COOPERATIVE LOCALIZATION AND CONTROL FOR HETEROGENEOUS MOBILE ROBOTS

by

Ahmet Mustafa Kangal

B.S., Mechanical Engineering, Yıldız Technical University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Mechanical Engineering

Boğaziçi University

2024

ACKNOWLEDGEMENTS

Firstly, I would like to extend my deep gratitude to my advisor, Assist. Prof. Sinan Öncü, for his invaluable guidance and support throughout my research and studies. His insightful advice and encouragement were crucial in the development and refinement of my concepts. I would also like to express my thanks to Assoc. Prof. Evren Samur and Assoc. Prof. Volkan Sezer for their participation on my jury.

This work was supported by the UTOPIA Project, an Automated Open Precision Farming Platform, co-funded by TUBITAK 1071 (Project No: 120N785) and the European Union's Horizon 2020 research and innovation program under grant agreement No 862665 ERA-NET ICT-AGRI-FOOD. The Boğaziçi University Research Fund also provided additional support through the MARE (Multi-Agent Robot Eco-System (MARE) Development) Project (Grant Number 16883).

I am grateful for the invaluable support, insightful comments, and suggestions from my colleagues in the Smart and Autonomous Mobility Laboratory: Yağız Koç, Ender Yağcılar, and Efe Karcı. I also appreciate Derya Akkaya and Uğur Cucu for their significant support during my thesis. I would also like to thank the entire staff of the Department of Mechanical Engineering at Boğaziçi University.

Finally, I sincerely thank my family for their unwavering support and encouragement. This study would not have been possible without their love and support. This work is dedicated to them, especially to my beloved daughter, Ayla Larin, whose birth news I joyfully received while conducting test studies in the laboratory, and to the memory of my father, who passed away while I was pursuing my master's degree.

ABSTRACT

COOPERATIVE LOCALIZATION AND CONTROL FOR HETEROGENEOUS MOBILE ROBOTS

This thesis contributes to the UTOPIA project, which focuses on developing cooperative localization and control strategies for heterogeneous mobile robots in agriculture. The research involves designing an Unmanned Aerial Vehicle (UAV) and an Unmanned Ground Vehicle (UGV), each with distinct sensors and maneuvering capabilities. These differences require a robust localization strategy for effective cooperation. The Extended Kalman Filter (EKF) was tested in various configurations to improve localization accuracy. Enhancements were first achieved through simulations and later validated in experiments. The EKF's performance was assessed in three cooperative missions, showing that integrating more sensor data into the EKF significantly reduced the Root Mean Square Error (RMSE), improving localization accuracy. While homogeneous robot systems benefit from consistent localization with similar sensors, heterogeneous systems face challenges due to varying sensor capabilities. This research underscores the importance of robust localization for effective cooperation in diverse robotic systems. UAVs, despite their fast movement and wide vision, are limited by load capacity and battery life, whereas UGVs offer greater load capacity and battery longevity but move slower. These findings highlight the need for effective cooperative localization and control strategies in complex environments.

ÖZET

HETEROJEN MOBİL ROBOTLAR İÇİN İŞBİRLİKÇİ KONUMLANDIRMA VE KONTROL

Bu çalışma, tarım gibi zorlu ortamlarda farklı kabiliyetlere sahip mobil robotlar için işbirlikçi konumlandırma ve kontrol stratejileri geliştiren UTOPIA projesine katkıda bulunmaktadır. Araştırma, farklı sensör setleri ve manevra yeteneklerine sahip bir İnsansız Hava Aracı (İHA) ve İnsansız Kara Aracı (UGV) tasarlanmasını kapsamaktadır. Bu farklılıklar, etkili işbirliği için sağlam bir konumlandırma stratejisi gerektirmektedir. Genişletilmiş Kalman Filtresi (EKF) çeşitli konfigürasyonlarda test edilerek, simülasyonlar ve deneysel testlerle doğrulanan iyileştirmeler sağlanmıştır. EKF'nin performansı, üç ortak görevde simülasyonlar ve gerçek dünya senaryolarında değerlendirilmiş ve sensör verilerinin entegrasyonunun Ortalama Karekök Takip Hatasını (RMSE) önemli ölçüde azalttığı ve konumlandırma doğruluğunu artırdığı görülmüştür. Heterojen sistemlerde farklı sensör kapasiteleri nedeniyle daha büyük zorluklar yaşanmakta, bu araştırma ise çeşitli robotik sistemler için sağlam işbirlikçi konumlandırma ve kontrol stratejilerinin gerekliliğini vurgulamaktadır. İHA'lar hızlı hareket edebilir ve geniş görüş alanına sahipken, yük kapasitesi ve batarya ömrü sınırlıdır. UGV'ler ise daha yüksek yük kapasitesi ve batarya ömrü sunar ama hareket hızları daha düşüktür. Bu bulgular, karmaşık ortamlarda etkili işbirlikçi konumlandırma ve kontrol stratejilerinin geliştirilmesini teşvik etmektedir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES ix
LIST OF TABLES xv
LIST OF SYMBOLS xvii
LIST OF ACRONYMS/ABBREVIATIONS xxii
1. INTRODUCTION 1
1.1. Motivation and Background 3
1.1.1. UTOPIA-Automated Open Precision Farming Platform 7
1.2. Problem Statement and Objectives 11
1.3. Outline of the Thesis 13
1.4. Literature Survey 14
1.4.1. Methods of Localization 15
1.4.1.1. Sensor-Based Approaches 15
1.4.1.2. Simultaneous Localization and Mapping (SLAM) . . . 15
1.4.1.3. Filter-Based Techniques 17
1.4.1.4. Deep Learning Approaches 18
1.4.2. Cooperative Localization and Control 19
1.4.3. Contribution of This Thesis 20
2. UTOPIA SYSTEM 21
2.1. UTOPIA's Components-Hardware Design 21
2.1.1. Design of the UAV 21
2.1.1.1. Mechanical Design and Installation of the UAV 21
2.1.1.2. Electrical Design and Electronics of the UAV 26
2.1.2. Design of the UGV 31
2.1.2.1. Mechanical Design and Installation of the UGV 32
2.1.2.2. Electrical Design and Electronics of the UGV 34

2.2. UTOPIA's Navigation 39
2.2.1. Trajectory Generation 39
2.2.1.1. Trajectory Generation Algorithm 40
2.2.1.2. Path Smoothing 41
2.2.2. Sensor Fusion for Localization 44
2.2.2.1. Sensor Fusion with Extended Kalman Filter 44
2.2.2.2. Robot Operating System (ROS) 47
2.2.3. Improvement Localization for The UAV 50
2.2.3.1. Sensor Set of The UAV 50
2.2.3.2. ROS Configurations with EKF for UAV Localization . 51
2.2.4. Improvement Localization for The UGV 54
2.2.4.1. Sensor Set of The UGV 54
2.2.4.2. ROS Configurations with EKF for UGV Localization . 55
2.2.5. Control Theory 56
2.2.5.1. Proportional-Integral-Derivative (PID) Control 57
2.2.5.2. Model Predictive Control (MPC) 59
2.2.5.3. Kinematics and Dynamics Analysis of the UAV 60
2.2.5.4. MPC Settings for Altitude Controller 64
2.2.5.5. Kinematics and Dynamics Analysis of the UGV 65
2.3. UTOPIA Framework And Coordinate Transformations 68
2.4. UTOPIA's Cooperative Missions For Heterogeneous Mobile Robots . . 70
2.4.1. Cooperative Mission 1: Trajectory Generation from Waypoints for Another Robot to Follow 70
2.4.2. Cooperative Mission 2: Detection and Estimation of Localization for a Lost Robot Using Onboard Sensors and Cooperative Localization 72
2.4.2.1. QR Detection 73
2.4.2.2. Vector Calculation: 73
2.4.2.3. Angle Calculation: 74
2.4.2.4. Estimation of Localization: 74

2.4.3. Cooperative Mission 3: Ensuring Compliance with Safe Zone through Limited Movement of Robots at a Relative Distance . . 75
3. SIMULATION WORKS AND RESULTS 78
3.1. UTOPIA's Digital Twin 78
3.1.1. Gazebo Plugins 79
3.1.1.1. Gazebo Model Plugins 79
3.1.1.2. Gazebo Sensor Plugins 79
3.2. Test Trajectories 80
3.3. Simulation Works for The UAV 81
3.4. Simulation Works for The UGV 85
3.5. Cooperative Localization and Control-Mission 1 89
3.6. Cooperative Localization and Control-Mission 2 91
3.7. Cooperative Localization and Control-Mission 3 92
4. EXPERIMENTAL WORKS AND RESULTS 94
4.1. Preliminary Works for Experimental Tests 94
4.1.1. Setup for The UTOPIA's Global Network 94
4.1.2. SITL Tests of The UAV 96
4.1.3. HITL Tests of The UAV 96
4.1.4. Calibration of The UAV 97
4.1.5. Calibration of The UGV 99
4.1.6. Indoor Tests of The UGV 100
4.1.7. Outdoor Tests of The UGV 102
4.2. Cooperative Localization and Control-Mission 1 105
4.3. Cooperative Localization and Control-Mission 2 107
4.4. Cooperative Localization and Control-Mission 3 110
5. CONCLUSION 112
REFERENCES 114

LIST OF FIGURES

Figure 1.1. The Quadcopter system (UAV) developed for this thesis. 2

Figure 1.2. The differential drive ground vehicle system (UGV) developed for this thesis. 2

Figure 1.3. Illustration of control structure for a mobile robot. 6

Figure 1.4. The UTOPIA's vineyard suitcase field. 8

Figure 1.5. Illustration of the hierarchical structure of the UTOPIA devices. . 9

Figure 1.6. Illustration of operational flow and control system for UTOPIA project. 10

Figure 2.1. A 2D drawing of the UAV, highlighting its important dimensions in mm. 23

Figure 2.2. The CAD model of the UAV. 23

Figure 2.3. Behind the Scenes: Capturing Moments from the UAV Installation Process. 24

Figure 2.4. The results of the static and drop analyses for the 3D printed parts of the UAV. 26

