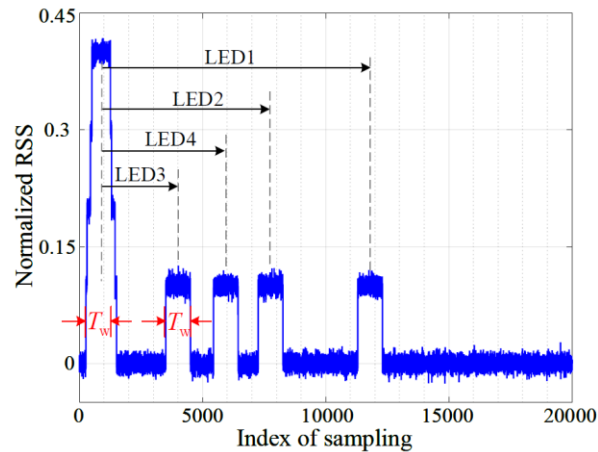


Figure 7.4.1: The received beacon waveform at the center point.

Secondly, the performance comparison is conducted to further verify the superiority of the proposed positioning scheme. The following results are obtained by analyzing the data measured from PD at random positions in MATLAB platform. Specifically, Figure 7.4.2 depicts the cumulative distribution function (CDF) related to positioning error with different sampling rate. Note that the transmit signal-to-noise ratio (SNR) is set as 30 dB and the red curve is reconstructed from 100 MHz to 2 GHz. It can be observed that the CDF achieved by positioning with pulse reconstruction significantly outperforms that obtained by positioning without pulse reconstruction. Moreover, the positioning performance achieved by reconstruction-based 2 GHz pulse is close to that obtained by the real 2 GHz sampling pulse. The result confirms the effectiveness of the high-precision pulse reconstruction method.

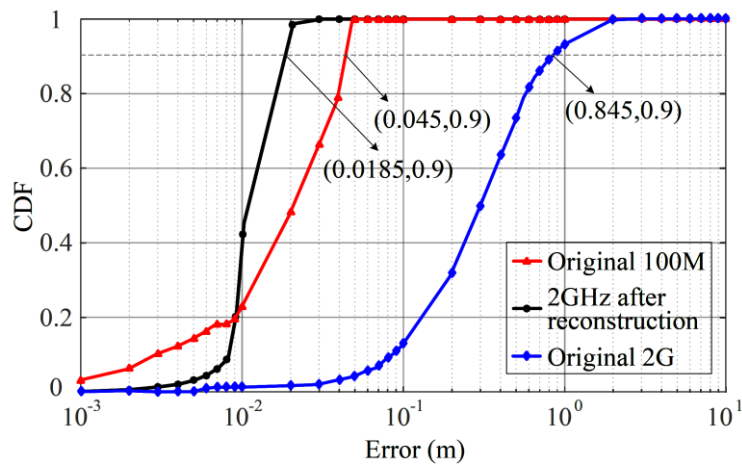


Figure 7.4.2: The CDF related to the location estimation error with and without high-precision pulse reconstruction.

Then, the positioning performance under different conditions is evaluated. Specifically, Figure 7.4.3(a) shows the CDF of positioning error achieved by the proposed positioning scheme versus the bandwidth. Note that the transmit signal-to-noise ratio (SNR) is set as 30 dB and the reconstructed sampling rate is 2 GHz. It can be observed that positioning performance in 90% positioning error corresponding to 50 MHz, 30 MHz, and 10 MHz bandwidth of LED is 4.5cm, 9.2cm, and 25.7cm, respectively. In other words, the positioning performance decreases with the decreasing bandwidth. This is attributed to the fact that the LED with

higher bandwidth can provide a flatter response and thus the corresponding beacon waveform is closer to the rectangle waveform. It is worth emphasizing that the positioning accuracy remains within 30cm even if the bandwidth of LED is generally less than 10 MHz. Therefore, the proposed positioning scheme achieves high-precision positioning under low bandwidth.

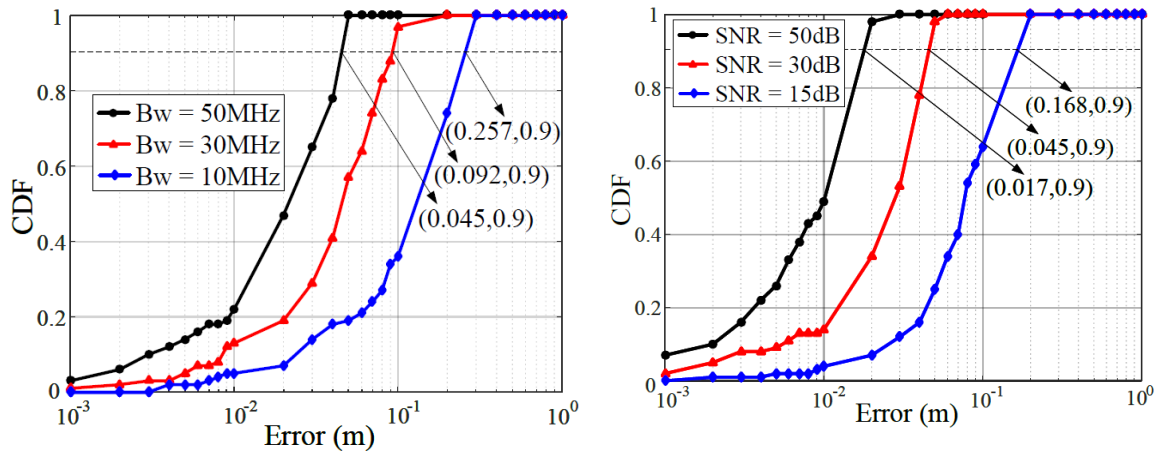


Figure 7.4.3(a) The CDF versus the transmit SNR. (b) The CDF versus the transmit SNR.

Figure 7.4.3(b) illustrates the CDF of positioning error achieved by the proposed positioning scheme versus the transmit SNR. Note that the bandwidth of LED is set as 50 MHz and the reconstructed sampling rate is 2 GHz. It can be seen that positioning performance increases with the increasing modulation of transmit SNR. Specifically, positioning performance corresponding to 50 dB, 30 dB, and 15 dB transmit SNR is 1.7 cm, 4.5 cm, and 16.8 cm, respectively. The experimental results further demonstrate that the proposed positioning scheme achieves high positioning performance even under low sampling rates and low bandwidth conditions.

8 Secure Mutual Localisation Simulation

8.1 Aim of Experiment

High accuracy localisation¹ schemes are highly dependent on continuous Line of Sight (LoS) access between at least three position-calibrated anchor nodes and the user equipment (UE) for which positions need to be determined. There are scenarios where three-anchor LoS cannot be guaranteed at all times. In these situations, other techniques should be considered, such as dead reckoning [8-1], using an inertial measurement unit (IMU) [8-2], different measurements (angle & distance) from fewer anchors, or *iterative trilateration* using neighbouring UEs that act as anchor nodes once they have determined their position beforehand [8-3].

In this experiment, we explored the feasibility of iterative trilateration based on a light-weight secure messaging scheme. For a multi-device environment where a UE's identity is unknown to neighbouring UEs a trust framework is required that excludes malicious UEs from the collective localisation process. We propose a *Control Point* based approach where UEs need to present proof of their identity, thus labelling them as trustworthy and initializing them to collaborate on the localisation effort using key-pair signed messages for both localisation information exchange as well as propagating membership changes within the UE node network.

The aim of the simulation is to demonstrate the feasibility of the messaging scheme for different setups of number of UEs, spatial scenarios with varying levels of obstructedness, wireless communication latencies, and different communication ranges.

8.2 Background Theory

8.2.1 Abstract Scenario

The messaging scheme presented here is designed to be applied to scenarios where devices move around a controlled area, e.g. a factory floor or a lab campus. The devices need to determine their positions periodically. For the positioning effort, they either communicate with base stations that have a calibrated position, or with each other, to leverage other devices knowledge of their own position.

Communication between nodes (referred to as “agents” from here onwards) must be secured such that only messages from trusted partners (other agents) should be accepted, i.e. information from malicious nodes should be discarded.

Participation in the communication network (i.e. entrance to the controlled area) is controlled by a so-called “Control Point”. Before a device is allowed to enter the area, it has to present its identity/origin to the Control Point in a secure manner - an approach for this identity proof presentation has been described within 6GBRAINS deliverable 6.2 [8-4] in the form of Verifiable Credentials, leveraging Hyperledger projects Indy and Aries. This initial communication between entering devices and the Control Point is “out-of-band” from the

¹ Note: The term *localisation* is used synonymously to *positioning* in this context, not in the *language localisation* sense.

perspective of the messaging scheme presented here. Once the device has entered the area, it only communicates via the proposed messaging scheme. See Figure 8.2.1.

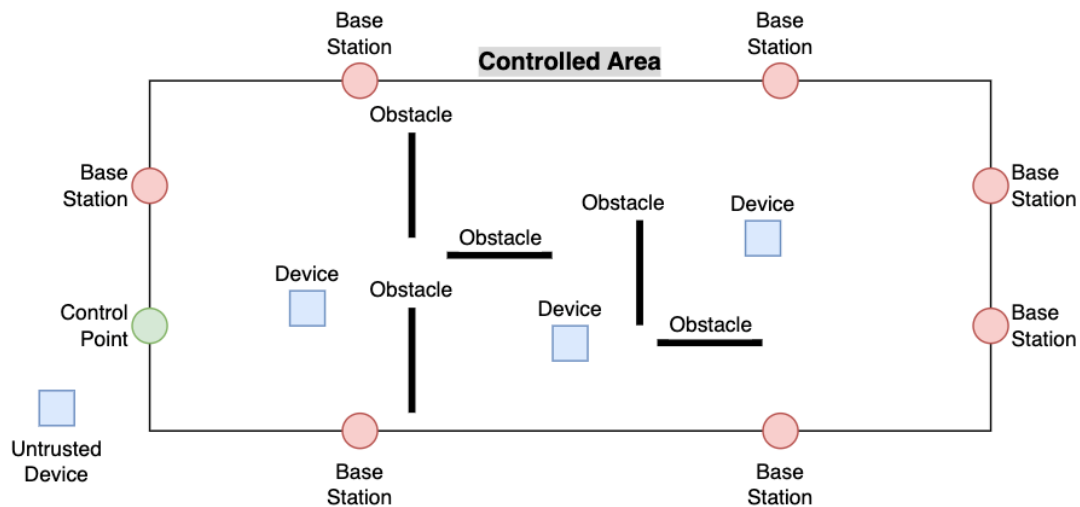


Figure 8.2.1. Abstract scenario for secure messaging scheme

The general security assumption for the messaging scheme is that agents can only be malicious as long as they have not passed the control point, e.g. they were not (yet) able to prove their identity. Once they accomplished this, they are considered trustworthy, so “hacking” of a device once it has passed the identification proof process is out of scope.

8.2.2 Iterative Trilateration

The idea of trilateration, i.e. to use distances to (in the 2D case) three position-calibrated anchors to calculate a device’s position, is covered in section 4.3.12. In this chapter, we leverage that idea to not only calculate the position based on fixed anchors, but to iterate on that, i.e. let a moving device be a positioning anchor itself for other devices once this device knows its own position. See Figure 8.2.2 for details: Device 1 can calculate its position based on the position+distance information from three fixed anchors, which could be radio base stations. Once device 1 knows its position, it can serve as an anchor for device 2 since device 2 can only see two fixed anchors, but can leverage the information from device 1 to perform trilateration.

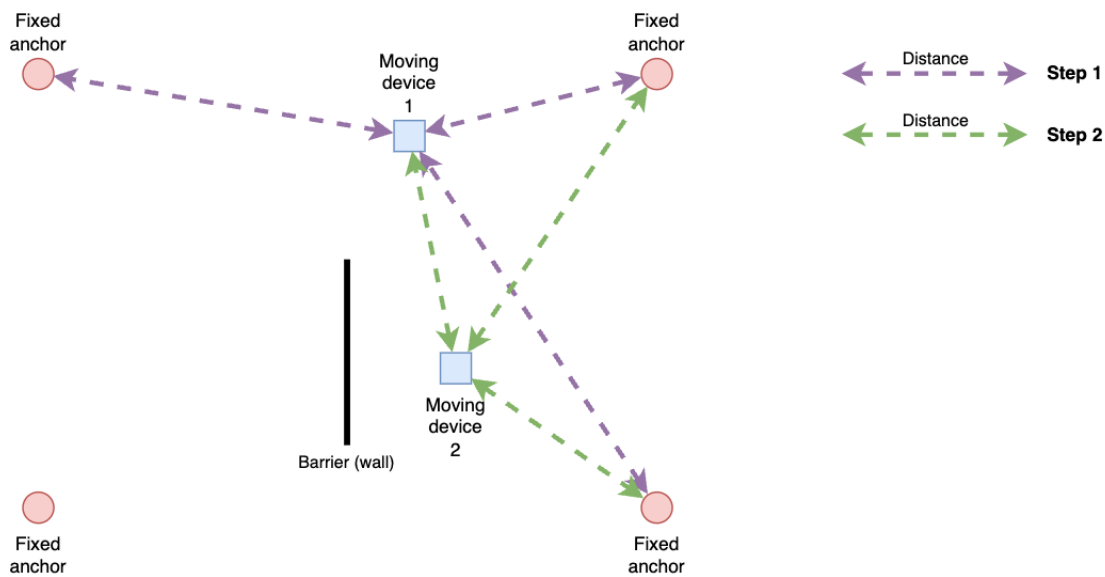


Figure 8.2.2. Iterative Trilateration

8.2.3 Digital Signature Scheme

The messaging scheme all agents communicate with is based on a Private/Public Key Pair (SK = secret key, PK = public key). The key pair is self-generated by the device.

The key pair satisfies the digital signature² scheme using the following cryptographic functions generate(), sign() and verify():

- generate(random number source) = (SK, PK)
- sign(SK, message) = signature
- verify(PK, signature, message) = True

generate() is executed only once by every agent, which then stores both keys locally. The SK should never leave the agent's security context.

sign() is executed by a message's sender for every message intended to be sent, and the resulting signature is sent alongside the message to the receiver. The receiver then executes the verify() function using the sender's public key, the signature, and the message received by the sender.

For the secure messaging scheme described here, we assume that localisation data does not require privacy measures, therefore, encryption is not leveraged.

8.2.4 Public Key List

For the messaging scheme to work, the public keys (PK) of participating agents need to be made available to each other. This is accomplished through the Public Key List where the public keys of all participating nodes are present. The list's "master" version is managed by the Control Point, which takes care of distributing it to all participating agents.

For position-fixed agents, like base stations and the Control Point, their public keys constitute version 1 of the Public Key List. For moving agents, their public key is presented to the Control

² Digital Signature: https://en.wikipedia.org/wiki/Digital_signature

Point as a special credential within the Hyperledger-based approach of the previous project stage (see respective deliverable [8-4]). The Control Point then updates the Public Key List by storing the presented public key and distributes (broadcasts) the update to the agent network through the secure messaging. The public key also serves as an ID for the device. This on-boarding procedure is visualized in Figure 8.2.3.

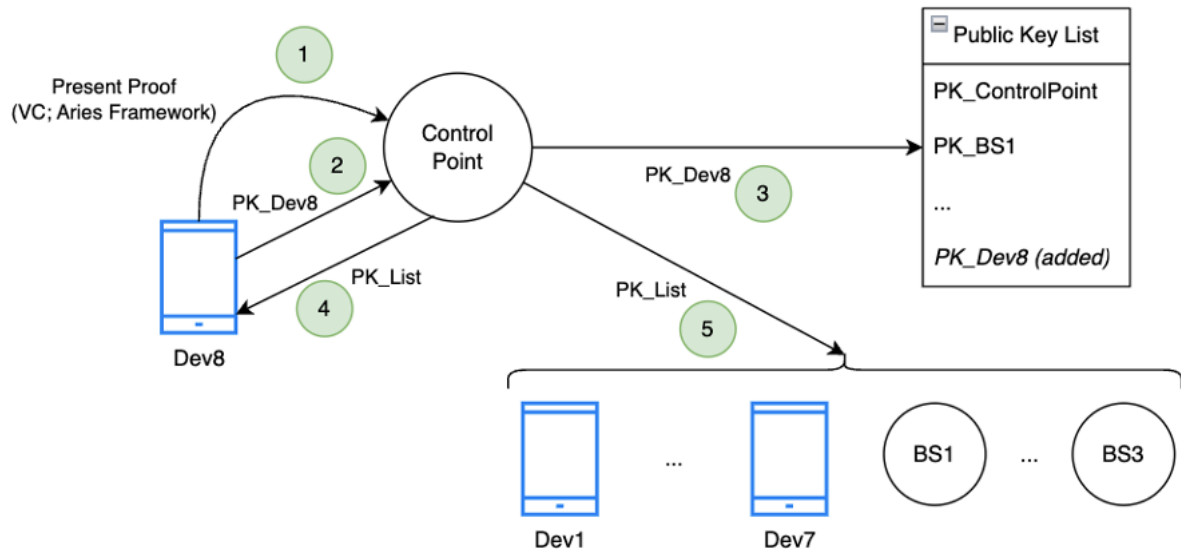


Figure 8.2.3. Public Key List distribution after device on-boarding

The Public Key List can be interpreted as the collection of agents that are in a controlled area. All agents that are in this list can be considered trustworthy since they have passed the Control Point proof verification using the Hyperledger-based Verifiable Credentials.