Figure 2.5. The results of the flow analysis for the UAV. 26

Figure 2.6. The electronic schematic of the UAV. 27

Figure 2.7. The UAV's electronics. 31

Figure 2.8. The completed assembly of the UAV. 31

Figure 2.9. A 2D drawing of the UGV, highlighting its important dimensions in mm. 32

Figure 2.10. CAD model of the UGV. 33

Figure 2.11. Behind the Scenes: Capturing Moments from the UGV Installation Process. 33

Figure 2.12. The small UGV dor indoor tests(right one), and the completed assembly of the UGV (left one). 34

Figure 2.13. The electronic schematic of the UGV. 35

Figure 2.14. The main components of the UGV. 36

Figure 2.15. The Wieltra transmitter & receiver on the UGV. 39

Figure 2.16. The diagram illustrating trajectory generation. 40

Figure 2.17. Comparison between unsmoothed (left) and smoothed (right) corners. 44

Figure 2.18. The diagram illustrating sensor fusion. 46

Figure 2.19. The diagram illustrating the URDF model for the robots. 50

Figure 2.20. The illustration of sensor data transformations (TF) for the UAV. 52

Figure 2.21. The illustration of sensor data transformations (TF) for the UGV. 56

Figure 2.22. The illustration of the PID controllers. 58

Figure 2.23. Schematic diagram for the kinematics of a quadcopter. 61

Figure 2.24. Schematic diagram for the kinematics of a differential drive robot. 66

Figure 2.25. The illustration of a global framework for a multi-robot system. . 69

Figure 2.26. The Rviz visualization represents the coordinate transformations for the UAV (on the right), the UGV (in the middle), and the global framework (on the left), respectively. 70

Figure 2.27. The illustration of the cooperative mission 1. 71

Figure 2.28. The illustration of the cooperative mission 2. 72

Figure 2.29. The illustration of the vector calculation. 74

Figure 2.30. The illustration of the cooperative mission 3. 75

Figure 2.31. The illustration of calculation for R_d. 76

Figure 3.1. The digital twin of the UTOPIA project's environment. 78

Figure 3.2. The detailed demonstration of the test trajectories. 81

Figure 3.3. Simulation test scenes for the UAV. 82

Figure 3.4. The plotting of recorded data for the straight line and the L-shaped trajectories of the UAV. 83

Figure 3.5. The plotting of recorded data for the ellipse and 8-figure trajectories of the UAV. 84

Figure 3.6. Simulation test scenes for the UGV. 86

Figure 3.7. The plotting of recorded data for the straight line and the L-shaped trajectories of the UGV. 87

Figure 3.8. The plotting of recorded data for the ellipse and the 8-figure trajectories of the UGV. 88

Figure 3.9. The scenes of performing cooperative mission 1 on UTOPIA's digital twin. 90

Figure 3.10. The trajectory of cooperative mission 1 on UTOPIA's digital twin. 90

Figure 3.11. Localization plots of cooperative mission 1 on UTOPIA's digital twin. 90

Figure 3.12. The plotting of recorded data for the straight line trajectory of the UGV. 92

Figure 3.13. The plotting of recorded data for the straight line trajectory of the UGV. 92

Figure 3.14. The plotting of recorded data for the straight line trajectory of the UGV. 93

Figure 4.1. UTOPIA's global network. 95

Figure 4.2. The position track record of the HITL test. 97

Figure 4.3. The UAV's actuator settings are on the left, and the RC settings are on the middle, and the sensor calibrations are on the right. . . 97

Figure 4.4. The scenes from the UAV's calibration test flights. 98

Figure 4.5. The scenes from the UAV's camera and algorithm tests. 98

Figure 4.6. The scenes from the UGV's indoor calibration. 99

Figure 4.7. The scenes from the UGV's indoor tests. 100

Figure 4.8. The plots from the UGV's indoor tests. 101

Figure 4.9. The scenes from the UGV's outdoor tests. 103

Figure 4.10. The plots from the UGV's outdoor tests. 104

Figure 4.11. The vineyard use-case of UTOPIA. 105

Figure 4.12. The records for cooperative mission 1. 106

Figure 4.13. The records for cooperative mission 1. 106

Figure 4.14. The scene view of the tests for cooperative mission 2 on the site manager screen. 108

Figure 4.15. The records of the first experimental work for cooperative mission 2.109

Figure 4.16. The records of the second experimental work for cooperative mission 2. 110

Figure 4.17. The records (UGV's Position and A_d) of the experimental work for cooperative mission 3. 111

LIST OF TABLES

Table 1.1. Advantages and disadvantages of UAVs and UGVs 5

Table 2.1. Selected motors and propellers 25

Table 2.2. Components of the UAV 28

Table 2.3. Components of the UAV 36

Table 2.4. Sensor types and sources 51

Table 2.5. EKF-Applied localization source topics for the UAV 53

Table 2.6. Robot localization package-source configuration matrix 53

Table 2.7. Sensor data availability for the UAV's EKF localization 54

Table 2.8. Sensor types and sources 55

Table 2.9. EKF-Applied localization source topics for the UGV 55

Table 2.10. Sensor data availability for the UGV's EKF localization 56

Table 3.1. Simulation test parameters for trajectories 79

Table 3.2. PID parameters for the UAV on simulation 82

Table 3.3. Configurations for localization-UAV 83

Table 3.4. RMSE for the UAV's configurations and trajectories 85

Table 3.5. PID parameters for the UGV on simulation 86

Table 3.6. Configurations for localization-UGV 87

Table 3.7. RMSE for the UGV's configurations and trajectories 89

Table 3.8. RMSE values for cooperative mission 1 on simulation tests 91

Table 4.1. Configurations for localization for UTOPIA components 95

Table 4.2. PID parameters for the UGV on experimental tests 99

Table 4.3. Configurations for localization of the UGV for indoor tests 101

Table 4.4. RMSE values for indoor calibration tests of the UGV 102

Table 4.5. Localization configurations of the UGV on outdoor tests 103

Table 4.6. RMSE values for outdoor calibration tests of the UGV 104

Table 4.7. RMSE-Cooperative mission 1 on experimental tests 107

Table 4.8. Estimated UGV localization and recorded covariance 109

LIST OF SYMBOLS

A	State transition matrix
$\mathbf{A}$	State transition matrix of a hidden Markov model
A_c	System matrix
A_d	The set limit distance between the UAV and the UGV
A_{dis}	Discrete system matrix
A_t	The Jacobian matrix of the measurement model
A_t	The Jacobian matrix of the process model
Ah	Ampere per Hour
a_{ij}	Description of a_{ij}
a_x	Acceleration in the x coordinate
a_y	Acceleration in the y coordinate
$arctan()$	Arctangent operator
B	Control input matrix
B_c	Input matrix
B_{dis}	Discrete input matrix
$c()$	Cosine function operator
C_t	Control input at time t
$d()$	Derivative operator
d_e	Euclidean distance between points
e	Error signal
e_{ang}	Angular error signal
e_{lat}	Lateral error signal
e_{long}	Longitudinal error signal
Ed	The Euclidean distance
Ed_i	i_{th} Euclidean distance
F	Force vector
f	Force
F_g	LQR gain matrix

F_i	i_{th} Propeller's force that affects around z-axis of BFF
I	The identity matrix
K	Kalman gain
K_d	Derivative gain
K_i	Integral gain
K_p	Proportional gain
K_t	Kalman gain at time t
kg	Kilogram
m	Weight of the vehicle
mAh	Milliampere per Hour
n	Sample number
N_c	Control horizon length
N_p	Prediction horizon length
N_t	The measurement noise covariance at time t
p	Angular velocity around X-axis of BFF
p_i	The i_{th} point matrix
p_{in}	The input matrix of points
p_{out}	The output matrix of points
P_t	The covariance matrix at time t
Q	State weighting matrix
Q_e	State error weighting matrix
Q_t	The process noise covariance at time t
q	Angular velocity around y-axis of BFF
$\mathbf{R}$	Rotation matrix for coordinate transformation
R_d	The relative distance between the UAV and the UGV
R_e	Effort weighting
R_w	Control weighting matrix
R_{BI}	Rotation matrix from IF to BFF
R_{IB}	Rotation matrix from BFF to IF
R_ϕ	Rotation matrix for roll
R_r	Transformation matrix

R_ψ	Rotation matrix for yaw
R_θ	Rotation matrix for pitch
R_{ICR}	The distance from ICR
r	Angular velocity around z-axis of BFF
r_T	Radius of perpendicular corners
S	Distance between the center of the wheels of the UGV
$s()$	Sine function operator
^{T}	Transpose operator
T	The required minimum thrust for the UAV
T_s	Timestep
T_{samp}	Sampling time
t	Time
U	Control input
U_{ang}	Angular control input
U_{lat}	Lateral control input
U_{long}	Longitudinal control input
u	Translational velocity on X-axis of BFF
$u_{0\|t}, \ldots, u_{Nc-1\|t}$	Control inputs at time t over the prediction horizon
$u_{\max}$	Maximum control input constraint
$u_{\min}$	Minimum control input constraint
v	Translational velocity on y-axis of BFF
v_x	Velocity in the x coordinate
v_y	Velocity in the y coordinate
v_T	Velocity for trajectory
v_{left}	Translational velocity of the left wheel
v_{right}	Translational velocity of the right wheel
V_B	Translational velocity vector in BFF
V_I	Translational velocity vector in IF
w	Translational velocity on z-axis of BFF
w	Orientation
x	Position in the x-direction

$\dot{x}$	Velocity in the x-direction
$\ddot{x}$	Acceleration in the x-direction
x_i	i_{th} Position in the x-direction
X_i	i_{th} Position in the X-direction
$x_{k\|t}$	State vector at time $t + k$ given the current time t
$x_{\max}$	Maximum state constraint
$x_{\min}$	Minimum state constraint
X_{ref}	Reference position in the X-direction
$\hat{x}_0$	The initial state
$\hat{x}_t$	The predicted state at time t
$\hat{X}_t$	The updated state estimate at time t
y	Position in the y-direction
$\dot{y}$	Velocity in the y-direction
$\ddot{y}$	Acceleration in the y-direction
y_i	i_{th} Position in the y-direction
Y_i	i_{th} Position in the Y-direction
Y_{ref}	Reference position in the Y-direction
z	Position in the z-direction
$\dot{z}$	Velocity in the z-direction
$\ddot{z}$	Acceleration in the z-direction
z_t	The actual measurement at time t
α	Blending parameter *or* scale
$\beta_t(i)$	Backward variable
Δ	Difference between two variables
η	Orientation vector in IF
ξ	Position vector in IF
ω	Angular velocity vector in BFF
ω_{ugv}	Angular velocity of the UGV
∂	Partial derivative operator
ϕ	Roll