The Public Key List contains an associative array (or *hashmap*) with the following key/value pairs:

public key => time stamp when added

Static agents with calibrated positions (Base stations, Control Point) have the smallest possible time stamp (e.g. Unix epoch time = 0).

8.2.4.1 Public Key List Update History

Additionally, the Public Key List also contains an (associative) array of all updates it has received and applied. This is used by a device to keep track of changes and identify missing gaps, and subsequently react accordingly.

Each entry in the updates history is referenced by the version number, and has the following format:

- **Insertions:** (Public key => Time stamp) entries of devices that are added through this version
- **Deletions:** List of public keys of devices that are deleted through this version

8.2.5 Messages and Message Format

Messages are the foundation of information exchange in the scheme presented here. For the positioning efforts, two message realms can be distinguished: Localisation requests and information, and identity (public key) updates and update requests.

A message in this context comprises the following elements:

- **Message ID**
When an agent creates a message, it assigns it a message ID.
When an agent receives and forwards a message, the ID is not changed.
When an agent receives a message that it has received before (same ID), it discards it.
- **Type**
The following types of messages are a minimum set for the secure localisation scheme:
 - **Localisation request messages** (Broadcast, no receiver's public key)
 - **Localisation information messages** (Dedicated messages after a localisation request)
 - **Public Key List update requests** (Broadcast)
Whenever a device encounters gaps in its copy of the list (or an old version), it requests the missing list elements.
 - **Public Key List updates** (Either broadcasts, or dedicated receiver if it is a reply to an update request)
Insertions and deletions to the list.
This message is sent by the Control Point if a device enters or leaves the controlled area, forwarded by agents that have received it (as a broadcast), or sent by agents that have received an update request message by a neighboring agent.
- **Payload:** The message's content, depending on the type of message. See subsection 8.2.5.1.
- **Time stamp:** ... when the message has been generated. Taken from the sender's device clock, so these clocks must be synchronized.
- **Receiver's public key:** If a message is not broadcast, but directed towards some specific receiver, the receiver's public key is used as an identifier. Dedicated messages (localisation, Public Key List update) are replies to request messages (localisation request, list update request).
- **Sender's public key:** The public key must be present in the Public Key List.

The whole message described here is signed with the sender's public key. The signature is sent alongside the message, in order for receivers to verify the message. See section 8.2.3 for details.

8.2.5.1 Payload per message type

Localisation request

No further information.

Localisation information

- *Distance* between an anchor node (other agent; message sender) and the device that requested localisation beforehand (message receiver)
- *(x,y)-position* of anchor node (message sender)

Public Key List update request

Either ...

- *Current version* of Public Key List at the sender's side. This is needed for receivers to compare it against their latest version and, if different, send Public Key List updates to the requesting party, subsequently.

This variant of the update request message is sent when a device has encountered several message receptions by senders whose public keys were not part of the device's copy of the Public Key List.

... or ...

- *List of missing updates*. This is needed for receivers to look up whether they can provide some of the missing updates to the requester.

This variant of the update request message is sent when a device detects gaps in its update history of the Public Key List (after having received an update).

Public Key List update

- *List of updates*. This is sent either by the Control Point when a device entered or left the controlled area (= modification of the "master" Public Key List that resides at the Control Point), or when an agent received an update request by a neighboring agent.

8.2.6 Message Verification

The verification of a sent message by the receiver consists of the following steps. If any of these steps fails, a message is discarded.

1. *Does the message have a known message type?*
See "Type" in section 8.2.5 above.
2. *Has the message not been received before?*
The message's ID is checked against a history buffer maintained by the receiver. If the message's ID is found on this list, the message is discarded. For practical reasons, we introduce a buffer size of received message IDs. If the buffer reaches its limit, the first ID added to it will be removed from the list (Circular buffer).
3. *Does the message satisfy the digital signature scheme?*
See section 8.2.3 above.
4. *Is the sender's public key part of the Public Key List?*
Messages are only considered valid that are sent from trustworthy senders. Senders are trustworthy when their public keys can be found in the Public Key List.
5. *Has the message been sent recently?*
If the time stamp (see previous section) of a message is too far away in the past, the message is no longer valid. This measure prevents the execution of replay attacks, i.e. re-sending messages that have been eavesdropped before.
6. *If the field **Receiver's public key** is set within a message, does it match the actual receiver's public key?*
If a message is dedicated to a different receiver, it should be discarded.

8.2.7 Events

The following events trigger actions at agents (devices, base stations, Control Point), and message transmissions between agents. In this section, “agent” refers to both moving and fixed agents, while “device” refers only to moving agents.

Device enters controlled area (through Control Point)

- Device sends Public Key to Control Point (through “out-of-band” communication channel, e.g. Hyperledger Aries, see previous deliverable [8-4])
- Control Point saves device’s Public Key onto Public Key List and thus generates new version
- Control Point broadcasts Public Key List update
- Control Point initializes device with current public key list (incl. complete update history) through “out-of-band” communication channel
- Device generates message: Localisation request

Device leaves controlled area (through Control Point)

- Control Point deletes device’s public key from Public Key List and thus generates new version
- Control Point broadcasts Public Key List update

Agent receives Public Key List update

- Agent updates its copy of the Public Key List
- Agent checks if copy is complete / the latest.
=> If not, device generates message: Public Key update request
- Agent broadcasts (=forwards) Public Key list update

Agent receives Public Key List update request

- Agent checks own copy of Public Key List if it contains updates that were requested.
=> If so, agent generates message dedicated to requester: Public Key update

Device’s position information gets invalidated (time- or movement-triggered)

- Device generates message: Localisation request

Agent receives Localisation request

- Agent checks if it knows its own position.
=> If so, agent generates message dedicated to requester: Localisation information

Device receives Localisation information

- Device saves localisation information
- Device checks whether enough information is present for trilateration (three localisation information receptions), and whether positions of anchors are not on or near a common line (ambiguous position calculation)
=> If so
 - Trilaterate own position
 - Set “knows own position” as true in order to serve as localisation anchor for other devices from there on

8.3 Experimental Simulation Setup

8.3.1 General assumptions

For the simulation experiments the following assumptions and simplifications were put in place:

- **2D scenario only**
We consider localisation using three anchors, i.e. only 2D scenarios are covered.
- **Line of Sight (LoS) and maximum communication range**
Our communication scheme abstracts from a specific radio communication technology, but we assume only *line of sight* communication. This means that whenever an obstacle blocks line of sight between two devices (or base stations), message transmission does not occur.
Furthermore, we introduce a *communication range*. Whenever two devices are more than this communication range away from each other, transmission of a message does not occur. For simplicity reasons, all transmitters, be they fixed or moving, have the same communication range.
- **Devices have no spatial extent.**
Devices don't block line of sight (and therefore, message transmission) between other agents.
- **Measurement-agnostic distance calculation**
The calculation of distances is not part of the messaging scheme. Localisation messages contain distances that have been determined with whatever method beforehand.
- **100% Accuracy**
Transmitted distances are 100% accurate within the simulation.
- **Expiration of position knowledge**
Within the simulation, a device forgets its position after a configured time span. This can be justified by the movement of a device and the increasing uncertainty. As a consequence, the device a) needs to request a new position, and b) it cannot serve as a positioning anchor itself until it has calculated its new position.
- **Simplified message verification**
The simulation does not cover the behaviour of malicious devices, since it focuses on the feasibility of the introduced messages and the improvement that iterative trilateration could bring to the scenario. Therefore, the above mentioned checks regarding digital signature scheme and time stamp validity do not need to be performed in the simulation.
- **Simplified movement: constant velocity, and only along x and y axis**
In the simulation, devices move only 1 unit (pixel) per time step, and only in one of the four cardinal directions (N, S, E, W). Movement of devices is only covered to distribute devices in the controlled area, starting at the Control Point, to render a semi-realistic campus-like motion behaviour. The movement frequency is configurable.
- **Distance-independent wired and wireless latency**
The communication latency, i.e. the time between transmission and reception of a message, is configurable, but independent from the distance between sender and receiver. Two different latencies are configurable, one for wired connections

(=between fixed anchors, including the Control Point) and one for wireless connections (=where a moving device is involved).

- **Constant scenario size**
Spatial scenarios all have the same size, so that both simulation parameters and results are comparable.
- **Obstacles**
Obstacles in the scenarios are all x-/y-axis aligned rectangular objects. They both block line of sight and device movement. When a moving device reaches an obstacle, it changes its movement direction by 90° in one of the two possible relative directions randomly.

8.3.2 Simulation implementation

The simulation has been implemented using Python based on the `asyncio`³ library. This library is used to execute concurrent code asynchronously. This property is leveraged to implement the above-mentioned messaging communication between agents, taking into account simulated latencies, as a discrete event simulation.

Base stations, the control point, and moving devices are implemented as a dedicated **Agent** class. Agents are able to verify messages (8.2.6), decode the defined message types (8.2.5) and react according to the defined events (8.2.7). The control point, implemented as the **ControlPoint** subclass of Agent, additionally manages the control flow and public key list update actions when devices enter or leave the controlled area.

The message passing between agents is managed by a central **MessageHandler** class that implements all the control logic needed to decide which agent receives which message, taking into account line-of-sight detection, and message deferral due to latency considerations.

The digital signature scheme is implemented using the Python cryptography⁴ library, specifically the Ed25519 classes/functions.

The spatial scenario that determines where base stations, the control point, and obstacles are situated and where devices can move, is implemented as a separate **Scenario** class. This class handles reading scenarios from a JSON file, giving information for line-of-sight and movement decisions, and is able to visualize either the static scenario (borders, obstacles, fixed anchors), a dynamic scenario (static + current positions of moving devices + LoS lines) and a trilateration coverage visualization.

A simulation follows these process steps:

- Initialization with scenario and parameter configuration
- Save configuration & scenario as metadata to a JSON file.
- Initialization of logging and measurement-capturing files.
- Creation of Agent objects for all defined base stations and the Control Point.
- Definition of recurring events as asyncio tasks:
 - Introduction of new devices to the controlled area
New devices are created at the control point periodically. The frequency of how often this occurs is configurable and part of the experiment parameter

³ asyncio - Asynchronous I/O: <https://docs.python.org/3/library/asyncio.html>

⁴ pyca/cryptography: <https://cryptography.io>

space. This behaviour is intended to represent a typical IoT factory scenario where new agents appear and helps to assess whether the distribution of changes in the node network works.

- Position invalidation (implicit): Once a device has been created, it requests to receive localisation information for itself. The device implicitly schedules events to refresh its localisation information, making the current one invalid. The frequency in which a position gets invalidated is also configurable.
- Movement of devices
Devices are moved around the area at some specified frequency. This is also configurable. Movement is part of the simulation since it helps assessing whether devices are able to establish a trilateration setup each coming from a starting position, which also represents the factory scenario where 3rd party devices enter a controlled area.
- Statistics collection
For the analysis of the simulation's outcomes several KPIs and measurements need to be collected and written to a file. Evaluation is done as a post-processing of multiple simulation runs in order to compare them.
- The simulation's duration is implemented as an `asyncio.sleep()` call. The duration is configurable.

8.3.2.1 Configuration Parameters

The simulation can be configured with a variety of parameters, from which only the most important ones are mentioned here.

Constant parameters for the simulation runs used for the experiments covered here:

- **Simulation duration** => 180 seconds.
- **Spatial scenario size** => 120x72 pixels
Note that all spatial parameters are given as pixels, but could be interpreted as any spatial unit.
- **Localisation refresh rate** => 1 second
Rate at which devices "forget" their positions and request new localization information.
- **Movement rate** => 0.5 seconds
Rate at which devices change their position, i.e. make one step.
Statistics collection rate => 0.05 seconds
Rate at which all relevant statistics are collected and written to a file.
- **Start at Control Point (Yes/No)** => No
If "Yes", all devices that are emitted to the controlled area start at the position of the Control Point, making it more realistic wrt. an actual physical scenario.
If "No", all devices' start positions are randomly seeded over the whole scenario area (except from obstacles), having a better distribution of devices for iterative trilateration.

Dynamic parameters, that constitute simulation runs when varying them:

- **Communication range**: see 8.3.1, "Line of Sight"
=> *Default value: 53 pixels*. This is the value for an 100% unobstructed scenario of size 120x72px with the basestations and the control point positioned as can be seen in

the following section.

=> *Variation: 10 .. 80 pixels*

- **Fixed trilateration coverage (FTC):** see following section 8.3.2.2
=> *Default value: ≈0.5* (random scenario with FTC = 0.5 .. 0.59)
=> *Variation: 0.08 .. 1*
- **Wireless latency:** see 8.3.1, “Distance-independent wired and wireless latency”
=> *Default value: 0.001 seconds*
=> *Variation: 0.001 .. 0.1 seconds* (logarithmic variation)
- **Device creation interval:** Interval in which devices are created and introduced to the controlled area. This affects the average number of devices roaming around in the area.
=> *Default value: 5 seconds*
=> *Variation: Simulation duration / (10 .. 110 devices)*
- **Devices as trilateration anchors (Yes/No)**
This parameter is varied in every simulation experiment: Since this parameter represents the feature that enables iterative trilateration and is therefore at the core of the evaluation, each simulation run is actually two runs: One where this parameter is enabled, and one where it is not. These two are then compared wrt. the effectiveness of the measure.

8.3.2.2 Spatial Scenario Generation

Studying the effect of variations in simulation parameters is quite straight-forward for most parameters, e.g. communication range, number of devices, frequency of position invalidation etc. Comparing different spatial scenarios is more complicated: It requires some metric that accounts for the “fixed trilateration effectiveness” of a scenario, and how the messaging scheme used for iterative trilateration can conquer this. Secondly, a method of creating scenarios with different levels of this metric is essential to run different simulations that vary this parameter.

In order to assess the “obstructedness” of a spatial scenario, we introduce a **fixed trilateration coverage (FTC)** metric that calculates how many points within an area can be determined using trilateration from any three of the fixed localisation anchors, like base stations and the Control Point, taking into account all obstacles. Note that the communication range is not factored into this metric, since it should be treated as a separate simulation parameter. To keep calculation effort reasonable, we take only a sample size of points, with their (x,y) coordinates evenly distributed. The **fixed trilateration coverage** of a scenario is thus calculated as

$$FTC = \frac{\text{number of points where trilateration is possible}}{\text{sample size}}$$

The possible values range from 0 (totally obstructed) to 1 (all points can be reached using fixed trilateration).

For the simulation experiments, multiple scenarios with varying FTCs are needed. There are several approaches to accomplish this. We have decided on a grid layout of vertical and horizontal line obstacles with gaps in it to allow device movement. The maximum grid is depicted in Figure 8.3.1. The blue dots mark base stations, the magenta dot is the control point.

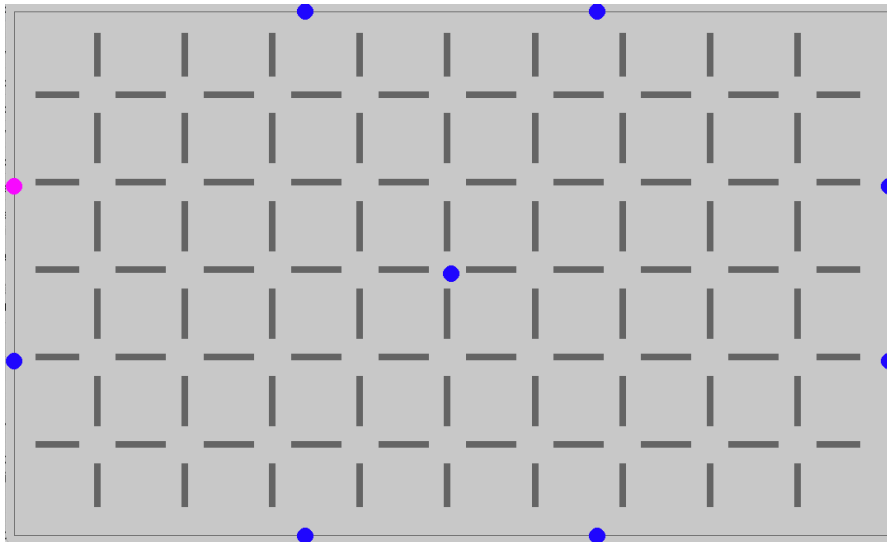


Figure 8.3.1. Grid scenario to derive different coverage levels

The “full” grid shown here has an FTC of around 7%. To come up with different FTC values, we randomly choose subsets of the obstacles to create new scenarios, and calculate the respective FTC values. Finally, we select a set of scenarios with FTC values ranging from 8% to 100% that make up the parameter space as to scenario selection.

8.3.3 Statistics collection

For each time step of the statistics collection task, the following metrics are collected:

- Time stamp
- Number of moving devices
- Number of moving devices that are aware of their own position
- Number of moving devices that use the latest Public Key list version
- For each message type (see 8.2.5)
 - Number of messages sent so far

8.4 Experimental Simulation Results and Analysis

8.4.1 List of experiments

The following parameter variations were used to perform sets of simulations. For all parameters not varied in a specific set, the default values mentioned in 8.3.2.1 were used.

- **Communication range:** 10 .. 80 pixels, step size 2, 10 random scenarios with $FTC \approx 0.5$
- **FTC:** 0.05 .. 1 (interval size 0.05), 17 scenarios per FTC interval
- **Latency:** 0.001 .. 0.1, logarithmic step size $10^{-0.2}$, 10 random scenarios with $FTC \approx 0.5$
- **Device creation interval:** Simulation time / (12 .. 112 devices), step size 4 devices, 10 random scenarios with $FTC \approx 0.5$
- **Communication range & FTC**
 - Communication range: 35 .. 83 pixels, step size 8
 - FTC: 0.05 .. 1 (interval size 0.05), 5 scenarios per FTC interval
- **Device creation interval & FTC**
 - Device creation interval: Simulation time / (12 .. 112 devices), step size 20 devices

- FTC: 0.05 .. 1 (interval size 0.05), 5 scenarios per FTC interval
- **Device creation interval & Communication range**
 - Device creation interval: Simulation time / (12 .. 112 devices), step size 20 devices
 - Communication range: 35 .. 83 pixels, step size 8

8.4.2 Effectiveness of secure messaging scheme

The first question to investigate is about the capability of the secure messaging scheme to ensure the communication between all of the participating agents. A good indicator for the effectiveness of the secure messaging is the fraction of moving devices that are working on the current (latest) version of the Public Key list, i.e. the one that is maintained by the control point. If a device is not up-to-date wrt. the Public Key list, that could mean that it is moving in an area where it does not receive identity updates at all, or only by devices that are new to it, i.e. which it could not find as trustworthy entities on its own copy of the Public Key list. A high percentage of agents working on the latest Public Key list version indicates that the messaging scheme works.

Figure 8.4.1 shows the fraction of devices working with the current version of the Public Key list for each time step over the course of a single simulation run. Nearly all of the time all devices work on the latest version, and when this is not the case, this is always corrected quickly.

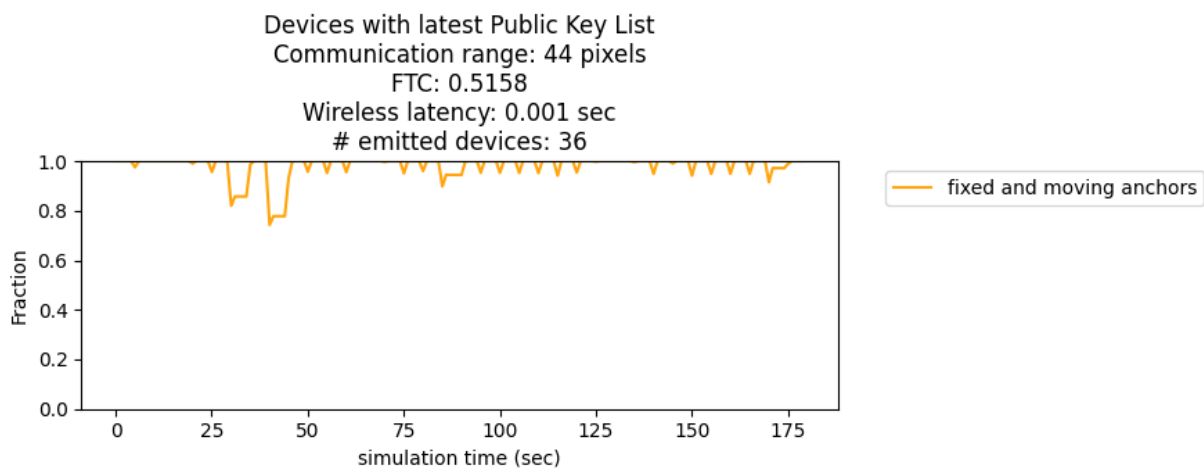


Figure 8.4.1. Fraction of agents with latest Public Key list version for one simulation run

This measurement can be summed up over the simulation run, and the result can be compared to other simulation runs with varying parameters. In nearly all scenarios, this metric is near 1, with two exceptions: When the communication range is low, or when the FTC is very low, the number of “current” agents decreases, as can be seen in Figure 8.4.2 and Figure 8.4.3. The number of devices in a scenario, at least in the range covered here, and the wireless latency don’t affect this metric.

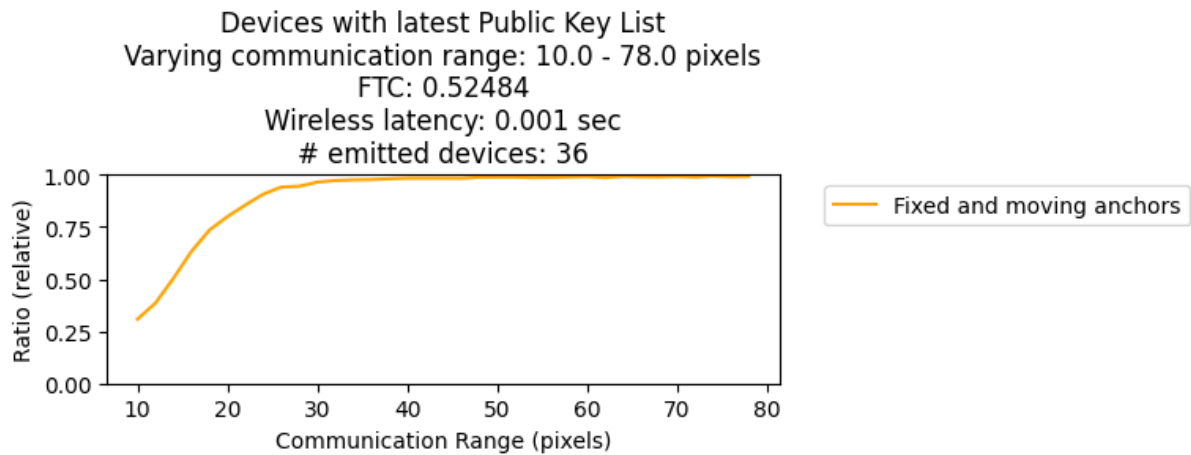


Figure 8.4.2. Fraction of agents with latest Public Key list for varying communication ranges

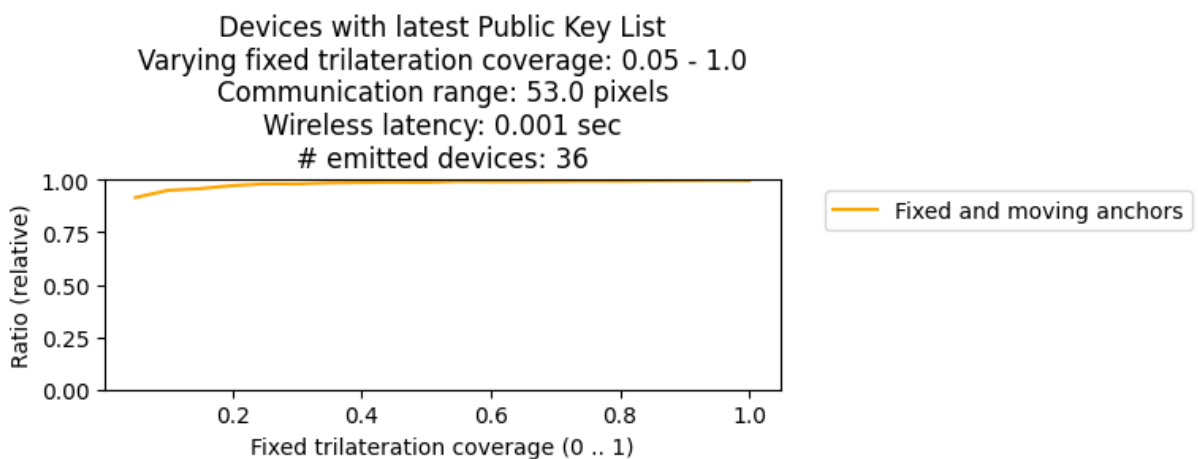


Figure 8.4.3. Fraction of agents with latest Public Key list for varying fixed trilateration coverage levels.

In summary, the secure messaging scheme works in almost all of the covered parameter combinations, except from edge cases.

8.4.3 Localisation success

The second question to tackle is the capability of the iterative trilateration to help agents determine their positions. For this evaluation, we calculated the fraction of agents that know their position, after having acquired localization information from three anchors, for every time step of a simulation run. As stated, we compare this value for these two cases: 1. Only fixed anchors (i.e. base stations) can provide positioning information, and 2. additionally, moving devices aware of their own position can provide positioning information to (other) devices. To measure the success of a simulation run based on a parameter set, we again sum up the single-time-step values.

As can be seen in Figure 8.4.4, the number of devices capable of determining their position using trilateration fluctuates around some value when only fixed anchors can be consulted (orange line). On the other hand, the number of successful localizations increases in the case of moving and fixed anchors (blue line) when more devices enter the area. Fluctuations in

both cases are due to individual devices moving into areas where no three anchors are reachable.

When this measurement is summed up over a whole simulation run, we can again compare it against other parameter sets. Figure 8.4.5 to Figure 8.4.8 show the results for different variations of one parameter each, keeping the others to default values according to section 8.3.2.1. The findings for the different variations are:

- *Communication range*: For short ranges, trilateration is not possible at all. For medium ranges, the success of trilateration increases, with a significant improvement when moving anchors can be leveraged, since they not only help getting around obstacles, but also “bridge” short communication ranges. For long communication ranges, the improvement compared to the “fixed anchors only” scenario diminishes a bit, but is still significant.
- *Fixed trilateration coverage (FTC)*: The trilateration success increases continuously in the fixed anchors case. In the moving anchors case, the improvement is small for low FTC values (i.e. high obstructedness) and significant over most of the middle FTC values. For highly unobstructed scenarios, devices do not help anymore since most of the fixed base stations can be reached for positioning efforts.
- *Wireless latency*: In the range covered in our simulations, latency does not matter very much. Only high latency values affect the positioning success negatively since devices need to refresh their position at some rate, making them unaware of it most of the time (see parameter “Localisation refresh rate” in section 8.3.2.1).
- *Number of devices* (... emitted to the area): Adding more devices to the controlled area does of course not help in the scenario where only fixed anchors are used. In the case of additional moving anchors, they help up to a saturation point where more devices don’t provide any more improvement. This finding is relevant when it comes to the examination of the “messaging footprint” of the scheme, since more devices means even more device-to-device messages being sent around (see below).