Φ	The process model
π	Pi number
ψ	Yaw
θ	Pitch
τ	Integration variable
$\int$	Integral operator
$\in$	Element of
$^\circ$	Degree
$\mathbb{R}^3$	Real numbers
$\dot{\phi}$	Angular velocity in Roll
$\dot{\psi}$	Angular velocity in Yaw
$\dot{\theta}$	Angular velocity in Pitch

LIST OF ACRONYMS/ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
AGV	Automated Guided Vehicle
AMCL	Adaptive Monte Carlo Localization
AMR	Autonomous Mobile Robot
ART-SLAM	Accurate Real-Time Simultaneous Localization and Mapping
BFF	Body Fixed Frame
BiSeNet	Bilateral Segmentation Network
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CPU	Central Process Unit
DC	Direct Current
DOF	Degree Of Freedom
EKF	Extended Kalman Filter
ESC	Electronic Speed Controller
GCF	Global Coordinate Frame
GNSS	Global Navigation Satellite Systems
GPIO	General-Purpose Input/Output
GPS	Global Positioning System
HITL	Hardware In The Loop
HLC	High Level Controller
ICC	Instantaneous Center Of Curvature
ICR	Instantaneous Center Of Rotation
IF	Inertial Frame
IMU	Internal Measurement Unit
KF	Kalman Filter
LeGO-LOAM	Lightweight and Ground-Optimized Lidar Odometry and Mapping

LIDAR	Light Detection and Ranging
LI-PO	Lithium-Ion Polymer Battery
LLC	Low Level Controller
LOAM	Lidar Odometry and Mapping
LQR	Linear Quadratic Regulator
MATLAB	Matrix Laboratory
MAVROS	Multi Aerial Vehicles in Robot Operating System
MCL	Monte Carlo Localization
MLC	Mid Level Controller
MPC	Model Predictive Control
NMEA	National Marine Electronics Association
OHDA	Opposition based High Dimensional optimization Algorithm
PF	Particle Filter
PID	Proportional Integral Derivative
PLA	Polylactic Acid
QR	Quick Response
RC	Remote Control
RCF	Robot Coordinate Frame
RGB-D	Red Green Blue - Depth
RMSE	Root Mean Square Error
ROS	Robot Operating System
RTK	Real-Time Kinematic
Rviz	Robot Operating System Visualization
SITL	Software In The Loop
SLAM	Simultaneous Localization and Mapping
SM	Site Manager
SPI	Serial Peripheral Interface
TF	Transformation
UART	Universal Asynchronous Receiver-Transmitter
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle

UKF	Unscented Kalman Filter
UMR	Unmanned Mobile Robot
USB	Universal Serial Bus
USV	Unmanned Surface Vehicle
UTOPIA	Automated Open Precision Farming Platform
vSLAM	Visual Simultaneous Localization and Mapping

1. INTRODUCTION

In today's rapidly evolving technology landscape, mobile robots have become vital tools in various fields, including both indoor and outdoor environments. Indoor mobile robots excel in warehouse logistics and healthcare assistance, while outdoor mobile robots are particularly useful in agriculture, military, construction, and shipping.

In agriculture, outdoor mobile robots play a pivotal role in crop monitoring, pest control, and precision farming, leading to significant improvements in efficiency and sustainability. Furthermore, the combination of different types of mobile robots, specifically Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs), broadens their capabilities and applications. These heterogeneous mobile robots enable collaboration in complex environments, utilizing the unique advantages of each platform. UAVs offer aerial reconnaissance and data collection, while UGVs provide ground-based mobility and manipulation.

This thesis explores the cooperative localization and control capabilities of heterogeneous mobile robots, with a particular emphasis on the integration of Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs) within outdoor and agricultural environments. Figure 1.1 depicts the quadcopter system as UAV and Figure 1.2 depicts the differential drive ground vehicle system as UGV, both of which have been purposefully developed for this thesis. The study aims to demonstrate the synergy between these two platforms, exploring how their collaboration can enhance operational efficiency and effectiveness in complex, real-world scenarios. By leveraging the unique strengths of UAVs in aerial observation and UGVs in ground-based tasks, this research investigates how their coordinated efforts can optimize mission outcomes, particularly in precision agriculture and environmental monitoring. The findings from this study are expected to contribute to the development of advanced robotic systems that can autonomously perform a wide range of tasks, improving productivity and reducing the need for human intervention in challenging environments.



Figure 1.1. The Quadcopter system (UAV) developed for this thesis.



Figure 1.2. The differential drive ground vehicle system (UGV) developed for this thesis.

1.1. Motivation and Background

This thesis is motivated by the need to tackle challenges faced in outdoor mobile robot applications, particularly in the agricultural sector. The integration of various mobile robots to help improve agricultural practices is purposed. By combining different robotic platforms such as UAVs and UGVs, these diverse systems can support each other, enhancing versatility, adaptability, and efficiency in a range of agricultural tasks. Therefore, the primary focus of this thesis is to explore how these diverse mobile robots can work together to advance outdoor mobile robot applications, especially in agriculture, with potential benefits that extend beyond this sector.

The decision to select a quadcopter for the heterogeneous mobile robot system in this thesis is based on its unique capabilities and suitability for outdoor and agricultural applications, especially when compared to other multi-copters. Although other multi-copter configurations can offer similar aerial views, quadcopters typically balance stability, maneuverability, and payload capacity. This balance makes them ideal for various tasks.

Furthermore, quadcopters are usually more energy-efficient and mechanically simpler than configurations with more rotors, improving reliability and ease of maintenance in field operations. They provide an unrivaled aerial perspective, facilitating high-resolution data collection and reconnaissance from vantage points unreachable for ground-based vehicles.

Their versatility and agility allow for efficient navigation in complex environments, and their quick deployment capability makes them perfect for time-sensitive missions. Quadcopters operate nonintrusively, reducing disturbances to crops or natural habitats during tasks like crop monitoring or wildlife surveillance. With redundant propulsion systems, they improve safety and reliability, guaranteeing robust performance under challenging conditions.

The choice of a differential drive ground vehicle as the second robot in the heterogeneous mobile robot system is due to its specific advantages and the complementary role it plays to the quadcopter. Compared to other ground-based platforms like wheeled or tracked vehicles, this configuration offers superior maneuverability and adaptability to various outdoor and agricultural environments. Its ability to handle rough terrain and tight spaces makes it ideal for tasks such as soil sampling, crop monitoring, and navigating narrow rows in agricultural fields. Additionally, its simple design and mechanical components increase reliability and ease of maintenance, which are vital for ongoing operation in remote or challenging environments. By incorporating a differential drive ground vehicle with the quadcopter, the heterogeneous mobile robot system becomes more versatile, resilient, and capable of efficiently performing a wide range of tasks in outdoor and agricultural areas.

In conclusion, combining a quadcopter and a differential drive ground vehicle into a heterogeneous mobile robot system enhances its functionality. This allows for thorough data collection and accurate control in outdoor and agricultural settings. The quadcopter's aerial viewpoint allows for high-resolution data collection and scouting, while the ground vehicle's agility and durability allow for efficient navigation across diverse landscapes. These robots, working together, increase the system's ability to perform various tasks with versatility and accuracy, furthering the potential for innovative solutions in outdoor and agricultural applications. A brief comparison of UAVs and UGVs is given in Table 1.1.

Table 1.1. Advantages and disadvantages of UAVs and UGVs

Aspect	**UAV**	**UGV**
Advantages	Aerial perspective for reconnaissance and surveillance.	Provides a stable platform for carrying heavy payloads.
	Rapid deployment and mobility over obstacles and rough terrain.	Offers greater endurance and longer operating times compared to UAVs.
	Ability to cover large areas quickly.	Has a lower risk of collisions and damage in cluttered environments.
	No ground contact, which reduces wear and tear.	Can operate in GPS-denied or indoor environments.
Disadvantages	Limited payload capacity compared to UGVs.	Limited mobility in complex or vertical environments.
	Restricted flight time due to battery limitations.	Slower speed compared to UAVs, especially over long distances.
	Vulnerability to adverse weather conditions, such as wind and rain.	Susceptibility to terrain constraints and obstacles.
	Requirement for airspace regulation compliance.	Limited line-of-sight perception compared to UAVs.

In mobile robotics, the control systems usually consist of two levels: the High-Level Controller (HLC) and the Low-Level Controller (LLC) as shown in Figure 1.3. In some instances, there may also be a Mid-Level Controller (MLC). MLCs are responsible for bridging the gap between high-level planning and low-level actuation in robotic systems. They typically handle tasks ensuring that higher-level commands are effectively implemented by the robot's hardware.

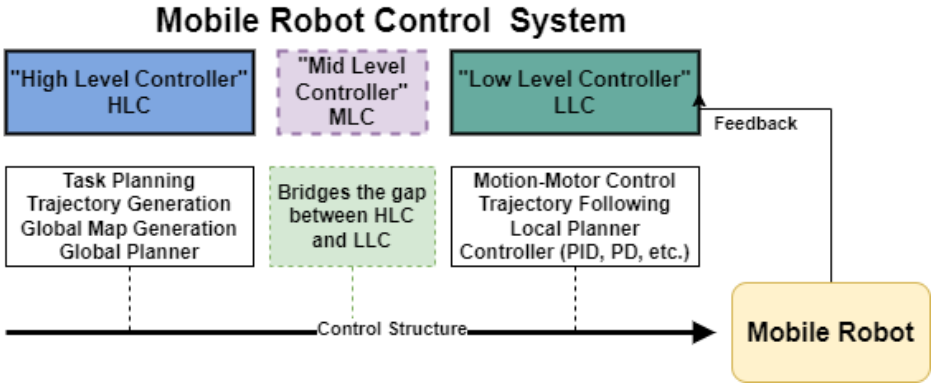


Figure 1.3. Illustration of control structure for a mobile robot.