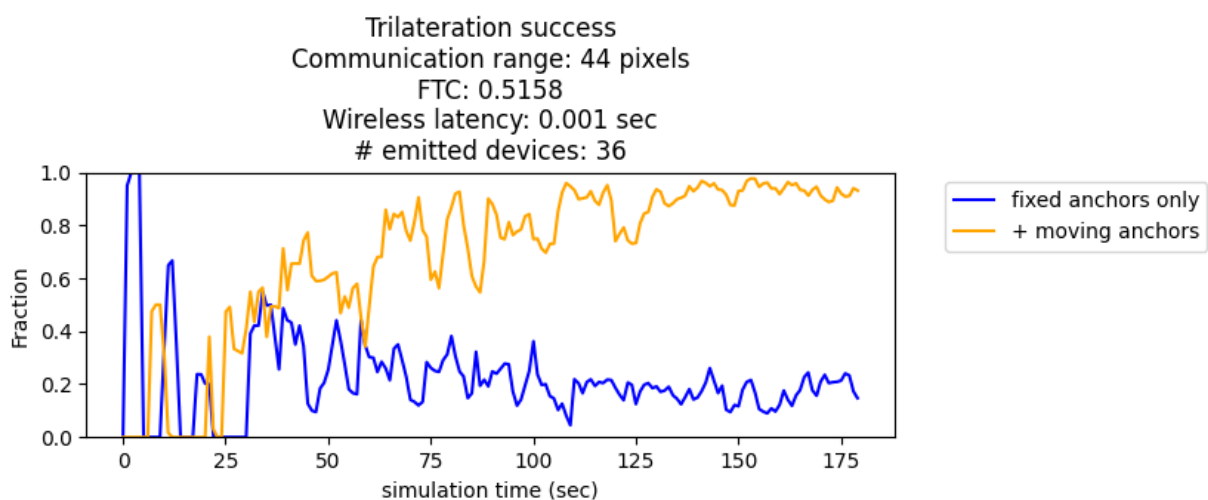


Figure 8.4.4. Trilateration success for one simulation run

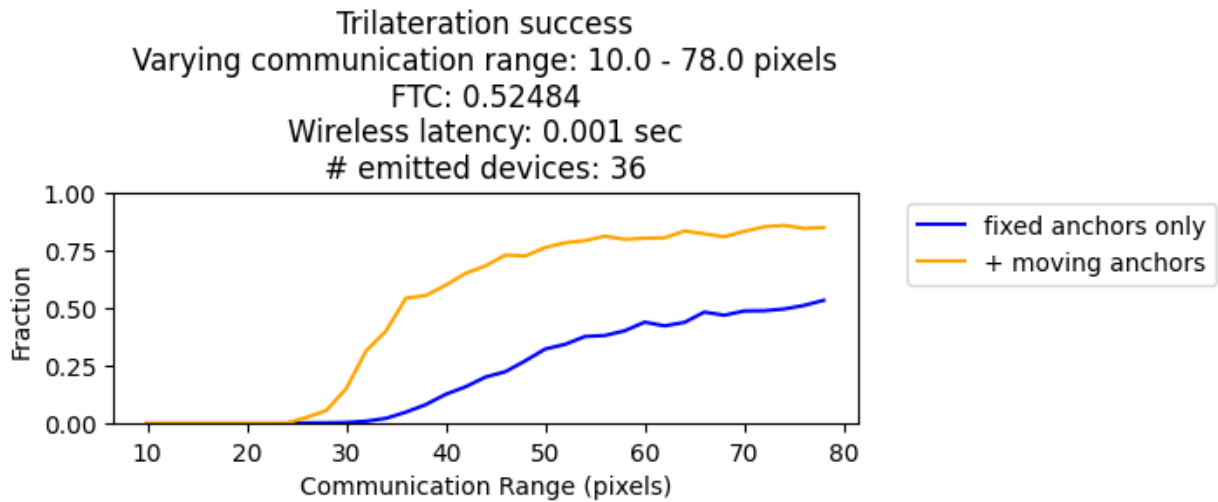


Figure 8.4.5. Trilateration success for varying communication ranges

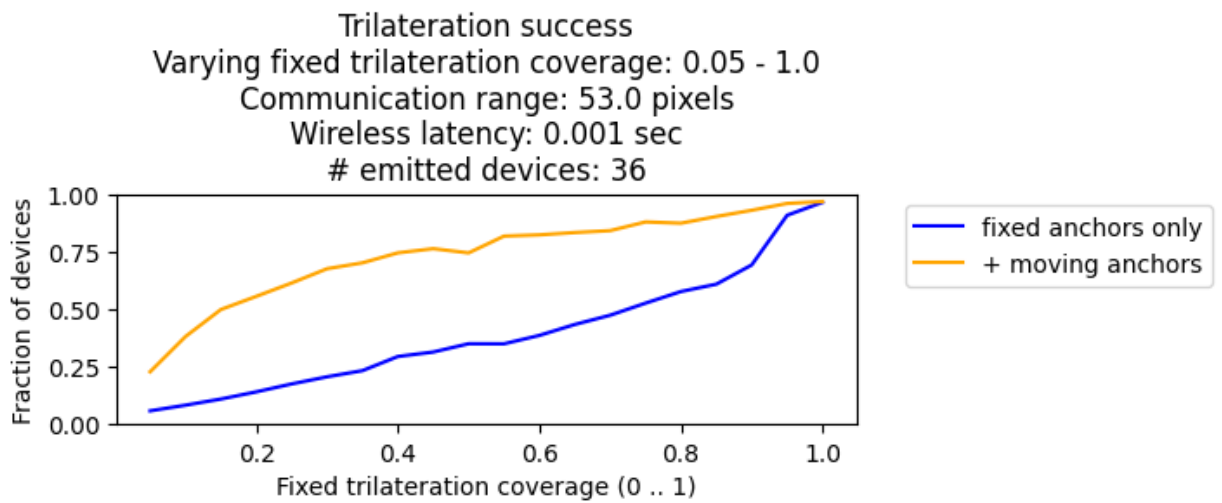


Figure 8.4.6. Trilateration success for varying FTC levels

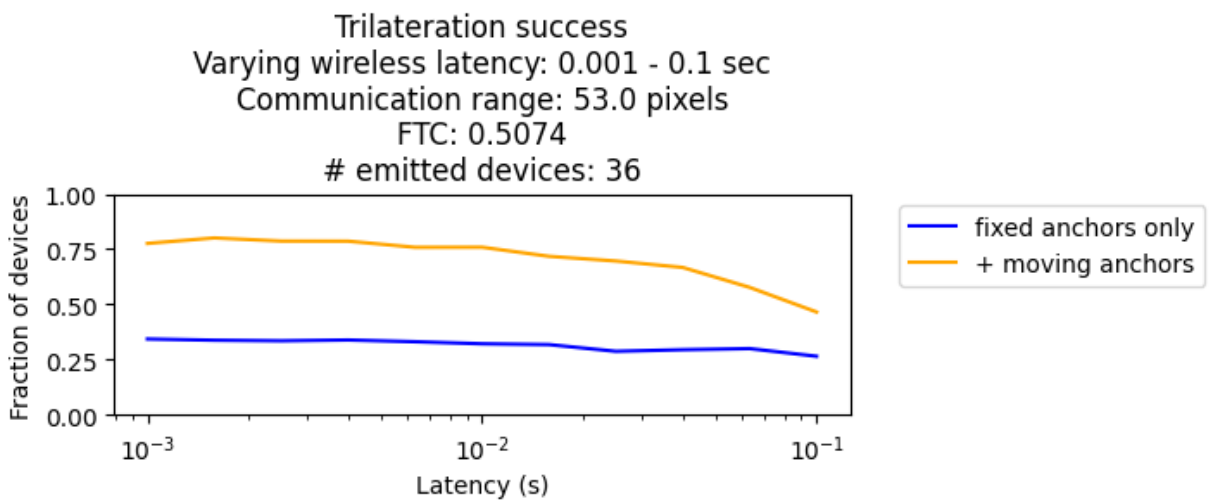


Figure 8.4.7. Trilateration success for varying wireless latencies

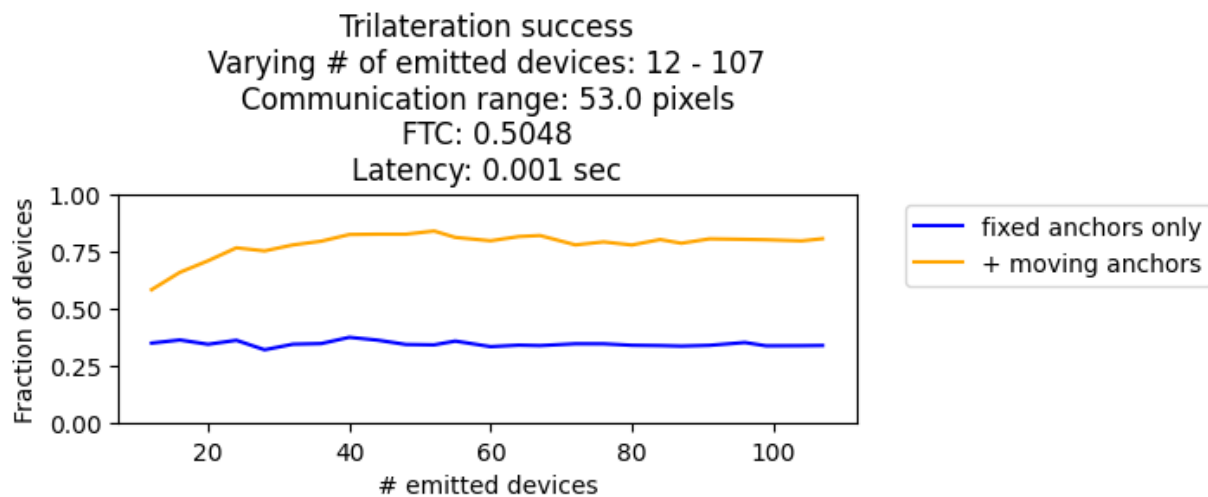


Figure 8.4.8. Trilateration success for varying number of devices within the scenario

We also evaluated parameter combinations where two of them are varied simultaneously. For this, we defined separate runs of simulations (see section 8.4.1). The metric to assess is the absolute improvement of the positioning success when moving anchors are introduced, additionally to fixed anchors, since we considered an improvement from 0% to 20% equally effective compared to one from 80% to 100%. The findings from these simulation runs are:

- *FTC and Communication Range* (Figure 8.4.9): For short communication ranges (blue and orange line) in very obstructed scenarios (small FTC values), there is no significant improvement in the positioning success since it is quite hard for moving devices to get to every point that needs an adequate anchor coverage. In contrast, unobstructed scenarios (high FTC values) benefit from moving anchors when the communication range is short due to the effect of “bridging” the communication range by devices. Once the communication range is high enough, highly “accessible” scenarios (high FTC) can be served by fixed anchors only.
- *FTC and number of emitted devices* (Figure 8.4.10): For highly unobstructed scenarios, again, there is no need for moving devices supporting the positioning effort. It also comes clear that the number of devices needs to be above some level to increase the positioning success significantly.
- *Communication Range and number of emitted devices* (Figure 8.4.11): One general finding is that the longer the communication range, the smaller the improvement of trilateration. Additionally, the number of devices needed to improve positioning is saturated at some point where more devices don’t improve the result anymore. For very small communication ranges, the number of devices shouldn’t be too small to see an effect at all.

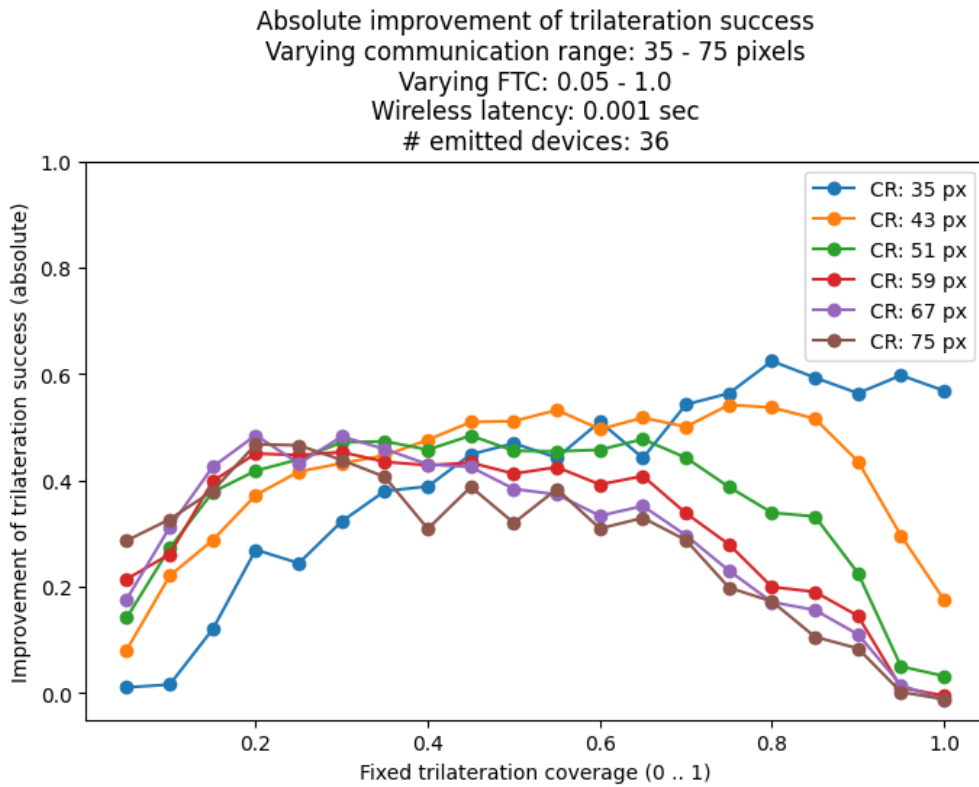


Figure 8.4.9. Improvement of trilateration success for varying FTCs and communication ranges

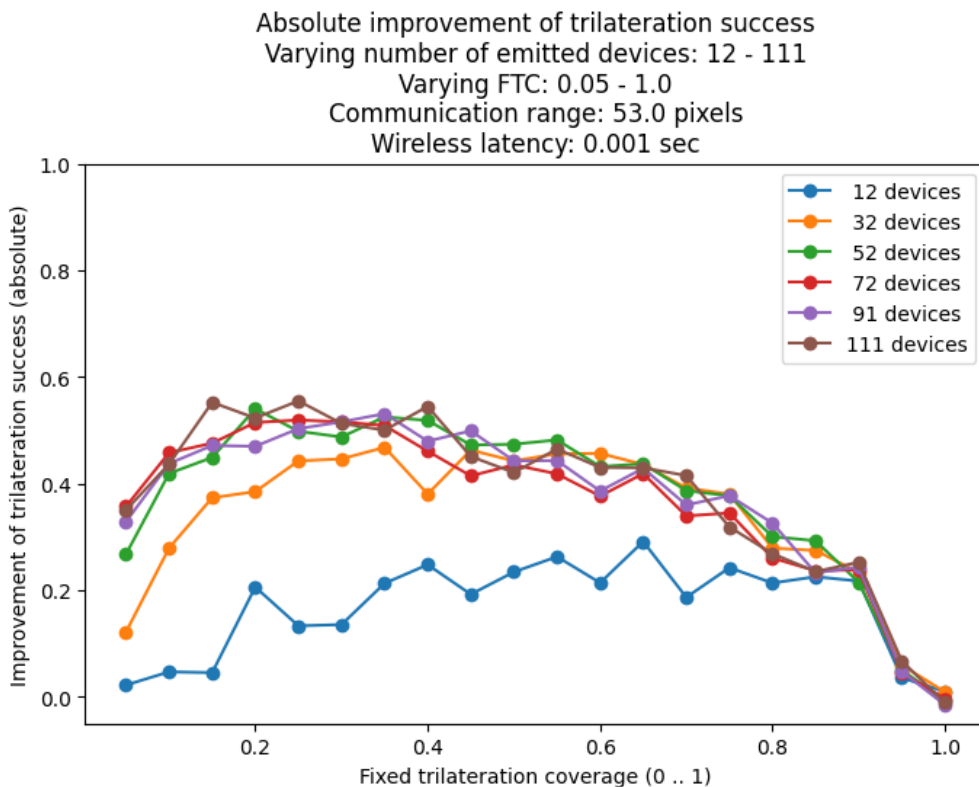


Figure 8.4.10. Improvement of trilateration success for varying FTCs and number of emitted devices

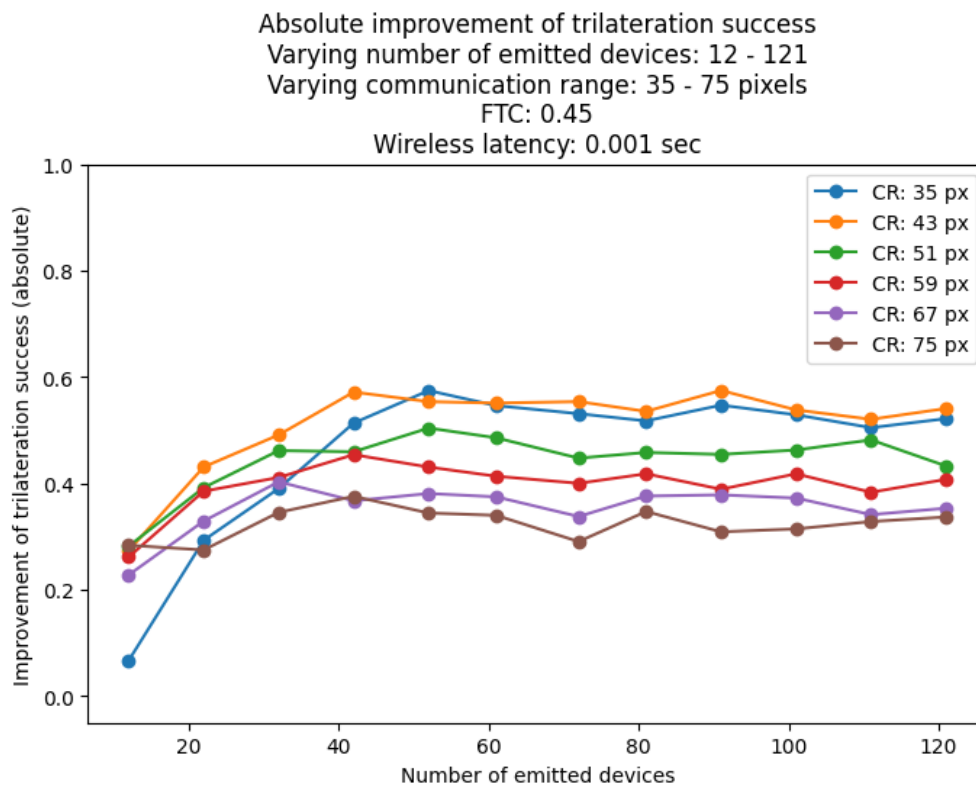


Figure 8.4.11. Improvement of trilateration success for varying number of devices and communication ranges

The success rate of the sheer identity updates (previous section) is way higher than the positioning success since for the former only one reachable device is needed, in contrast to three anchors needed for trilateration.

The findings in this section can only serve as a baseline consideration for trilateration. To assess a concrete scenario, the simulator can be used when planning a setup and dimensioning it properly.

8.4.4 Messaging footprint

Finally, we want to examine the “cost” the messaging scheme causes when moving anchors are introduced for the positioning effort, and its composition.

Similar to the previous chapter, we collected the number of sent messages for different parameter variations, and compared the two cases with only fixed and additional moving anchor agents. As visualized in Figure 8.4.12, the number of messages increases when the obstructedness of scenarios goes down (increasing FTC). What’s more important is that when moving anchors are enabled, the number of messages is increased by a factor of four, due to the following causes:

1. When only fixed anchors can be used for trilateration, localization requests can only be answered by fixed anchors.
2. Additionally, there is no need for the Public Key list messages to be sent around, or to even manage a Public Key list at all, since no trust between devices is needed.

A similar curve can be observed for the number of emitted devices (diagram not shown here): The more agents are emitted to the area, the higher the number of messages between them. When it comes to latency, as can be seen in Figure 8.4.13, the number of sent messages decreases for high latencies since most messages are replies to requests, which results in fewer messages during the constant simulation time over all experiments.

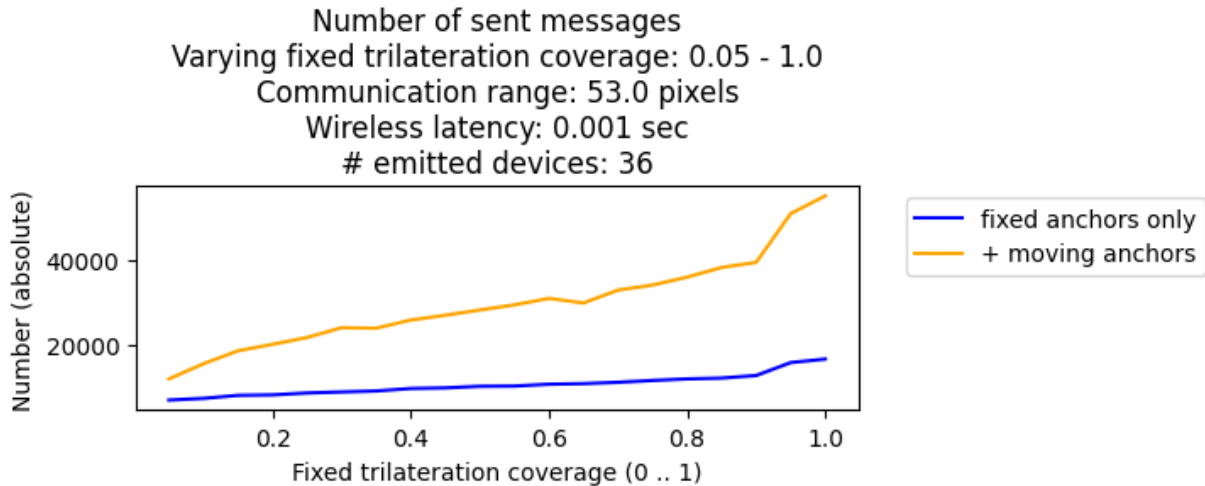


Figure 8.4.12. Number of sent messages for varying FTC values

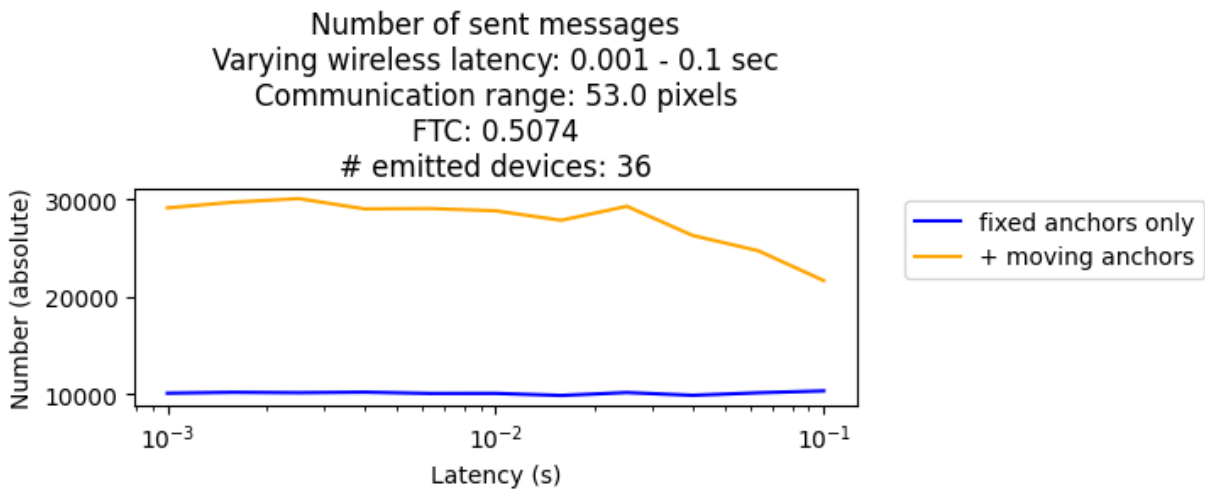


Figure 8.4.13. Number of sent messages for varying wireless latencies

One final inspection has been made regarding the composition of message types. Figure 8.4.14 shows the proportion of the four message types for varying communication ranges. For short communication ranges, the overall number of messages is comparably small due to the previous finding that most of the messages are answers to request messages. This makes the relative number of localization request messages (green) quite high since most devices don't get enough answers and retry to get localization information. For public key list messages, the initial messaging starts at the control point that distributes the Public Key list changes, which are relayed by all agents (base stations, moving devices) that have received it. Since the absolute number of messages is small for short communication ranges, even update requests (red) are relatively significant. For longer communication ranges, the LOCALIZATION_INFO messages prevail over LOCALIZATION_REQUEST's since the former happen as answers to the

latter by all nearby devices, most of which are ignored by the requesting device because after three answers trilateration can be executed. The number of Public Key list messages is only a fraction of that of localization. This is partly due to the *Localisation Refresh Rate* of 1s (see section 8.3.2.1), but should be a typical proportion in general since new agents should not appear (or leave) that often compared to the frequent need of devices to calculate their positions.

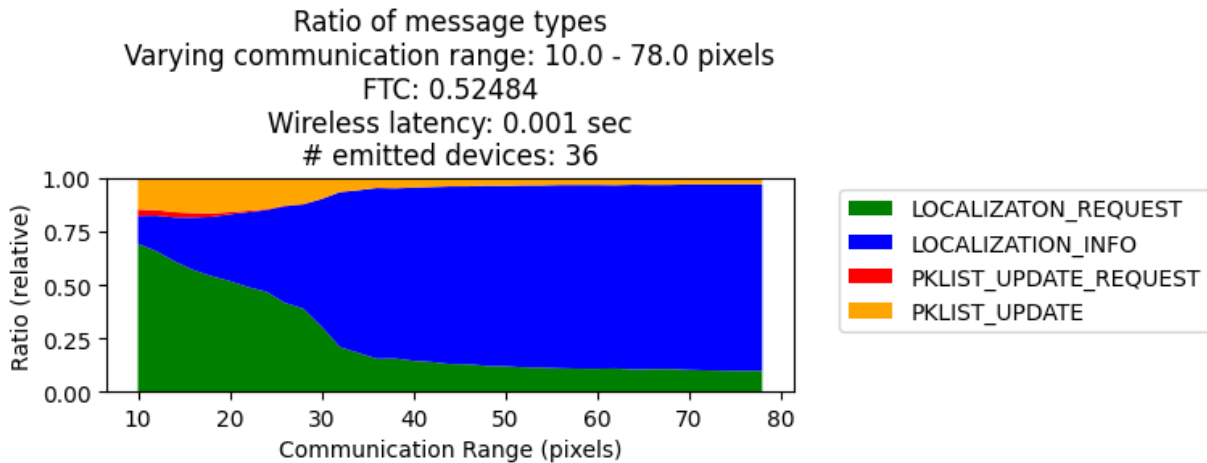


Figure 8.4.14. Proportion of message types for varying communication ranges

9 Optimised Architecture for Block Chain Integration in Industrial IoT

9.1 Architecture

The proposed system architecture leverages a combination of Raspberry Pi modules, decentralized IPFS storage, and blockchain technology to enable efficient storage, retrieval, and processing of data in an Industrial IoT (IIoT) environment. The framework consists of three main modules, described in the following subsections.

9.1.1 Raspberry Pi Module for Data Collection and Processing

To maintain the security of the system, the processed data is encrypted using the Tiny-AES encryption algorithm, bolstered by a shared key. This cryptographic shield secures the data against unauthorized access and supports transferring the data blocks safely. After sending the blocks of data to IPFS for storage, we retrieve the IPFS hash. At this point, the system initiates a blockchain transaction with relevant metadata, including IPFS hash, for smart contract execution and storage in the Harmony blockchain.

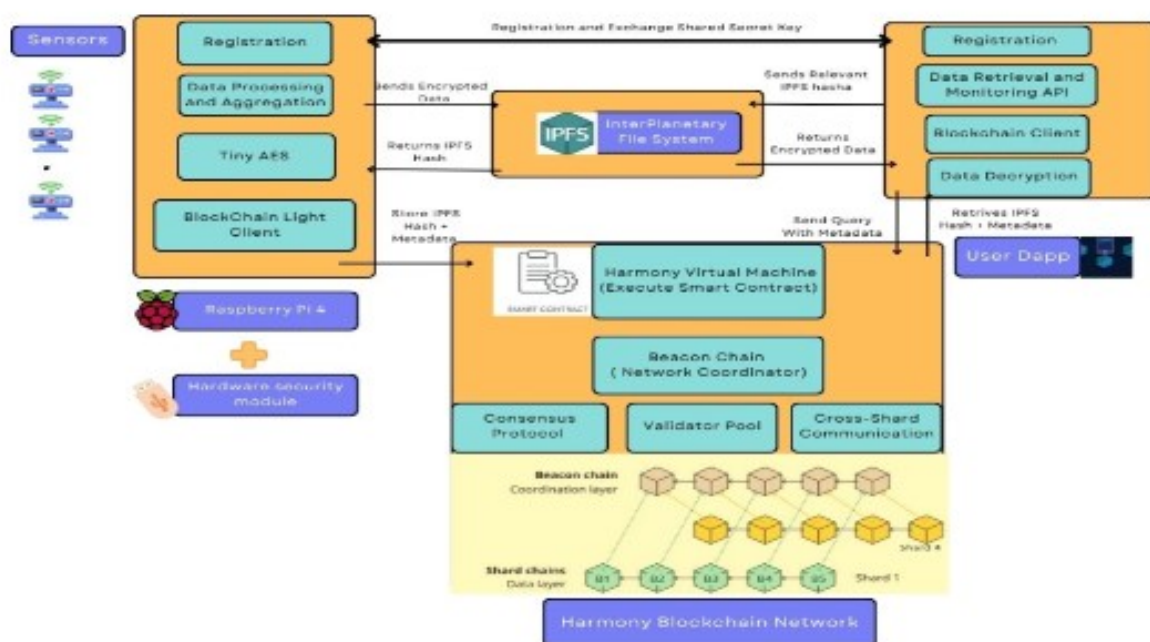


Figure 9.1.1: Proposed System Architecture

9.1.2 Decentralized IPFS Storage and Blockchain Smart Contract Module

The IPFS, ensures data ensures data integrity and global accessibility, while data encryption is done via with Tiny-AES to ensure privacy. The IPFS hashes are stored in the Harmony blockchain along with metadata using a sharding mechanism for concurrent transaction processing. The system utilizes the Beacon chain consensus mechanism to coordinate the network, validate transactions across shards, and update the overall network state. The Harmony Virtual Machine executes smart contracts in respective shards, updating contract states and incorporating modified data into the shard blockchain. After the transaction, the system issues transaction receipts, which contain essential details, including shardID, transaction fee, gas used, and cryptographic signatures.

9.1.3 User DApp Module for Data Retrieval and Monitoring

The User DApp is a decentralized application with a frontend and backend for industrial users, it enables users to register with unique Network ID, enabling authentication and connection to the deployed IIoT setups. During deployment, shared secret keys are exchanged securely for data encryption. Users are capable to query the blockchain with metadata (e.g., Raspberry Pi Node ID, location, date range) to retrieve data. The smart contract returns IPFS hashes, which are used to fetch encrypted data from IPFS. The application utilises shared secrets for decrypting data, making data readable for analysis and monitoring. User DApp can run either a Harmony blockchain light client or a fully functioning Harmony node, depending on resource constraints.

In summary, the proposed framework utilizes decentralized storage to ensure secure, efficient, and privacy-preserving data handling in Industrial IoT environments. The system promotes data integrity, immutability, and accessibility while providing a user-friendly interface for data retrieval and monitoring.

9.2 Aim of Experiment

9.2.1 Overall Aim

Explore the integration of blockchain in Industrial IoT (IIoT) for efficient data handling with a focus on privacy and trust.

9.2.2 Research Question 1 (RQ1): Privacy and Trust in IIoT

Identify and study security vulnerabilities in IIoT deployment.

Examine the role of Encryption and Distributed Ledger technology in ensuring privacy and trust.

9.2.3 Research Question 2 (RQ2): Integration of Blockchain with IIoT:

Survey and review different approaches to integrate blockchain in IIoT.

Identify challenges, benefits, and complexities in adopting blockchain in IIoT.

Design a blockchain-based architecture for storing, processing, and retrieving data in IIoT.

Develop and deploy the system using available resources on User equipment (mUEs).

9.2.4 Research Question 3 (RQ3): Evaluation of Blockchain Integration in IIoT

Identify evaluation metrics for Security, Efficiency, and Sustainability.

Evaluate and compare the performance and sustainability of the proposed blockchain-based architecture with other solutions.

9.3 Background Theory

9.3.1 Technological Evolution

Breakthroughs in electronics, wireless communication, and miniaturized technologies have led to a shift from the physical to the digital world.

Increased number and reduced costs of electronic devices contribute to the growth of the Internet of Things (IoT).

IoT, comprising wireless sensor networks and RFID, facilitates real-time data generation, enabling automation, actionable insights, and the development of smart cities, grids, and homes.

9.3.2 Industrial IoT (IIoT) and Industry 4.0

IIoT aims to create smart factories and enhance manufacturing through process automation and data exchange.

Industry 4.0 integrates IIoT with Cyber-Physical Systems to digitize and optimize supplier markets, manufacturing, and sales.

Real-time monitoring and data sharing enhance productivity and product quality, making businesses smarter and more efficient.

9.3.3 Challenges in IIoT and Role of Blockchain

Trustworthy handling of IoT data is crucial, requiring a distributed service that is reliable and ensures data integrity.

Blockchain, with its decentralized structure and storage mechanism, addresses IoT challenges, particularly in data protection and privacy.

9.3.4 Scope of Research

Focus on addressing challenges in the growing Industrial IoT sector, emphasizing privacy, efficient storage, and retrieval of IIoT data.

Aim to provide insights into Blockchain layers, optimizing energy consumption, and integration with IIoT environments.

9.4 Research Motivations

9.4.1 Privacy and Trust Mechanisms

Rapid growth in IIoT faces challenges like single points of failure, lack of transparency, privacy risks, and security vulnerabilities.

Blockchain technologies explored to ensure privacy and trust, creating a secure system for storing, processing, and retrieving information in IIoT.

9.4.2 Performance Evaluation

Blockchains gain popularity in enterprise use but incur performance overheads.

Investigation of performance evaluation metrics to optimize blockchain integration in power-constrained environments of IIoT.

9.5 Experimental Setup

9.5.1 Proof-of-Concept Implementation

For the practical validation of our proposed architecture, we have implemented a laboratory-scale TestBed. This experimental setup consists of three Raspberry Pi RPI400 devices and one laptop. Each Raspberry Pi is equipped with a 1.8GHz 64-bit quad-core ARM v8 CPU, a Micro Storage Card running Raspberry Pi OS, 4GB RAM, and essential wireless, Bluetooth, and other ports. These devices collectively function as data collection and processing modules within our proposed framework.

9.5.2 Key Components and Functions

The Raspberry Pi devices serve as the core components, responsible for initial data interaction with sensors, implementing processes such as data normalization, encryption, and processing as outlined in our framework. Their specifications are carefully chosen to ensure ample computing power, memory, and connectivity for seamless execution of the proposed architecture. The proof-of-concept aims to demonstrate the feasibility and functionality of the architecture in a controlled laboratory environment.

9.6 Experimental Procedure

In our proof-of-concept implementation, the TestBed integrates three Raspberry Pi RPI400 devices, each running blockchain clients such as geth v1.11.6. To simulate the interaction between nodes, we employed web3.js for submitting transactions and invoking smart contracts, utilizing JSON-RPC and bash scripts. Complementing these devices is an ASUS laptop, featuring 8GB RAM, a 2.1 GHz Octacore AMD Ryzen 5 processor, and running Windows 10. This laptop serves a dual role, hosting both the Private Blockchain Network and the User DApp.

The TestBed is configured to replicate real-world data collection and processing scenarios within the proposed architecture. This controlled environment facilitates systematic testing and refinement of the framework. The Raspberry Pi devices, equipped with wireless and Bluetooth capabilities, enhance communication channels, while standardized operating systems and storage ensure seamless compatibility.

9.7 Experimental Results and Analysis

9.7.1 Performance Evaluation and Proposed Architecture

The study comprehensively evaluates various performance metrics of blockchain-based systems, including block production time, confirmation time, system throughput, transaction cost, gas cost, and transaction latency. Results indicate that the Proposed Architecture consistently outperforms existing platforms across all parameters. Notably, the architecture addresses security concerns related to blockchain forks and double-spending attacks by achieving an average 2-second block production and confirmation time, enhancing real-time integration effectiveness. The system's throughput stands out, reaching nearly 4000, surpassing other mechanisms, and the proposed Harmony-based architecture demonstrates scalability potential up to 1 million transactions per second.

Furthermore, the investigation extends to transaction latency and cost measurements, showcasing the Proposed Architecture's superiority. With a 1-second transaction

latency and minimal gas consumption, the architecture aligns with the resource constraints of IIoT devices, making it ideal for resource optimization and real-time responsiveness. The study also delves into data storage and retrieval time measurements, revealing that the integration of IPFS in the proposed architecture ensures global data access without significant processing delays. Despite the impressive results, the study acknowledges the limitation of a homogeneous environment in the lab-testbed and outlines intentions to address this by designing and extensively testing the architecture in a heterogeneous environment. Additionally, future work aims to evaluate the impact of large-scale deployment on registration and authorization and improve resource utilization in the consensus and network layers.

9.7.2 Challenges and Future Directions

The integration of IIoT and blockchain presents challenges in computation, scalability, privacy, and storage. Existing solutions struggle to achieve higher throughput without compromising security and decentralization. The proposed architecture emerges as an optimized solution, showcasing the fastest block finality, minimal transaction cost, and the highest throughput while ensuring privacy. Lightweight encryption models and interoperability features contribute to the architecture's operational versatility. Future work involves implementing the proposed architecture in real-world scenarios for further performance evaluation, investigating the impact of registration and authorization in large-scale deployment, and enhancing resource utilization in the consensus and network layers. Additionally, the study aims to develop an end-to-end energy consumption measurement for the proposed architecture, ensuring a holistic understanding of its sustainability in diverse deployment scenarios.

10 Conclusions

In Section 3 an experiment to measure distance using sub 6GHz ToA was performed. It was found that signal multipath from UE transmissions of an isotropic antenna rendered the system to be inoperable. Using UE transmissions of a beamforming antenna produced a sufficiently good received signal for distance accuracy of between 3.5 to 4.5 cm accuracy was obtained. Close analysis of the frequency distribution of the measured data recorded Standard Deviations of around 1.7 cm which corresponded to a 95% Confidence interval of 3.4 cm accuracy. This largely agreed with the distance accuracies that was measured. This accuracy was obtained despite the presence of periodic noise which was suspected to have been induced from either flatbed plotter stepper motor, or cooling fan motor or induced internally within the RU FPGA.