The HLC, also known as the mission or task planner, serves as the system's cognitive hub. It interprets and processes higher-order commands from human operators or the overall system, formulating a series of tasks for the robot to accomplish its goals. It uses decision-making algorithms, path-planning techniques, and mission optimization strategies to define the robot's strategy and actions for maximum efficiency and goal attainment. These tasks can vary, including navigation to specific waypoints, exploration of unknown environments, or interaction with objects in the robot's surroundings.

The LLC, also known as the motion or motor controller, focuses on executing specific motor commands to manipulate the robot's actuators effectively. It acts as the interface between the high-level directives and the robot's physical hardware, translating commands into precise control signals for the robot's motors, servos, or other actuators. Its main goal is to regulate the robot's speed, orientation, and motion in real-time, ensuring accurate and agile maneuvers. The low-level controller often uses feedback control mechanisms such as PID (Proportional-Integral-Derivative) control, which continuously adjusts motor outputs based on sensor feedback to maintain desired performance levels despite any disturbances in the robot's operating environment.

In conclusion, the hierarchical structure of a mobile robot's control system, including both high-level and low-level controllers, allows for effective navigation, task execution, and overall system performance. The high-level controller handles the strategic decision-making and mission planning, while the low-level controller executes precise

motor commands, enabling the robot to turn abstract directives into precise physical actions. This structured approach is the foundation for autonomous operation and intelligent behavior in mobile robotic systems.

It is aimed to explore the complex dynamics of multi-robot systems comprising different platforms. The focus is on programming, coordination, and operational optimization through cooperative localization and control. The integration of these cooperative mechanisms between UGVs and UAVs is crucial in autonomous systems. It leads to significant efficiencies and improves situational awareness. This cooperation allows for shared sensor data and synchronized movements among unmanned vehicles. In addition, it opens new possibilities for managing complex tasks and missions. For instance, a cooperative team of UAVs and UGVs could execute tasks like search and rescue operations, environmental monitoring, or infrastructure inspection more effectively than individually. By thoroughly examining cooperative localization and control, this thesis seeks to highlight the potential of diverse multi-robot systems to transform various domains, offering superior capabilities and performance beyond single-platform solutions.

1.1.1. UTOPIA-Automated Open Precision Farming Platform

This thesis presents research that contributes to the UTOPIA (Automated Open Precision Farming Platform) project. This international initiative aims to advance precision agriculture through the development of an open framework. The project's goal is to establish a cloud-based repository for storing relevant precision agriculture data, accessible via an intuitive user interface. This framework enables the deployment of robotic platforms such as UAVs, Unmanned Surface Vehicles (USVs), and UGVs for missions specified by stakeholders like farmers, without additional labor or time investments. Figure 1.4 shows the UTOPIA's vineyard suitcase field where some tests are performed.



Figure 1.4. The UTOPIA's vineyard suitcase field.

The study's primary use case involves vineyard operations, where a quadcopter and a ground vehicle conduct cooperative localization and control missions. UTOPIA emphasizes user-friendliness and accessibility, enabling farmers to utilize advanced precision agriculture technologies without the need for extensive re-education or technical expertise.

By aligning robotic paths and data formats, the UTOPIA system supports multiple use cases across terrestrial, marine, and aerial environments, even under limited bandwidth conditions. UTOPIA aims to usher in a new era of efficiency and sustainability in precision agriculture by reducing development costs for agricultural equipment companies and enhancing autonomy in data-driven decision-making.

Figure 1.5 illustrates the hierarchical structure of the UTOPIA devices, showcasing how data moves through stages to inform real-world experimentation. Initially, various data streams are collected, representing a 0D stage where the data lacks context. Through processing and analysis, this data is transformed into meaningful information, transitioning to a 1D representation.

Next, insights from this processed data are used to generate 2D path planning and odometry data, which facilitate spatial navigation and trajectory planning. These

2D representations are then expanded into a 3D digital twin, offering a detailed visualization of the operational environment.

After iterative improvements within the digital twin framework, experimental tests are conducted in the real world. These tests validate and refine the proposed methodologies and algorithms. This hierarchical progression highlights the evolution of data from its raw form to actionable insights, culminating in real-world implementation and validation.

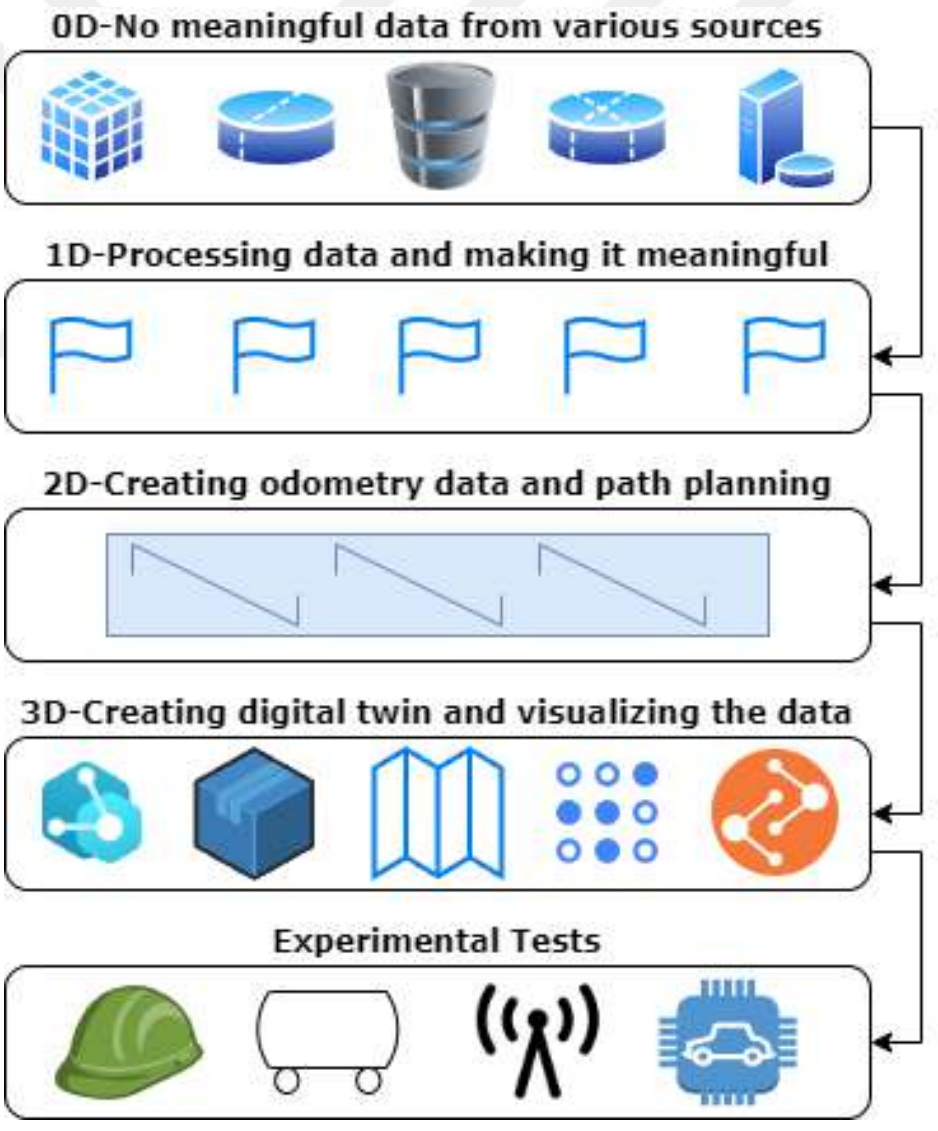


Figure 1.5. Illustration of the hierarchical structure of the UTOPIA devices.

Figure 1.6 illustrates the operational flow and control system for UTOPIA. In this project, both UAVs and UGVs are equipped with multiple sensors to gather environmental and agricultural data, which is crucial for farm monitoring and interventions. After data collection, it is labeled and sent to the Site Manager through a cloud service. This process enhances the speed and precision of agricultural monitoring.