In section 4 an object detector (YOLOv5) was paired with a LIDAR to produce a location from landmarks system. This detector was trained to find certain passive landmarks within this environment. The detector then communicates with this LIDAR to measure distances from the closest point of these objects, and provided that these objects coordinates were already measured, a trilateration algorithm was able to find the coordinates of the UE within the environment.

Upon completion of this system, testing was conducted to determine whether this is a viable option for localisation. Although the system passed its functional requirements, it did not meet the KPIs. When tested against 16 different positions, 15% of the estimated coordinates were deviated by 5%. This was not the complete case however, as there were also occasions where accuracy was below 1% for both x and y coordinates in certain positions. A list of potential points of failures were drawn with the major issue being the lack of detail for YOLOv5 to accurately detected images upon. A potential solution to this would be to introduce sensor fusion with a 360-degree camera. The object detector could work on the higher quality camera images, and then pass on the bounding box data to the LIDAR. The LIDAR would then have to simply measure distances, ultimately resulting in better position estimations.

Overall, the system has the potential to estimate positions of LIDAR-equipped UEs but requires further assistance in terms of better visual sensors, such as a camera to become reliable.

In section 5 the localization performance of the fusion technology is analysed and improved by a novel hybrid localization system. The proposed localization system uses a decision process to choose which localization technologies to apply. Simulation results verify the effectiveness of the proposed system. In the future, the NN generalization will be studied and fulfilled to generalize the model used for localization and adapt it to different environments. We intend also to validate our model based on experimental real data collected in our laboratory.

In section 6 the localization performance of the FL and CL models was compared. The training of models demonstrated encouraging progress, with both training and validation losses steadily decreasing over the epochs. The FL model, executed for 1000 epochs, showcased impressive improvements in localization accuracy. The model's localization error significantly reduced from an initial high of approximately 30 meters to almost 10 meters at the end of the validation phase. Also, the CL model demonstrated steady progress reaching around 5 meters by the conclusion of the 1000 epochs.

Our proposed algorithm is going to be validated and tested using real data collected in Bosch Germany as mentioned before in order to prove that it can be adapted to different types of input data independently from the used communication technology.

In section 7, a novel indoor beacon construction and low sampling rate positioning scheme using on-off keying (OOK) modulation pulse pairs has been proposed. This scheme, unlike time of arrival (TOA)-based schemes, doesn't require strict time synchronization, and it achieves high positioning accuracy with low sampling rates and bandwidth. The OOK modulation, beneficial for its minimal modulation bandwidth, is employed in VLP systems. The OOK-based pulse pairs are designed to form positioning beacon signals without synchronization needs. Additionally, a high-precision pulse reconstruction method is introduced to counteract the distortion from low sampling rates and reduce positioning errors due to time measurement inaccuracies. This research has been recognized at the International Conference on Indoor Positioning and Indoor Navigation (IPIN) 2023, where it received the 4th best paper award.

In section 8 a messaging scheme based on public/private key pairs was proposed that distributes identity information as well as localization data where both fixed and moving IoT devices can serve as trilateration anchors for other devices. The application of the scheme has been simulated for different configurations of number of devices, communication ranges, and different obstructedness levels and shows significant improvements in localization success compared to fixed-anchors-only trilateration while discussing the trade-off as to the increased messaging effort the scheme introduces.

In section 9 the integration of IIoT and blockchain presents challenges in computation, scalability, privacy, and storage. Existing solutions struggle to achieve higher throughput without compromising security and decentralization. The proposed architecture emerges as an optimized solution, showcasing the fastest block finality, minimal transaction cost, and the highest throughput while ensuring privacy. Lightweight encryption models and interoperability features contribute to the architecture's operational versatility. Future work involves implementing the proposed architecture in real-world scenarios for further performance evaluation, investigating the impact of registration and authorization in large-scale deployment, and enhancing resource utilization in the consensus and network layers. Additionally, the study aims to develop an end-to-end energy consumption measurement for the proposed architecture, ensuring a holistic understanding of its sustainability in diverse deployment scenarios.

References

- [1-1] Alexander Artemenko et al “Definition and Description of the 6G BRAINS Primary Use Cases and Derivation of User Requirements” 6G BRAINS, Deliverable 2.1, 31.07.2021
- [3-1] Xun Zhang et al. “3D Location Simulation Models and Lab Prototypes” 6G BRAINS Deliverable 6.2, 31st January 2023
- [3-2] Israel Koffman et al “Final specification and evaluation of human-centric control interfaces for advanced industrial scenarios” 6G BRAINS Deliverable 4.4, 01.08.2023
- [3-3] Clare Somerville et al. “5G FAPI: RF and Digital Frontend Control API” DOCUMENT 222.07.00, August 2023
- [4-1]. “Life of a bot: building a mobile robot using automated solutions,” Ocado Group. <https://www.ocadogroup.com/technology/blog/life-bot-building-mobile-robot-using-automated-solutions>
- [4-2]. J. A. del Peral-Rosado, R. Raulefs, J. A. López-Salcedo and G. Seco-Granados, "Survey of cellular mobile radio localization methods: From 1G to 5G", *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1124-1148, 2nd Quart. 2017.
- [4-3]. R. Di Taranto, S. Muppisetty, R. Raulefs, D. Slock, T. Svensson and H. Wymeersch, "Location-aware communications for 5G networks: How location information can improve scalability latency and robustness of 5G", *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 102-112, Nov. 2014.
- [4-4]. G. Bresson, Z. Alsayed, L. Yu and S. Glaser, "Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving," in *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194-220, Sept. 2017, doi: 10.1109/TIV.2017.2749181.
- [4-5]. E. S. Lohan et al., "Benefits of Positioning-Aided Communication Technology in High-Frequency Industrial IoT," in *IEEE Communications Magazine*, vol. 56, no. 12, pp. 142-148, December 2018, doi: 10.1109/MCOM.2018.1701057.
- [4-6]. K. Antonakoglou, X. Xu, E. Steinbach, T. Mahmoodi and M. Dohler, "Toward Haptic Communications Over the 5G Tactile Internet," in *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3034-3059, Fourthquarter 2018, doi: 10.1109/COMST.2018.2851452.
- [4-7]. N. M. Drawil, H. M. Amar, and O. A. Basir, “GPS Localization Accuracy Classification: A Context-Based Approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 262–273, Mar. 2013, doi: <https://doi.org/10.1109/tits.2012.2213815>.
- [4-8]. O. Kanhere and T. S. Rappaport, “Position Location for Futuristic Cellular Communications: 5G and Beyond,” *IEEE Communications Magazine*, vol. 59, no. 1, pp. 70–75, Jan. 2021, doi: <https://doi.org/10.1109/mcom.001.2000150>.
- [4-9]. J. Gante, G. Falcão, and L. Sousa, “Deep Learning Architectures for Accurate Millimeter Wave Positioning in 5G,” *Neural Processing Letters*, vol. 51, no. 1, pp. 487–514, Aug. 2019, doi: <https://doi.org/10.1007/s11063-019-10073-1>.
- [4-10]. H. Chen, H. Sameddeen, T. Ballal, H. Wymeersch, M.-S. Alouini, and T. Y. Al-Naffouri, “A Tutorial on Terahertz-Band Localization for 6G Communication Systems,” *IEEE*

- Communications Surveys & Tutorials, pp. 1–1, 2022, doi: <https://doi.org/10.1109/comst.2022.3178209>.
- [4-11]. S. Fan, Y. Wu, C. Han and X. Wang, "A Structured Bidirectional LSTM Deep Learning Method For 3D Terahertz Indoor Localization," IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, Toronto, ON, Canada, 2020, pp. 2381-2390, doi: [10.1109/INFOCOM41043.2020.9155458](https://doi.org/10.1109/INFOCOM41043.2020.9155458).
- [4-12]. Y. Lee and D. Lim, "Vision/UWB/IMU sensor fusion based localization using an extended Kalman filter," 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2019, pp. 401-403, doi: [10.1109/ECICE47484.2019.8942733](https://doi.org/10.1109/ECICE47484.2019.8942733).
- [4-13]. A. Muis and R. Prasetya, "Implementation of landmark-based localization on wall-based environment for mobile robot with limited sensor resources," TENCON 2011 - 2011 IEEE Region 10 Conference, Bali, Indonesia, 2011, pp. 1050-1054, doi: [10.1109/TENCON.2011.6129271](https://doi.org/10.1109/TENCON.2011.6129271).
- [4-14]. T. H. Chan, H. Hesse and S. G. Ho, "LIDAR-Based 3D SLAM for Indoor Mapping," 2021 7th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 2021, pp. 285-289, doi: [10.1109/ICCAR52225.2021.9463503](https://doi.org/10.1109/ICCAR52225.2021.9463503).
- [4-15]. T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized LIDAR Odometry and Mapping on Variable Terrain," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 4758-4765, doi: [10.1109/IROS.2018.8594299](https://doi.org/10.1109/IROS.2018.8594299).
- [4-16]. J. Feng, L. Wang, J. Li, Y. Xu, S. Bi and T. Shen, "Novel LIDAR-assisted UWB positioning compensation for indoor robot localization," 2021 International Conference on Advanced Mechatronic Systems (ICAMechS), Tokyo, Japan, 2021, pp. 215-219, doi: [10.1109/ICAMechS54019.2021.9661496](https://doi.org/10.1109/ICAMechS54019.2021.9661496).
- [4-17]. D. R. -Y. Phang, W. -K. Lee, N. Matsuhira and P. Michail, "Enhanced Mobile Robot Localization with LIDAR and IMU Sensor," 2019 IEEE International Meeting for Future of Electron Devices, Kansai (IMFEDK), Kyoto, Japan, 2019, pp. 71-72, doi: [10.1109/IMFEDK48381.2019.8950726](https://doi.org/10.1109/IMFEDK48381.2019.8950726).
- [4-18]. M. Sefati, M. Daum, B. Sundermann, K. D. Kreisköther and A. Kampker, "Improving vehicle localization using semantic and pole-like landmarks," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 2017, pp. 13-19, doi: [10.1109/IVS.2017.7995692](https://doi.org/10.1109/IVS.2017.7995692).
- [4-19]. H. Lezki and İ. Ş. Yetik, "Localization Using Single Camera and LIDAR in GPS-Denied Environments," 2020 28th Signal Processing and Communications Applications Conference (SIU), Gaziantep, Turkey, 2020, pp. 1-4, doi: [10.1109/SIU49456.2020.9302512](https://doi.org/10.1109/SIU49456.2020.9302512).
- [4-20]. Q. Zou, Q. Sun, L. Chen, B. Nie and Q. Li, "A Comparative Analysis of LIDAR SLAM-Based Indoor Navigation for Autonomous Vehicles," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 6907-6921, July 2022, doi: [10.1109/TITS.2021.3063477](https://doi.org/10.1109/TITS.2021.3063477).
- [4-21]. W. Wibisono and A. D. Wicaksono, "An Improved Accuracy of Indoor Positioning System Based on Trilateration using Kalman Filter," 2022 10th International Conference

- on Information and Communication Technology (ICoICT), Bandung, Indonesia, 2022, pp. 174-178, doi: 10.1109/ICoICT55009.2022.9914863.
- [4-22]. J. Machaj, P. Brida and R. Piché, "Rank based fingerprinting algorithm for indoor positioning," 2011 International Conference on Indoor Positioning and Indoor Navigation, Guimaraes, Portugal, 2011, pp. 1-6, doi: 10.1109/IPIN.2011.6071929.
- [4-23]. Boesch, G. (2021). Object Detection in 2021: The Definitive Guide. [online] viso.ai. Available at: <https://viso.ai/deep-learning/object-detection/>.
- [4-24]. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.
- [4-25]. R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.
- [4-26]. COMPANY, R.I.C.O.H. (2017) RICOH THETA V, RICOH THETA V. Available at: <https://theta360.com/en/about/theta/v.html> (Accessed: March 12, 2023).
- [4-27]. Ouster (2021) High-resolution OS1 LIDAR SENSOR: Robotics, trucking, mapping, Ouster. Available at: <https://ouster.com/products/scanning-LIDAR/os1-sensor/> (Accessed: March 13, 2023).
- [4-28]. Shi, F. (2022) Object detection and tracking using deep learning and ouster python SDK, Ouster. Available at: <https://ouster.com/blog/object-detection-and-tracking-using-deep-learning-and-ouster-python-sdk/> (Accessed: March 13, 2023).
- [4-29]. Dell (2022) Poweredge R740 Rack Server: Dell Middle East, Dell. Available at: <https://www.dell.com/ae/business/p/poweredge-r740/pd> (Accessed: March 14, 2023).
- [4-30]. C. -Y. Wang, H. -Y. Mark Liao, Y. -H. Wu, P. -Y. Chen, J. -W. Hsieh and I. -H. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 1571-1580, doi: 10.1109/CVPRW50498.2020.00203.
- [4-31]. Solawetz, J. (2023) How to train yolov5 on a custom dataset, Roboflow Blog. Roboflow Blog. Available at: <https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/> (Accessed: March 16, 2023).
- [4-32] Prabhveer Mujral "Position estimation in known environments using LIDAR and Sensor Fusion" BEng Final Year Project Report, Brunel University London, 2023.
- [7-1] K. Zhang, Z. Zhang, and B. Zhu, "Beacon LED coordinates estimator for easy deployment of visible light positioning systems," IEEE Trans. Wireless Commun., vol. 21, no. 12, pp. 10208-10223, Dec. 2022.
- [7-2] M. F. Keskin, A. D. Sezer, and S. Gezici, "Localization via visible light systems," Proc. IEEE., vol. 106, no. 6, pp. 1063-1088, Jun. 2018.
- [7-3] X. Liu, L. Guo, H. Yang, and X. Wei, "Visible light positioning based on collaborative LEDs and edge computing," IEEE Trans. Computat. Soc. Syst., vol. 9, no. 1, pp. 324-335, Feb. 2022.

- [7-4] Y. Wang, X. Chen, X. Liu, Z. Wang, L. Yu, and X. Zhang, "Experimental testing of high-capacity bandwidth efficient visible light communication with silicon-based RGBY-LED," 2021 IEEE International Symposium on BMSB, Chengdu, China, 2021, pp. 1-3.
- [7-5] S. Tacolu, M. Kse, and Z. Telatar, "Effect of sampling rate on transient based RF fingerprinting," Int. Conf. Electr. Electron. Eng. ELECO., Bursa, Turkey, 2017, pp. 1156-1160.
- [7-6] B. Soner and S. C. Ergen, "Vehicular visible light positioning with a single receiver," IEEE Int. Symp. Person Indoor Mobile Radio Commun., Istanbul, Turkey, 2019, pp. 1-6.
- [7-7] X. Huang, X. Pan, Z. Wan, M. Xu, Z. Wang, X. Liu, Y. Wang, and X. Zhang, "A novel experimental visible light positioning system with low bandwidth requirement and high precision pulse reconstruction" International Conference on Indoor Positioning and Indoor Navigation (IPIN) 2023, to be published, Nuremberg, Germany.
- [7-8] T. Liu, T. Jiang, C. Chung, and Y. Chu, "A maximum logarithmic maximum a posteriori probability based soft-input soft-output detector for the coded spatial modulation systems," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 69, no. 9, pp. 3816-3828, Sept. 2022.
- [8-1] H. Rashid, A.K. Turuk (2015), "Dead reckoning localisation technique for mobile wireless sensor networks", in IET Wireless Sensor Systems, vol. 5, no. 2, pp. 87-96, Apr. 2015, doi: 10.1049/iet-wss.2014.0043
- [8-2] F. Höflinger, R. Zhang, L. M. Reindl (2012), "Indoor-localization system using a Micro-Inertial Measurement Unit (IMU)", 2012 European Frequency and Time Forum, Gothenburg, Sweden, 2012, pp. 443-447, doi: 10.1109/EFTF.2012.6502421
- [8-3] M. N. Rahman, M. T. I. A. T. Hanuranto and S. T. M. T. R. Mayasari, "Trilateration and iterative multilateration algorithm for localization schemes on Wireless Sensor Network," 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), Yogyakarta, Indonesia, 2017, pp. 88-92, doi: 10.1109/ICCEREC.2017.8226710
- [8-4] M. Weh et al. "Mutual Localisation Sensing Lab prototype", in 6GBRAINS Deliverable 6.2 "3D Location Simulation Models and Lab Prototypes", Jan 2023

Appendix I Cassino Cards setup

1.1 RunEL Cards Driver (Cassino Cards)

Open directory with name cassino_26 and copy all the driver files into this directory, get the files from RunEL.

Go to xdma_cassino_v537/xdma and do make and sudo make install.

Go to xdma_cassino_v537/tools and do make and sudo make install.

Go to cassino_26/rawcardtool_cassino_v1.3.0 and do make and sudo make install.

Reboot system, after reboot do this:

#First remember to load xdma drivers every time after reboot:

```
cd cassino_26/xdma_cassino_v537/tools/
```

```
sudo ./load_driver.sh
```

you should get DONE or OK , if problem go to directory

xdma_cassino_v537/xdma and do make clean and make and sudo make

install, no need to reboot again, just go to xdma_cassino_v537/tools/ and

again sudo ./load_driver.sh.

To check if Cassino cards ok do this:

Go to cassino_26/rawcardtool_cassino_v1.3.0 and run:

For first card run this:

```
sudo ./rawcardtool_cassino --device /dev/xdma0_user --status
```

For Second card run this:

```
sudo ./rawcardtool_cassino --device /dev/xdma0_user --status
```

1.2 Ethernet fifo Cards Driver (Cassino Cards)

To connect Cassino cards from Linux we are using special drivers, open cassino_32 and copy all the driver files into this directory, get the files from RunEL.

#To compile drivers for fifo(xdma0) to host do this:

#First remember to load xdma drivers:

```
cd cassino_26/xdma_cassino_v537/tools/
```

```
sudo ./load_driver.sh
```

#After load xdma drivers from cassino_26 go to cassino_32 dir:

```
cd /home/runel/cassino_32/axis-fifo-tools/files
```

```
make
```

```
sudo make install-host
```

```
sudo install -m 0755 axis-fifo-send /usr/local/bin/
```

```
sudo install -m 0755 axis-fifo-recv /usr/local/bin/
```

```
sudo install -m 0755 axis-fifo-eth /usr/local/bin/
```

```
sudo test -d /etc/systemd/system && sudo install -m 0644 init.d/axis  
fifo-eth.service /etc/systemd/system
```

```
sudo test -d /etc/systemd/system && systemctl daemon-reload
```

```
sudo systemctl start axis-fifo-eth
```

```
sudo ip addr add dev asf0 172.31.0.10/24
```

```
sudo ip link set dev asf0 up
```

#Check if asf0 exist:

```
ifconfig
ping 172.31.0.80
#With puty ssh to 172.31.0.80 , save it as DU_RX
#To copy files to ps , with the ssh to PS mkdir rcv (or other name)
scp -r * root@172.31.0.80:/home/root/rcv
#To compile drivers for fifo1(xdma1) to host do this first if need to load
drivers:
#First remmber to load xdma drivers:
cd cassino_26/xdma_cassino_v537/tools/
sudo ./load_driver.sh
#After load xdma drivers from cassino_26 go to cassino_32 dir:
cd /home/runel/cassino_32/axis-fifo1-tools/files
make
sudo make install-host
sudo install -m 0755 axis-fifo1-send /usr/local/bin/
sudo install -m 0755 axis-fifo1-recv /usr/local/bin/
sudo install -m 0755 axis-fifo1-eth /usr/local/bin/
sudo test -d /etc/systemd/system && sudo install -m 0644 init.d/axis
fifo1-eth.service /etc/systemd/system
sudo test -d /etc/systemd/system && systemctl daemon-reload
sudo systemctl start axis-fifo1-eth
sudo ip addr add dev asf1 172.31.1.11/24
sudo ip link set dev asf1 up
#Check if asf1 exist:
ifconfig
ping 172.31.1.100
#With puty ssh to 172.31.1.100, save it as DU_TX
#To copy files to ps , with the ssh to PS mkdir rcv (or other name)
scp -r * root@172.31.1.100:/home/root/rcv
```

1.3 After boot always do just this:

```
#First remember to load xdma drivers:
cd cassino_26/xdma_cassino_v537/tools/
sudo ./load_driver.sh
#Note: if problem go to xdma dir, make clean, make, sudo make install ,
do same in tools dir
sudo systemctl start axis-fifo-eth
sudo ip addr add dev asf0 172.31.0.10/24
sudo ip link set dev asf0 up
sudo systemctl start axis-fifo1-eth
sudo ip addr add dev asf1 172.31.1.11/24
sudo ip link set dev asf1 up
#Open Putty and ssh to 172.31.0.80 (or load DU_RX if already saved).
#Only if need (usually no need), Open Putty and ssh to 172.31.1.100(or
load DU_TX if already saved).
```

```
runel@runel-PowerEdge-XR11:~$ cd cassino_26/xdma_cassino_v537/tools/
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo ./load_driver.sh
[sudo] password for runel:
Loading xdma driver...
The Kernel module installed correctly and the xdma devices were recognized.
DONE
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo systemctl
start axis-fifo-eth
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo ip addr ad
d dev asf0 172.31.0.10/24
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo ip link se
t dev asf0 up
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo systemctl
start axis-fifo1-eth
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo ip addr ad
d dev asf1 172.31.1.11/24
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$ sudo ip link se
t dev asf1 up
runel@runel-PowerEdge-XR11:~/cassino_26/xdma_cassino_v537/tools$
```


Appendix II Power up DU & RU Sequence.

Power first LINUX server and immediately after RU0/RU1

Both RU should connect to same power on button.

Do 1.3 section above.

#Open Terminal and run if only on RU as TX and RX:

./ping.sh

#If we have 2xRU:

./ping_ru1.sh

```
runel@runel-PowerEdge-XR11:~$ ./ping_ru1.sh
[sudo] password for runel:
ARPING 10.0.0.141 from 10.0.0.103 eno8603np3
Unicast reply from 10.0.0.141 [00:0A:35:03:00:24] 0.562ms
Unicast reply from 10.0.0.141 [00:0A:35:03:00:24] 0.546ms
Unicast reply from 10.0.0.141 [00:0A:35:03:00:24] 0.534ms
Sent 3 probes (1 broadcast(s))
Received 3 response(s)
ARPING 10.0.0.134 from 10.0.0.103 eno8603np3
Unicast reply from 10.0.0.134 [00:0A:35:03:00:12] 0.559ms
Unicast reply from 10.0.0.134 [00:0A:35:03:00:12] 0.542ms
Unicast reply from 10.0.0.134 [00:0A:35:03:00:12] 0.530ms
Sent 3 probes (1 broadcast(s))
Received 3 response(s)
ARPING 10.0.0.136 from 10.0.0.103 eno8603np3
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.578ms
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.609ms
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.542ms
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.576ms
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.541ms
Unicast reply from 10.0.0.136 [00:0A:35:03:00:09] 0.575ms
Sent 3 probes (1 broadcast(s))
Received 6 response(s)
```

#Open Putty serial interface to /dev/ttyUSB0 (speed 115200)

#Login: root/root

ls

#if you see "i_m_rrh0" it means it is RU0 (RU TX)

#if you see "i_m_rrh1" it means it is RU1 (RU RX)

#Open Putty serial interface to /dev/ttyUSB4 (speed 115200)

#Login: root/root

ls

#if you see "i_m_rrh0" it means it is RU0 (RU TX)

#if you see "i_m_rrh1" it means it is RU1 (RU RX)

Note: /dev/ttyUSB0 or 4 can be 8 or 12

#Open Putty and ssh to 172.31.0.80 (or load DU_RX if already saved).

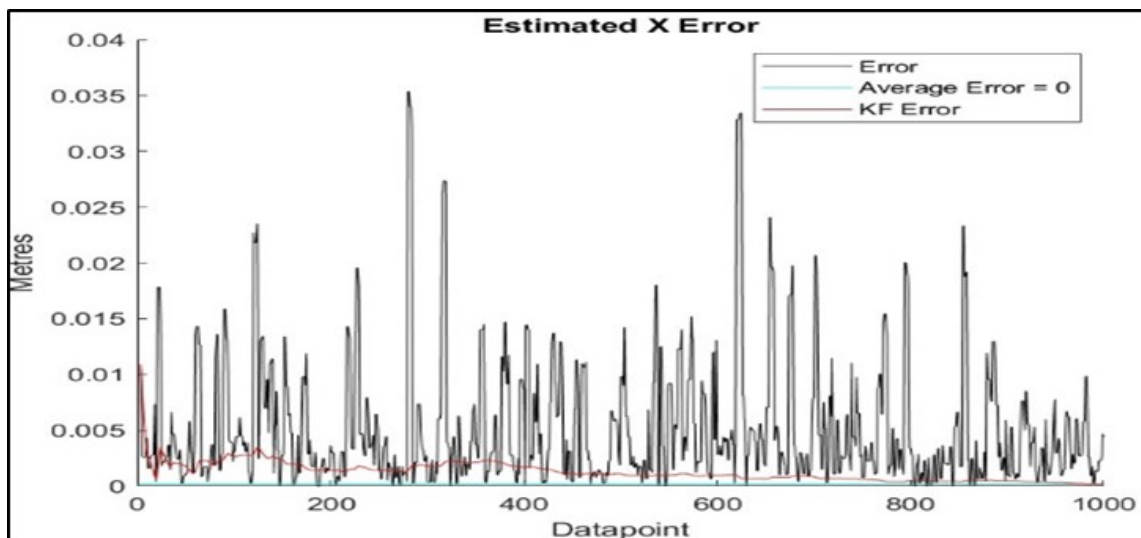
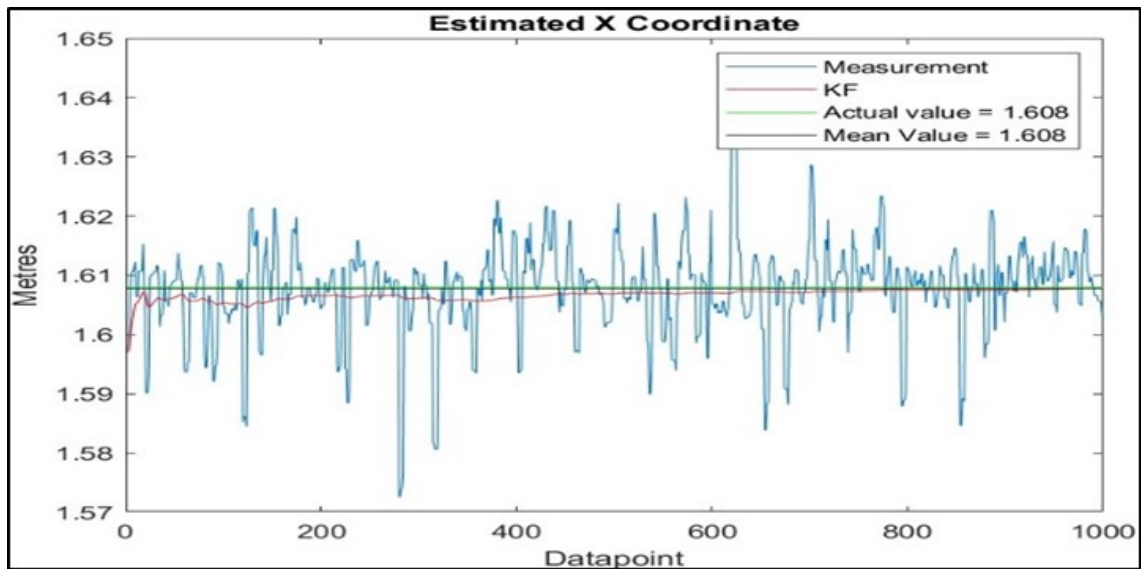
Appendix III Operate DU and RU for Distance measurement

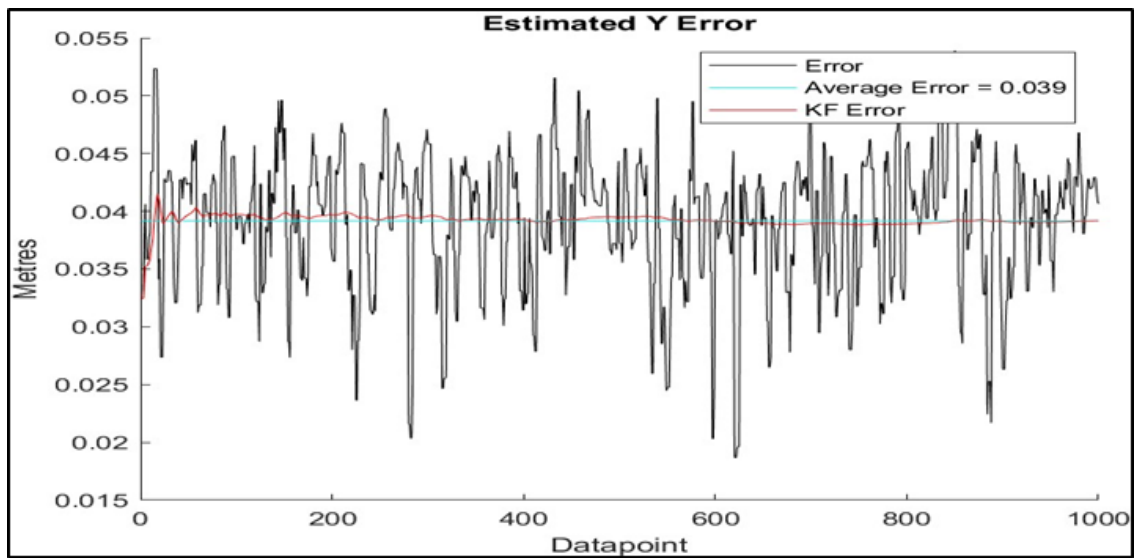
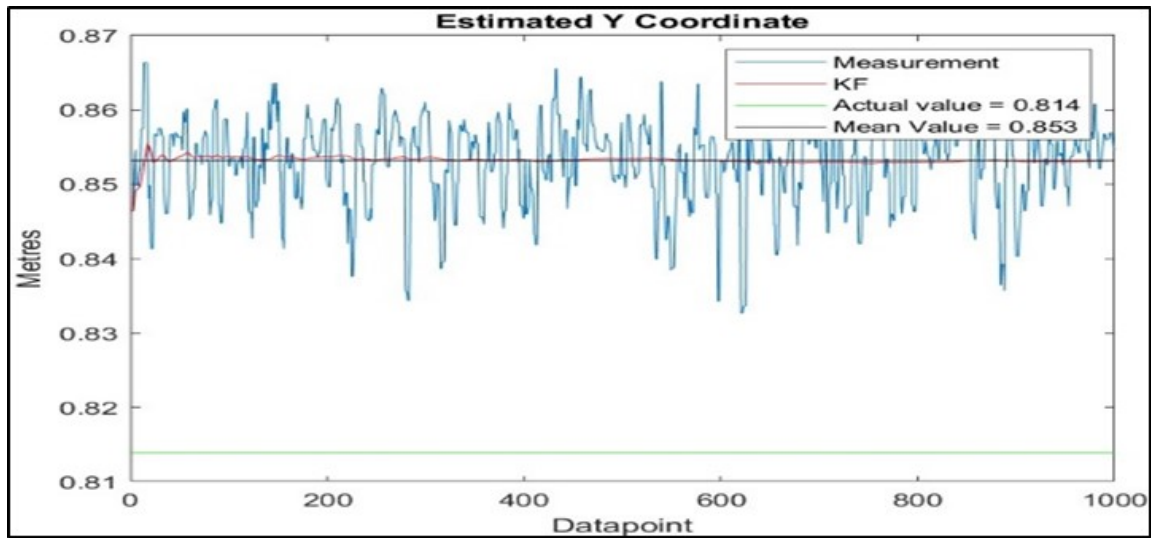
```
#On DU server, open new terminal, and type from downlink:
cd /home/runel/almog/du_downlink_test
#Open new tab in terminal and type for uplink only when one DU as TX
and RX (RU0 as TX and RX):
cd /home/runel/almog/git/uplink_test
#Upling when rx from second RU (RU1):
cd /home/runel/arik/full_spect_4rx_01_64_2xru/uplink_test/uplink_test
#open new tab for distance program:
cd /home/runel/almog/git/statistics_animate
#Please do in this orer
#1. In downlink tab type:
./run.sh
2. in distance program type:
./zion.sh or ./zion_nolna.sh in case of only one RU (RU0)
3. in uplink tab type:
./run.sh (password 1234 or another password if it will be changed).
```