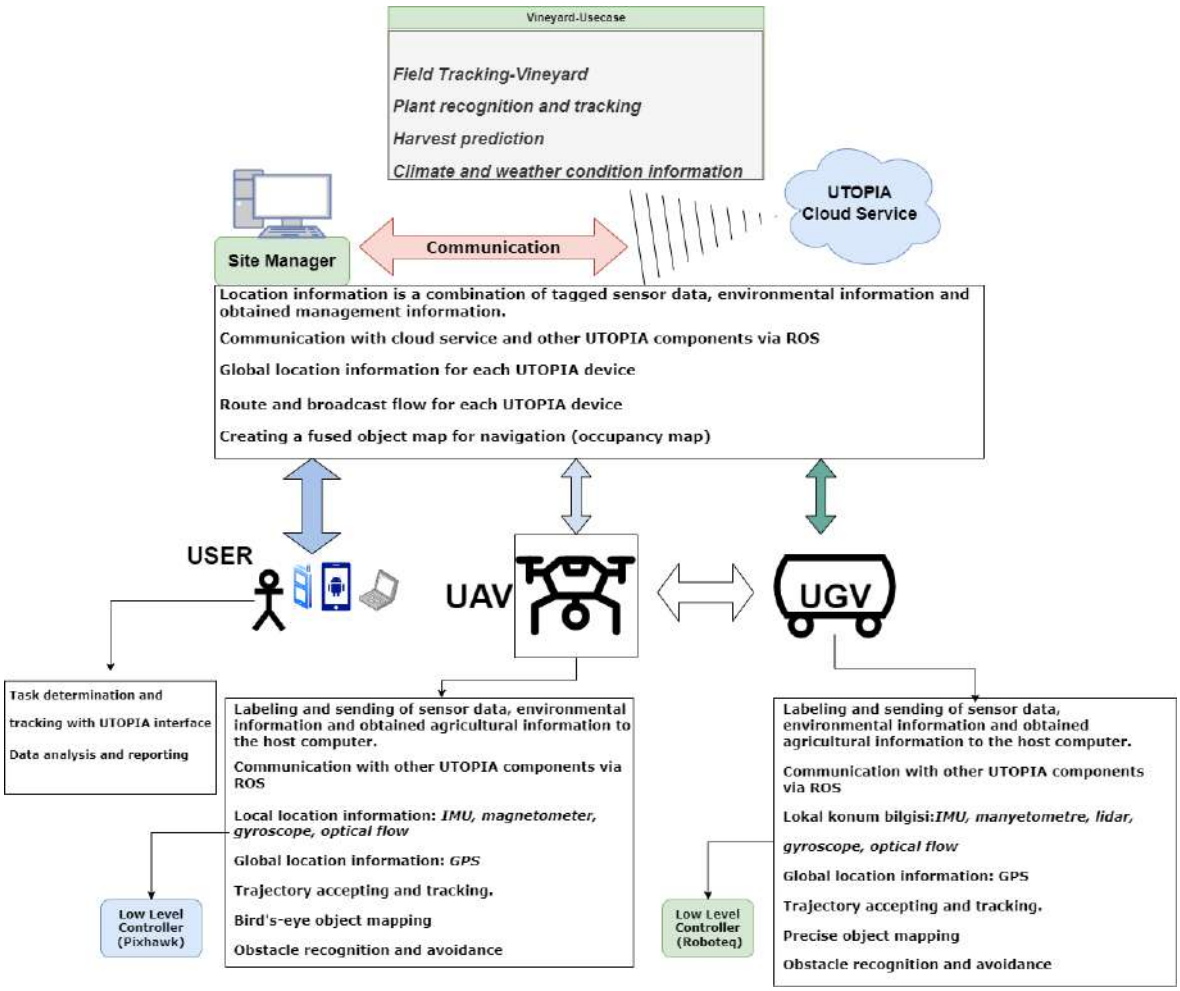


Figure 1.6. Illustration of operational flow and control system for UTOPIA project.

The operations of both UAV and UGV rely heavily on the Low-Level Controller. Each platform uses specialized controllers: a Pixhawk-flight controller for the UAV and a Roboteq-motor controller for the UGV. These controllers act as central processing units, executing commands and managing real-time adjustments during tasks or navigation. This centralized control ensures the precise operation of both UAV and UGV, enhancing their performance in various agricultural tasks.

The combination of user input and the technological capabilities of the UAV and UGV allows for a high level of automation in smart farming practices. This leads to more efficient and sustainable agricultural operations.

1.2. Problem Statement and Objectives

The integration of quadcopter as UAV and differential drive ground vehicle as UGV offers a promising solution for outdoor applications, especially in agriculture, where each platform has unique advantages. UAVs, particularly quadcopters, are known for their maneuverability and versatility, making them valuable for tasks that need aerial perspectives. However, to fully utilize quadcopters, robust control systems and precise localization capabilities are necessary.

Similarly, while UGVs perform well in ground-based tasks, they need effective localization to accurately navigate outdoor environments. To maximize the benefits of both platforms, each robot requires a robust localization and control system. This ensures that they can operate autonomously and effectively in their respective domains. Additionally, the creation of cooperative localization and control mechanisms within a global map framework is crucial for seamless collaboration between different types of mobile robots.

Addressing these needs is key to advancing the field of autonomous and unmanned robotic systems and enhancing their performance in outdoor applications. The objectives of the dissertation are outlined as follows:

- To design and build a fully functional quadcopter framework for the UAV and a fully functional differential drive ground vehicle for the UGV. This includes incorporating the necessary hardware elements.
- For the low-level controllers, the plan is to use PixHawk flight controller board for the UAV and Roboteq Direct Current (DC) motor driver for the UGV. This set includes the Pixhawk flight controller board, equipped with the PX4 autopi-

lot system. This system is renowned for its robustness and versatility, making it ideal for autonomous flight operations. Furthermore, the Roboteq DC motor driver, when used in combination with the Tracking-PID and MPC-Tracker packages, allows for a comparison between PID and Model Predictive Control (MPC) controllers and enables precise motor control. These choices are driven by their open-source nature, extensive community support, and compatibility with a variety of hardware, making them flexible and reliable options. All algorithms will be developed using the C, C++ and Python programming languages.

- To implement Sensor Fusion using an Extended Kalman Filter (EKF) for each robot. This widely used technique will integrate data from various onboard sensors. By using EKF, we aim to improve each robot's state estimation accuracy, strengthening the basis for control actions and boosting the system's resilience. This enhancement aims to increase the reliability and efficiency of the robots' operations, thereby improving the robustness of the entire system.
- To develop a reliable localization and control system for each robot to ensure precise navigation and task completion in outdoor settings.
- To set up individual Robot Operating System (ROS) environments and the high level controllerd for each robot. This will facilitate seamless integration, communication, and control within the diverse robotic system.
- To create digital twins of the robots and the vineyard environment to facilitate the refinement and testing of cooperative localization and control algorithms, aiming to enhance the robots' performance in real-world agricultural settings.
- To create a global map framework that integrates cooperative localization techniques, facilitating seamless collaboration among different types of mobile robots.
- To design and execute cooperative missions to assess and improve cooperative localization and control abilities. Use Matrix Laboratory (MATLAB) to create an algorithm for generating mission paths that coordinate robot actions.
- To use the Gazebo simulation platform to simulate individual robot control capabilities and cooperative missions. Employ Robot Operating System Visualization (RViz) to visualize simulation results for real-time monitoring and analysis. Implement ROS nodes to log and plot collected data, providing a thorough eval-

uation and visualization of cooperative mission performance.

- To design and construct a proof-of-concept UGV to serve as a test-bed for evaluating the efficacy of developed algorithms in real-world scenarios, starting with indoor environments. Refine and adapt the algorithms based on the findings from these tests. Incorporate insights from real-world experimentation and implement these enhancements in the simulation environment. This ensures alignment between simulated and real-world performance.
- To implement the developed ROS environment and algorithms on both the UAV and UGV platforms. Conduct necessary calibrations and adjustments to ensure seamless integration and functionality. Start with improving indoor localization for the UGV, then move on to refining outdoor localization capabilities. For the UAV, begin with software in the loop (SITL) simulations to validate algorithm performance. Then, proceed to hardware-in-the-loop (HITL) mode for real-world testing and validation. Continually refine the algorithms based on insights from both simulation and real-world experimentation to ensure optimal performance in diverse operational environments.
- To carry out cooperative missions in real-world settings to thoroughly evaluate the effectiveness of cooperative localization and control mechanisms across different mobile robot platforms. During these missions, gather empirical data to assess algorithm performance under genuine operating conditions. Following this, analyze the data to gain insights into the effectiveness and robustness of the cooperative localization and control strategies. Convey the results through detailed reports and visualizations, offering concrete evidence of how the algorithms enhance overall system performance in outdoor and agricultural environments.

1.3. Outline of the Thesis

Below is a summary of the thesis structure.

- Chapter 2 offers a comprehensive review of the UTOPIA system. It covers hardware and electronics design for each component, the navigation system (including

trajectory generation), control theory, and sensor fusion techniques.

- Chapter 3 outlines the setup for simulations and presents the results. It offers a detailed analysis of each robot's performance and behavior in various simulated scenarios.
- Chapter 4 outlines the calibration and testing process, presents the test results, and offers an in-depth discussion of the results from practical experiments on cooperative missions. These missions were designed to evaluate robot performance under sensor fusion. This chapter emphasizes the improvement of the robots' localization capabilities, assessing the effectiveness of the developed algorithms in enhancing localization accuracy and robustness. Through detailed analysis and discussion, Chapter 4 provides crucial insights into the practical implications of sensor fusion and cooperative missions. It also identifies the strengths and limitations of the developed system in real-world scenarios.
- Chapter 5 concludes the thesis. It summarizes the findings and implications, while also pointing out potential directions for future research and development.

1.4. Literature Survey

Localization is vital for effective navigation in mobile robots. It enables a robot to comprehend its environment and navigate reliably, which is essential for tasks from straightforward navigation to intricate interactions in dynamic environments. Consequently, enhancing localization capabilities is a central objective in robotics research.

In a comprehensive literature review on mobile robotics, discussions on control techniques are vital due to their interaction with localization mechanisms. Effective control strategies ensure that a robot not only recognizes its location but can also act on this knowledge to perform tasks accurately. The literature review is instrumental in understanding the current state of research in a particular field. This review brings together and scrutinizes these studies to set the groundwork for future research aimed at addressing the challenges and opportunities in cooperative localization and control for heterogeneous mobile robots. This thesis includes an extensive literature review

on sensor fusion applications in multi-robot systems and cooperative localization and control within these systems. This thorough review highlights efforts to enhance the localization capabilities of multi-robot systems, explains methodologies used in collaborative missions, and delves into the emerging field of UAV-assisted UGV cooperative localization studies for heterogeneous mobile robots.

1.4.1. Methods of Localization

1.4.1.1. Sensor-Based Approaches. These methods employ sensors such as LiDAR (Light Detection and Ranging), GPS (Global Positioning System), encoders, IMUs (Internal Measurement Unit), ultrasonic, and visual cameras. Each sensor type has its own advantages and disadvantages. For example, while GPS is excellent for outdoor localization, it's less effective indoors. On the other hand, LiDAR provides precise distance measurements, but its integration can be costly and complex. Various studies touch on this topic in existing literature. For instance, a research group from Seoul National University of Science and Technology proposed a study to enhance indoor localization using a camera and encoders of an AGV [1]. They corrected the robot's position by detecting QR (Quick Response) codes, resulting in decreased error. However, they found that while QR codes positively impact localization, sensor-based localization alone is insufficient. To achieve more accurate localization, they applied EKF. A research group from Sakarya University's Department of Mechatronics Engineering presents another study [2]. They combined wheel odometry and visual odometry from a camera to enhance mobile robot localization. They also applied EKF to improve localization, as it was found to be insufficient without filtering.

1.4.1.2. Simultaneous Localization and Mapping (SLAM). SLAM techniques allow a robot to build a map of an unknown environment while simultaneously tracking its location within that map. Variants of SLAM, such as visual SLAM (vSLAM) and LiDAR SLAM, utilize different sensors for these tasks. A recent study from a research group at the Georgia Institute of Technology explored various SLAM algorithms, including vSLAM, LOAM (Lidar Odometry and Mapping) derivatives, and ART-SLAM

(Accurate Real-Time SLAM) for precise localization and mapping for mobile robots in construction environments [3]. These environments, like agricultural ones, present difficulties in achieving accurate localization. A significant challenge is the need for precise mapping and localization in environments that change daily. The researchers discovered that using various algorithms can enhance localization and mapping. However, SLAM alone isn't sufficient in challenging environments. Therefore, GPS or IMU fused SLAM algorithms like ART-SLAM and LeGO-LOAM (Lightweight and Ground-Optimizated Lidar Odometry and Mapping) offer superior performance and provide accurate localization. A recent study was conducted by a research group from OWL University [4]. The team examined and compared renowned SLAM (Simultaneous Localization and Mapping) approaches such as vSLAM, LOAM derivatives, LiDAR-based SLAM techniques, and RGB-D SLAM for AGVs (Automated Guided Vehicles) and AMRs (Autonomous Mobile Robots). They concluded that SLAM algorithms can effectively integrate various sensors, like LiDAR, cameras, and inertial sensors, to provide more precise and reliable localization and mapping in complex situations. These sensors are becoming increasingly important in the field.

Another research group from China Agricultural University has presented a significant study on SLAM-based localization for underwater robots [5]. The underwater environment introduces challenges such as unstructured surroundings, noise, lack of GPS, and low light sources. The researchers discovered that due to these challenges, vSLAM-based approaches are not entirely reliable for accurate localization. As a result, the traditional visual SLAM, which relied on filtering methods, is gradually evolving towards optimization-based methods.

Adaptive Monte Carlo Localization (AMCL) is a widely used method for SLAM-based localization, which is effective in known environments. Unlike traditional localization techniques that depend solely on pre-built maps, AMCL uses a probabilistic approach to estimate the robot's position and orientation relative to the map. It uses a particle filter framework to adaptively update the robot's belief about its location, based on sensor measurements. This effectively accommodates uncertainties in both

the robot's motion and environmental features. Its adaptive quality allows AMCL to localize the robot robustly, even in dynamic or changing environments, leading to its popularity in real-world robotic applications. A research team from the University of Quebec recently presented a significant study [6]. It showcased the improved localization of a mobile robot using the AMCL method, combined with information theory and statistical approaches. The team demonstrated that the enhanced AMCL algorithm, applied to a mobile robot, improved the accuracy of the local planner.

1.4.1.3. Filter-Based Techniques. Filter-based methods for robot localization are techniques used to estimate a robot's pose (its position and orientation) in its environment. These estimates are based on both sensor measurements and prior knowledge. Mathematical models known as filters are used to combine sensor data and gradually update the assumed location of the robot. The term "filter" refers to algorithms like the Kalman Filter (KF), which sequentially processes sensor measurements. This allows the estimation of a dynamic system's state, such as a robot's pose, while considering uncertainties and noise in the measurements.

As highlighted in references [1–5], approaches that do not use filtering tend to have lower performance for reliable localization, particularly in unstructured and outdoor environments. Sensor fusion is essential for robot localization as it compensates for individual sensor limitations like noise, drift, and restricted sensing range. By merging data from multiple sensors, filter-based localization methods can provide more robust and precise estimates of the robot's position, enabling effective navigation in diverse environments. A research group from The Institute for Research in Informatics conducted a significant study on estimating the location of an autonomous agricultural vehicle [7]. They achieved this by using sensor fusion with an EKF from RTK-GNSS (Real Time Kinetic-Global Navigation Satellite Systems) and an IMU. They demonstrated that sensor fusion using an EKF significantly impacts localization, particularly if there is no GPS loss. Another major study from the Munich Institute of Robotics and Machine Intelligence introduced a novel method to estimate the misalignment between SLAM and Earth-fixed coordinate systems [8]. This method entailed fusing data from an IMU,

GNSS/GPS, and both 2D and 3D lidar sensors. This approach notably improved the global localization accuracy of obstacles in an agricultural setting. A notable study on sensor fusion for localization in agricultural settings was conducted by the University of Southern Queensland and the University of Technology Baghdad [9]. They fused data from a camera and a LiDAR sensor using the EKF algorithm. This allowed for precise localization of a mobile robot in orchard mapping. Results demonstrated that the EKF algorithm performed well and maintained the robot's position close to the ground truth. Another research team from the Polytechnic University of Bari and the National Research Council of Italy has unveiled an important study [10]. The study showcases a strong method for estimating the orientation of a UGV in agricultural applications. The team created an algorithm to adjust the vehicle's orientation using multiple IMU sensors and dual GPS correction with an EKF. They achieved high accuracy and effectively demonstrated the positive impact of EKF on improving localization even in challenging environments like agriculture.

1.4.1.4. Deep Learning Approaches. Deep learning methods for localization use neural networks to estimate a robot's pose (position and orientation) in its environment based on sensor data or raw sensory input. These methods utilize the representation learning abilities of deep neural networks to extract intricate features from sensor data. They then map these features to the robot's pose, eliminating the need for explicit models or feature engineering. A significant study on deep learning methods for localization in agricultural settings was conducted by South China Agricultural University [11]. The research team investigated a real-time localization method and semantic map reconstruction for unstructured citrus orchards. They achieved this by integrating vSLAM with the BiSeNet (Bilateral Segmentation Network) algorithm. Their findings suggest that combining vSLAM with a deep learning algorithm can yield high accuracy and real-time performance in semantic map reconstruction. This provides valuable technical insights and support for mapping in agricultural scenarios and smart agriculture. A combined research group from Toosi University of Technology and the University of Calgary has investigated a new evolutionary Particle Filter (PF) based on the Opposition-based High Dimensional Optimization Algorithm (OHDA) [12]. They

concluded that using deep learning algorithms to solve SLAM problems is a promising approach, especially when vision information is available.

1.4.2. Cooperative Localization and Control

Cooperativeness in robotics refers to multiple robots, with or without human collaboration, working together in the same network towards a common goal. Cooperative localization, therefore, is the process of estimating the location of each member within this network. It often involves the use of various methods and techniques to enhance the accuracy and robustness of localization. For example, a research group from the University of Science and Technology Beijing conducted an influential study on multi-vehicle cooperative localization, in which they integrated GPS and inter-vehicle position data for each vehicle [13]. A research team from Seoul National University of Science and Technology conducted another significant study. They created mapping and cooperative localization for two AGVs using ROS and SLAM. For communication and control, they utilized a computer as a master [14].

Moreover, a reliable control system is essential to make sure each robot precisely follows its intended path. A research team from the University of Stuttgart recently published a study, demonstrating cooperative localization and control for multiple UGVs during object transportation [15]. The team utilized MPC for managing UGVs and combined sensor fusion with image processing for localization. A research team from the French military academy and Czech University of Defence conducted a notable study on UAV-assisted landing on a moving UGV [16]. They utilized sensor-based localization with an IMU sensor and PID control with a camera to accomplish the task. A research team from Seoul National University presented a study on creating 2D cooperative localization for three omnidirectional mobile robots [17]. They used an EKF-based cooperative localization algorithm for this task. One robot was equipped with GPS-corrected absolute position information while the others had relative position data. The team applied PID control to the robots for trajectory following. The study demonstrated that with relative pose measurements to another mobile robot,

each robot could follow the desired trajectory with a bounded error.

1.4.3. Contribution of This Thesis

Current research on cooperative localization and control applications for heterogeneous mobile robots is limited, suggesting room for further exploration. Most prior studies have concentrated on multi-robot systems, not specifically heterogeneous mobile robots. They've typically either focused on cooperative control or localization, predominantly in indoor settings. This thesis will address both cooperative localization and control for heterogeneous mobile robots, particularly in challenging outdoor environments such as agricultural fields.

2. UTOPIA SYSTEM

2.1. UTOPIA's Components-Hardware Design

This chapter explores the hardware design considerations for both the UAV and the UGV. It begins by explaining the kinematic principles that govern the motion of each robot, which gives insight into their operational capabilities. Next, it details the mechanical design and manufacturing processes used in the creation of the UAV and UGV, providing a thorough understanding of their physical structures. Finally, it examines the complex electronics integrated into each robot, clarifying the circuitry and components that allow for their autonomous operation within the diverse robotic system.

2.1.1. Design of the UAV

The UAV used in this thesis is a carefully designed and built X-configured quadcopter. Its design was thoroughly verified through static and flow analysis to ensure structural integrity and top performance. The UAV, equipped with a broad sensor suite, excels at precise localization using EKF fusion. The following sections will explain these design elements in detail and how they help achieve accurate localization.

2.1.1.1. Mechanical Design and Installation of the UAV. Mechanical design is essential for control. If a vehicle experiences mechanical failures, it becomes impossible to apply a reliable control algorithm. The mechanical design process involves analyzing the kinematics and dynamics of the UAV. After this analysis, the design is completed using Computer Aided Design (CAD) and Computer Aided Engineering (CAE) modules in SolidWorks. During the design process, certain key parameters must be considered [18, 19]. These certain keys are listed as follows:

- The UAV is an X-configured quadcopter.
- The total thrust must be at least 1.8 times greater than the vehicle's weight.
- The sizes of the motors and propellers must be compatible with the axle length of the quadcopter's frame.
- The motors must provide the desired thrust with the selected battery
- The Electronic Speed Controllers (ESCs) must provide the desired current for the motors
- The frame must be both rigid and symmetric.
- The motor connections and landing feet should be rigid and durable. They must also be capable of absorbing shocks during crashes or hard landings to safeguard the main body.
- The center of mass of the UAV must be at the center of the frame where the quadcopter's arms converge - the center of the X-shape.