Please note that after every powerup needs to calibrate again with: ./calib.sh

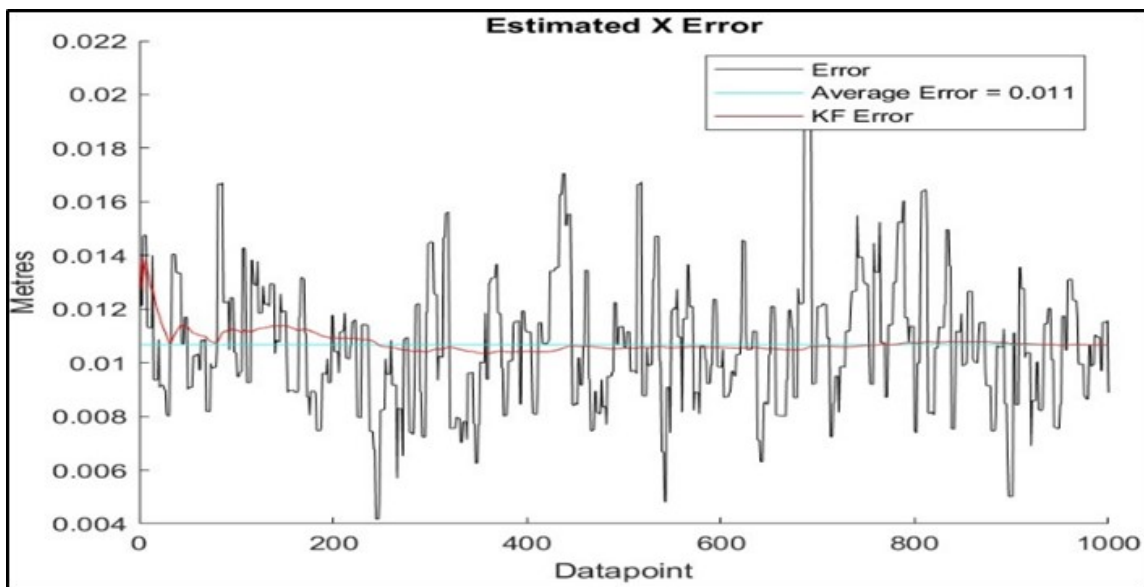
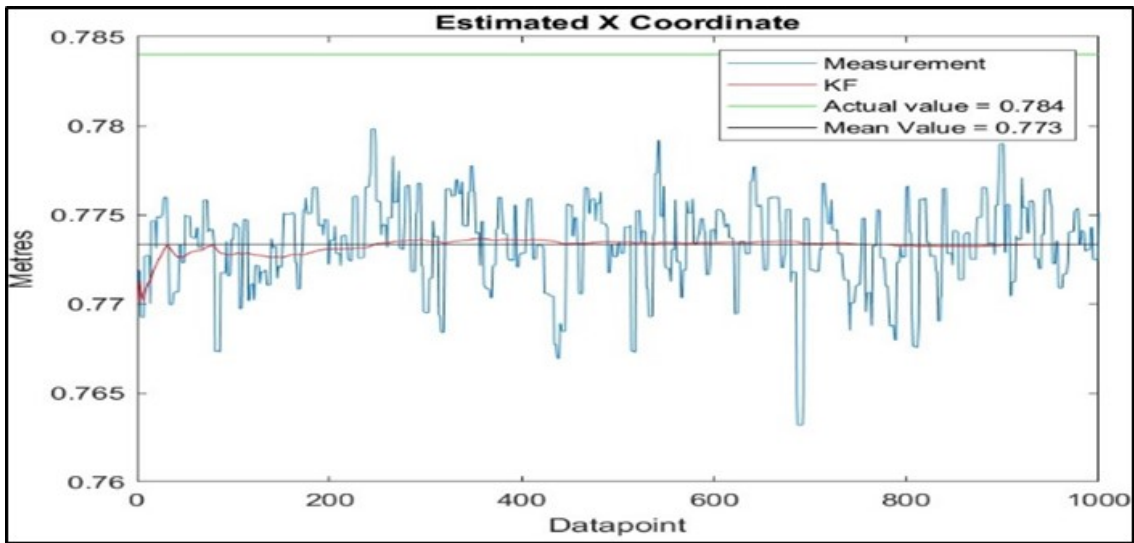
Appendix IV Position estimation

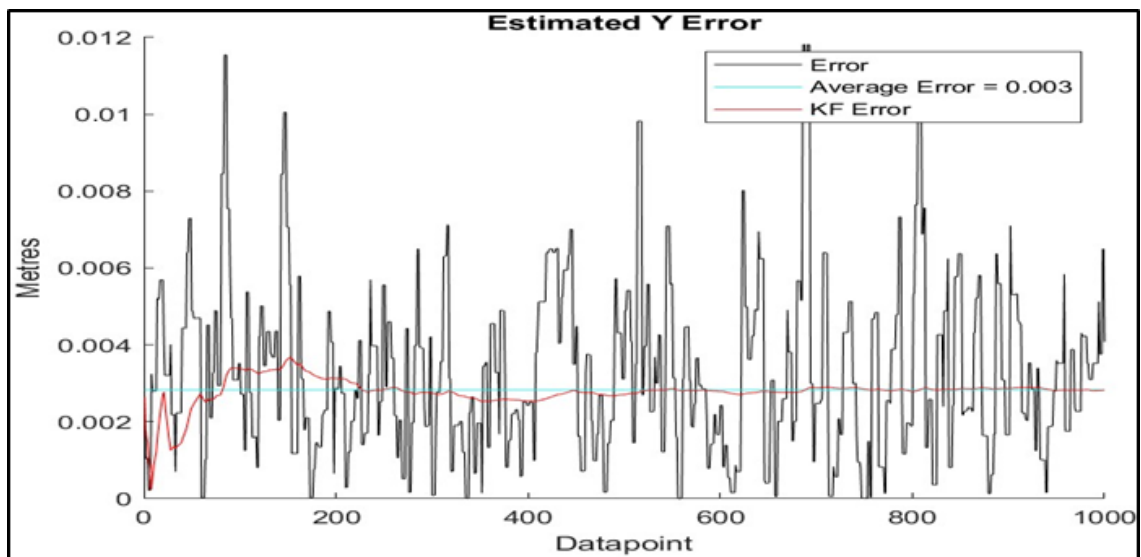
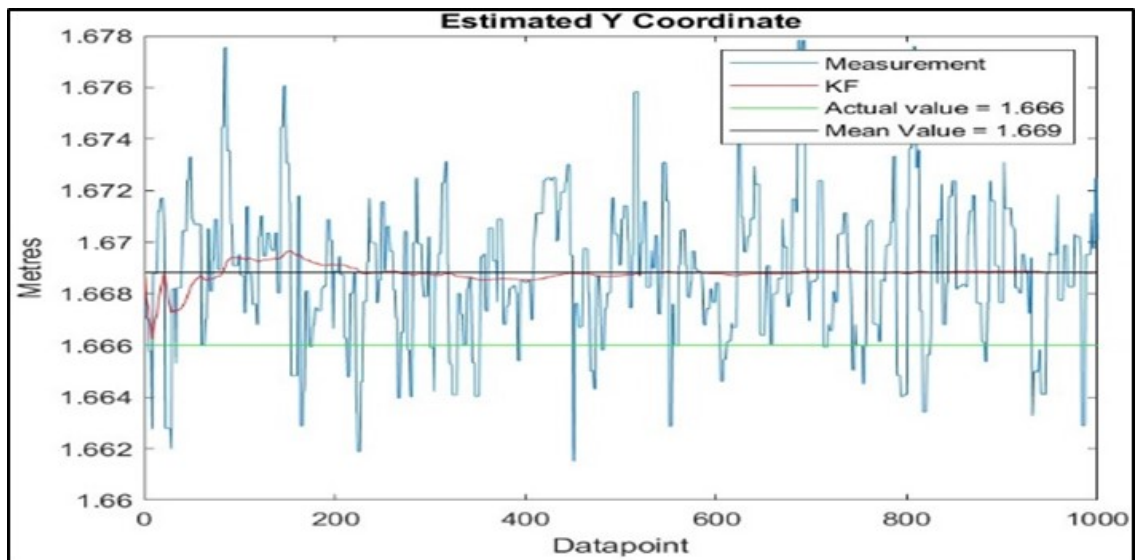
IV.1 Appendix IV-1. Position 1 Estimation



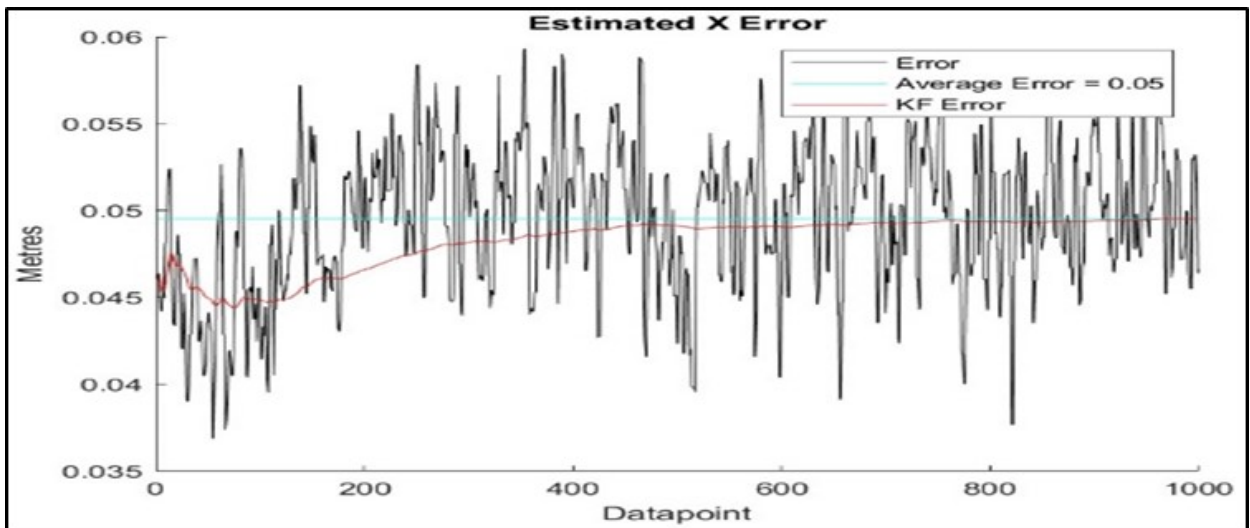
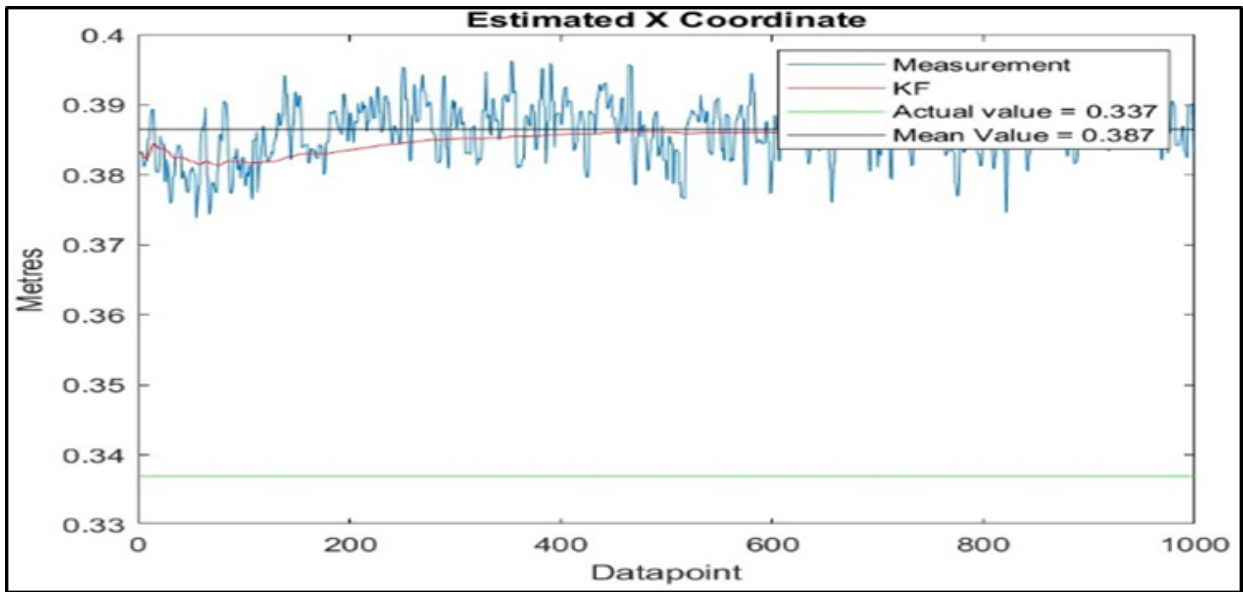


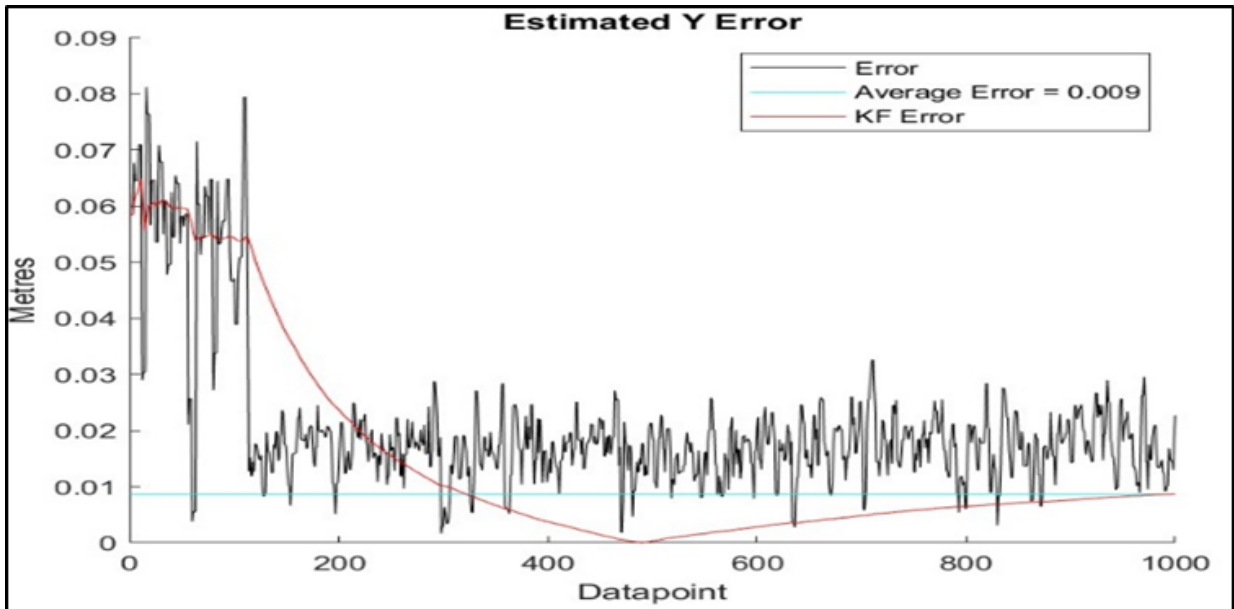
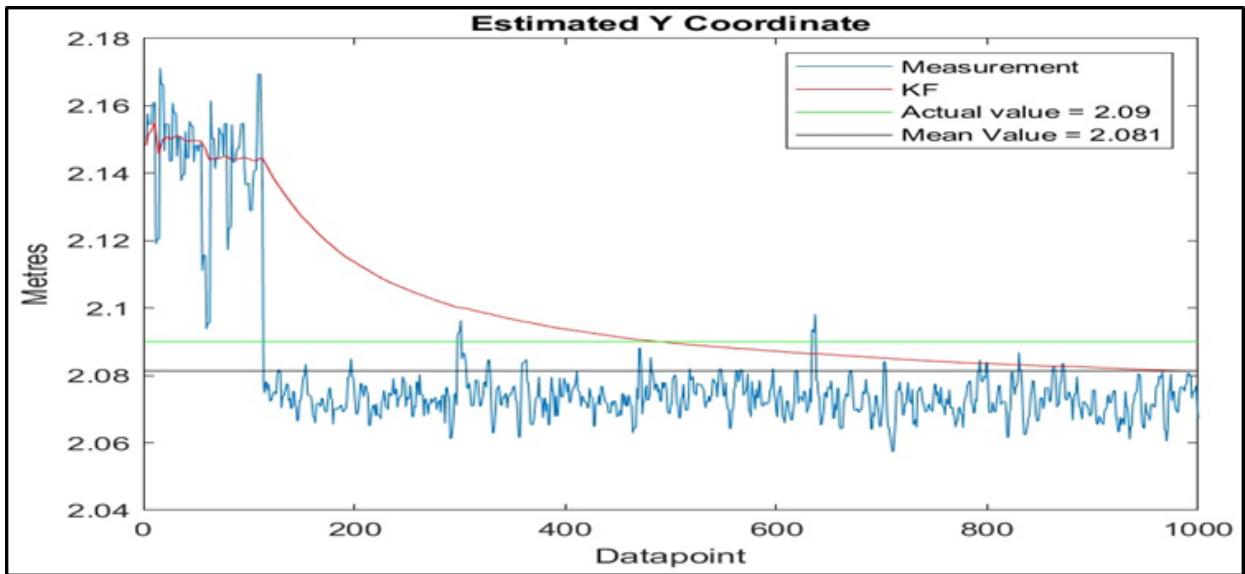
IV.2 Appendix IV-2. Position 11 Estimation





IV.3 Appendix IV-3. Position 16 Estimation





[end of document]