In the frame design, aluminum tubes with a 16mm diameter and 1mm thickness, as well as carbon fiber tubes with an 18mm diameter and 1mm thickness are used. The aluminum tubes are tightly fitted into the carbon fiber tubes to form a composite that combines their mechanical properties. The aluminum tubes lend strength against torsion, while the carbon fiber tubes resist bending. These tubes serve as the drone's arms. These arms are then symmetrically mounted in an X-shape to laser-cut aluminum plates with a thickness of 2mm, creating a rigid frame. All electronics, components, and the battery are positioned at the center of the laser-cut aluminum plates. This arrangement is to maintain the center of mass at the origin of the quadcopter's frame. The design and installation process of the UAV is depicted in the following figures: Figure 2.1 shows the 2D technical drawing, Figure 2.2 displays the CAD model, and Figure 2.3 presents the installation collage.

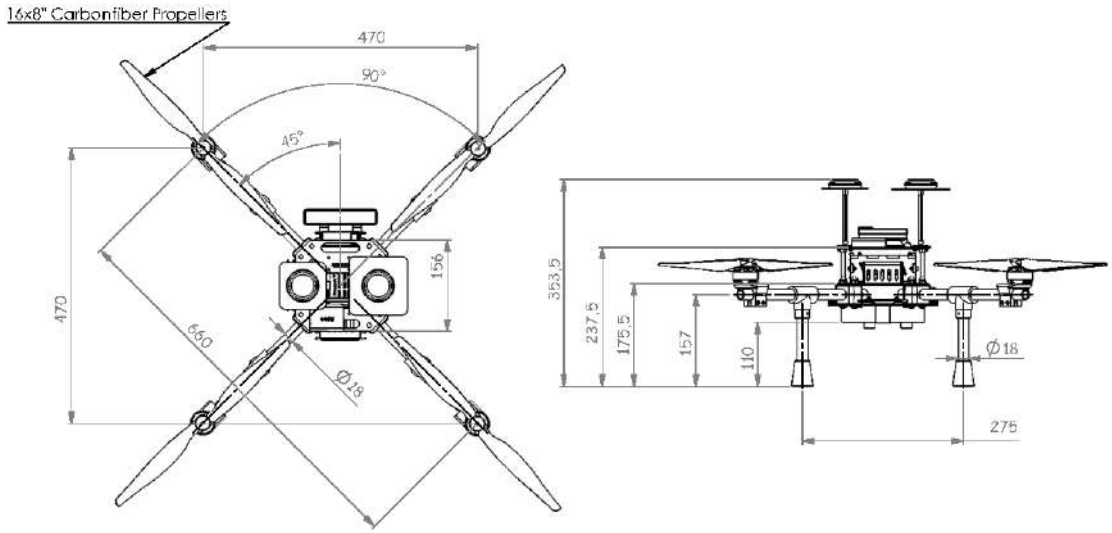


Figure 2.1. A 2D drawing of the UAV, highlighting its important dimensions in mm.



Figure 2.2. The CAD model of the UAV.



Figure 2.3. Behind the Scenes: Capturing Moments from the UAV Installation Process.

After installation, the total weight of the UAV, as calculated during CAD modeling, is 3.5 kg which is notated as m. As previously stated, the total thrust should be at least 1.8 times the vehicle's weight. Let F_i represents the thrust force of the i_{th} propeller, F represents the force vector, and $\sum_{i=1}^{4} F_i$ represents the total thrust forces of all propellers. Then the required thrust f , can be calculated as

$$T = 1.8 \times m \text{ (kg)}, \tag{2.1}$$

$$T = 1.8 \times 3.5 = 6.3 \text{ (kg)}, \tag{2.2}$$

$$\sum_{i=1}^{4} F_i = F_1 + F_2 + F_3 + F_4, \tag{2.3}$$

$$f = \frac{T}{4} = 1.575\,\text{kg} \text{ (note: } F_i = f). \tag{2.4}$$

where the required minimum thrust per propeller f becomes $\frac{T}{4} = 1.575$ kg. In the equations (2.4), (2.3), and (2.4), it is assumed that the motors are identical, and it can be considered that the thrust is evenly distributed among the propellers.

Motors(T-Motor U3 kV700) and propellers (16x5.5") are chosen based on the vehicle's mass and the required minimum thrust. Taking these factors into account, along with the selected propeller-motor combination, the axle length is set at 660 mm. Table 2.1 below provides the specifications for the selected motors and propellers.

Table 2.1. Selected motors and propellers

Component	**Specification**
Motor	T-Motor U3 kV700
Max Thrust	1.8 kg
kV	700
Battery	3-4S
Weight	128g
Max Current	25A
Propeller	Carbon Fiber 16x5.5 inches
Battery	4S Li-Po 12000 mAh

The design was validated using CAE tools in SolidWorks, including static, drop, and flow analyses, as seen in Figures 2.4 and 2.5 respectively. The static and drop analyses were performed on the 3D printed reinforced PLA (Polylactic Acid) plastic parts, specifically the landing feet mounting parts and motor holders. Conversely, the flow analysis was applied to the entire vehicle. These parts were purposefully designed for 3D printing to absorb shocks during crashes or drops, thereby protecting the main body and components. Upon impact, the thicker zones of these parts break, effectively shielding the main body. Given their PLA composition, these parts can be quickly replaced using 3D printing technology, facilitating prompt replacement if damaged.

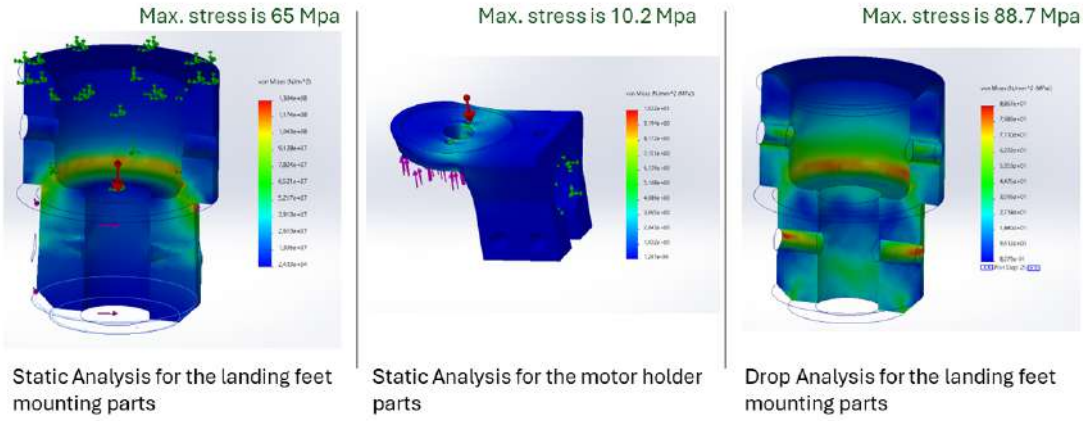


Figure 2.4. The results of the static and drop analyses for the 3D printed parts of the UAV.

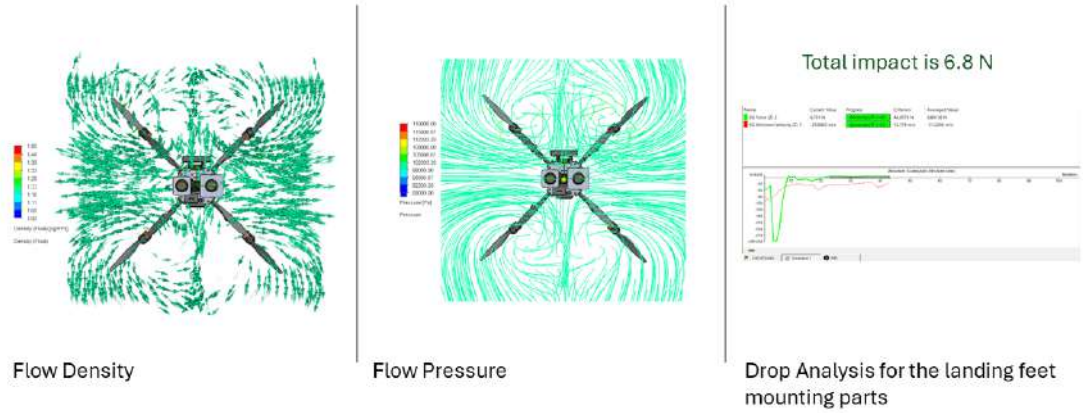


Figure 2.5. The results of the flow analysis for the UAV.

2.1.1.2. Electrical Design and Electronics of the UAV. The electronic design of the UAV includes a variety of essential components, such as sensors for navigation, localization, and environmental sensing. The choice and integration of these components

greatly influence the UAV's overall functionality and performance. After the mechanical design is complete and the motors and battery are selected, the Electronic Speed Controller (ESC) and power distribution module are chosen. Additionally, a sensor set for navigation is selected. The power distribution scheme is then designed as shown in Figure 2.6. The components are listed in Table 2.2 below.

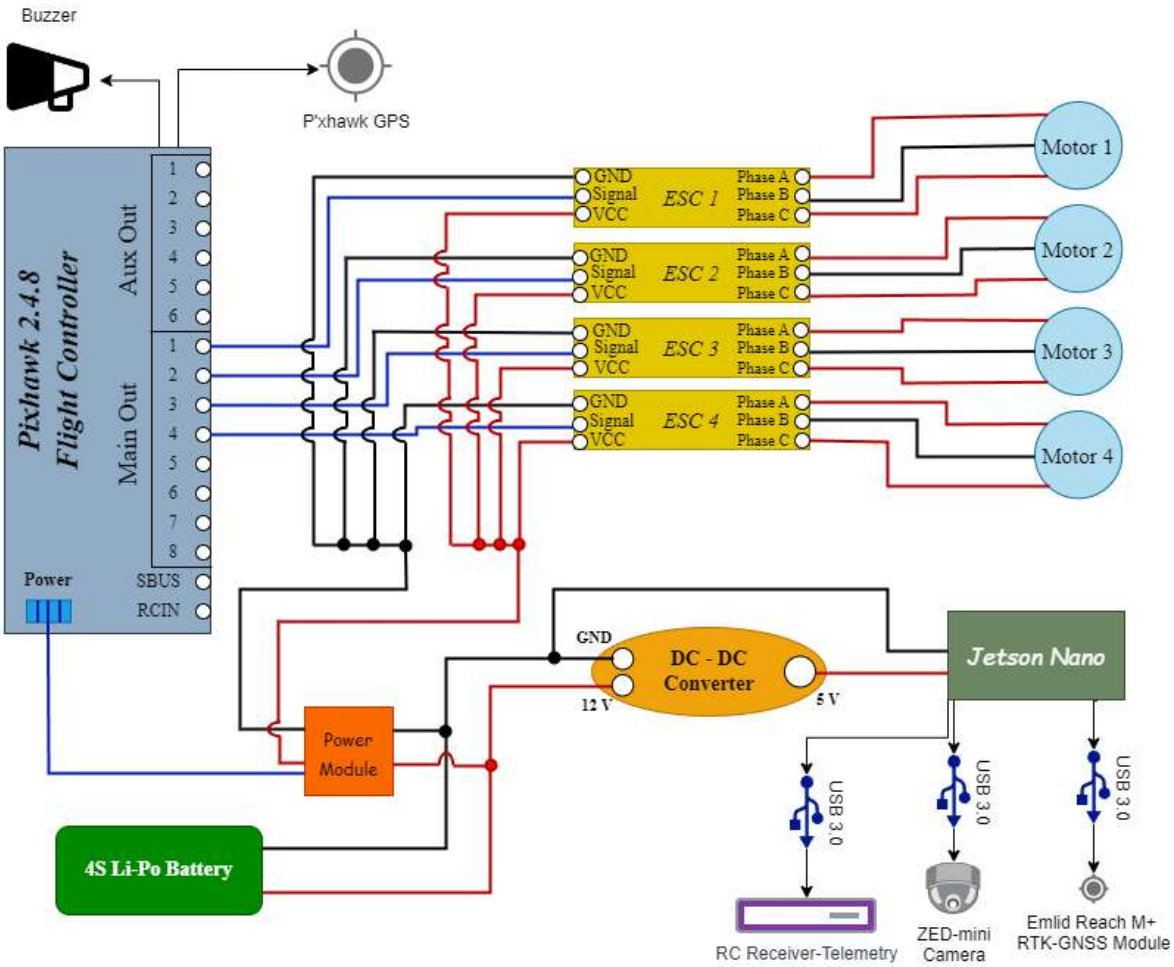


Figure 2.6. The electronic schematic of the UAV.

Table 2.2. Components of the UAV

No.	Component	Quantity
1	4S Li-Po Battery	1
2	Adjustable 5V DC-DC Converter	1
3	Buzzer	1
4	Brushless DC Motor (T-Motor U3)	4
5	Emlid Reach M+ RTK-GNSS Module	1
6	ESC 30A	4
7	Nvidia Jetson Nano Controller Board	1
8	Pixhawk 2.4.8 Flight Controller Board	1
9	Radio Controller Receiver and Transmitter	1
10	RadioLink Telemetry	1
11	ZED-mini Stereo Camera	1

The main electronics of the UAV are described as follows:

- 4S Li-Po battery is a reliable, high-power energy source suitable for extended periods of operation. The "4S" denotes its configuration of four individual lithium polymer cells connected in series, resulting in a nominal voltage of around 14.8 volts. This type of battery balances power output, energy density, and weight, making it appropriate for various robotics applications. After selecting the motors, a 4S battery is chosen due to the power requirements of the motors. T-motor U3 motors provide 1.8kg of thrust at 100 % throttle with the selected propeller size and a 14.8 V battery (4S). At 85 % throttle, the same setup provides 1.6kg of thrust [20].
- Adjustable 5V DC-DC converter is used. The battery provides 14.8V DC, suitable for the ESCs and motors. However, some components like the Pixhawk flight controller board and the Nvidia Jetson Nano Controller Board need 5V DC. Therefore, the battery's voltage must be reduced to 5V. This is achieved using a 150W DC-DC converter. The converter accepts an input voltage range of 7-40V

and offers an adjustable output of 1.25-35V. It can supply up to 30A current at 5V DC, accommodating multiple components such as the Jetson Nano controller board that requires a minimum of 4A, amongst others. A case is designed and 3D printed to safeguard and insulate the DC-DC converter.

- Buzzer is connected to the Pixhawk flight controller board to provide sound notifications, which help to understand the status of the UAV. It also delivers information during the UAV's sensor calibration.
- Emlid Reach M+ RTK-GNSS Module is a positioning device designed to observe and correct common errors in existing satellite navigation systems. It includes a 9DOF (Degree Of Freedom) IMU with an update rate that varies from 5 Hz to 14 Hz. Capable of measuring location with up to 14 mm precision while in motion, it is significantly more accurate than standard GPS devices. The module can function as a rover, self-correcting by receiving updates from the base module. It has been chosen for integration with EKF to enhance localization accuracy. GPS data can be monitored on EmlidFlow software. On EmlidFlow, the GPS data has three statuses: Single-No Base module, Float-corrections from the base module exist but are not stable, and Fixed-corrections from the base module are stable.
- ESC 30A enables flight control boards to manage the speed of the motors. The flight controller board adjusts the motor's speed by sending signals to the ESC to either increase or decrease the voltage. The ESC's current capacity is set at 30A to meet the motors' requirements.
- Nvidia Jetson Nano Controller Board is a digital controller as a small computer that can be used for both aerial and land vehicles. It includes a 4-core CPU (Central Process Unit) and a 128-core GPU. It also has numerous pins and USB (Universal Serial Bus) connection points, supporting various communication protocols such as GPIO (General-Purpose Input/Output), I2C, I2S, SPI (Serial Peripheral Interface), and UART (Universal Asynchronous Receiver-Transmitter). The Jetson Nano is chosen for the UAV due to its compact size, high computational capacity, and ability to connect multiple devices simultaneously via USB ports and GPIO pins.

- Pixhawk 2.4.8 flight controller board is a robust open-source autopilot system. It's mainly used for autonomous UAVs but can adapt to other robotic systems. Renowned for its reliability, Pixhawk provides sensor integration, real-time data processing, and accurate control algorithms. In this thesis, it is equipped with a GPS module and a RadioLink RC (Remote Control) receiver for the UAV. It also incorporates internal sensors like an IMU, magnetometer, gyroscope, and barometer.
- RC Transmitter-RadioLink AT9S is used for manual remote control of the UAV. This 12-channel transmitter features an internal display that monitors flight modes. It also enables flight mode control and switching via its switch buttons. Additionally, telemetry monitoring can be conducted on a PC or other devices.
- ZED-mini Stereo Camera is a compact, lightweight stereo camera designed for mixed reality and robotics applications. It comes with an integrated 6DOF IMU, depth-sensing, and accelerometer sensors. Using passive stereo imaging, the camera generates a depth map of the surroundings, allowing for real-time capture of 3D images and videos.

The main electronics are given in Figure 2.7, and the completed assembly of the UAV is shown in Figure 2.8.



Figure 2.7. The UAV's electronics.



Figure 2.8. The completed assembly of the UAV.

2.1.2. Design of the UGV

The UGV used in this thesis is built from an electric wheelchair chassis. This two-wheeled differential drive vehicle provides a robust base for autonomous navigation tasks. Equipped with a range of sensors, the UGV is optimized for accurate localization

using the EKF and efficient mapping operations.

2.1.2.1. Mechanical Design and Installation of the UGV. The UGV discussed in this thesis is constructed using an electric wheelchair's chassis. This method not only saves time and labor but also minimizes manufacturing and installation periods. The brackets and mounts for the sensors and other components are designed using SolidWorks and produced via 3D printing. The design and installation process of the UGV is depicted in the following figures: Figure 2.9 shows the 2D technical drawing, Figure 2.10 displays the CAD model, and Figure 2.11 presents the installation collage. As indicated in the 2D technical drawing, the UGV has a track width of 500mm— the distance between the centers of the drive wheels. This is referred to as the "S" distance in equation (3.7).

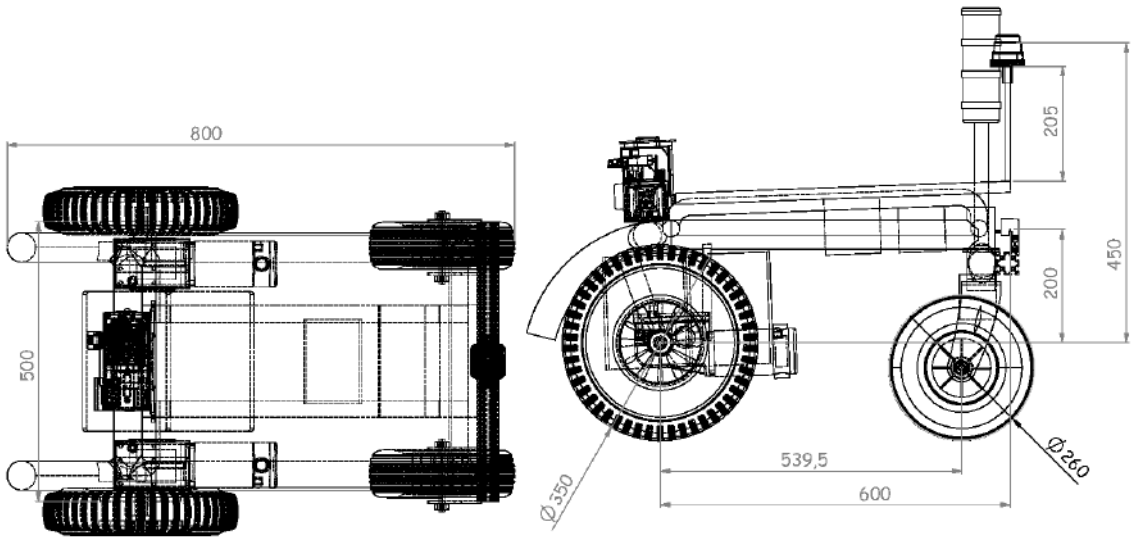


Figure 2.9. A 2D drawing of the UGV, highlighting its important dimensions in mm.

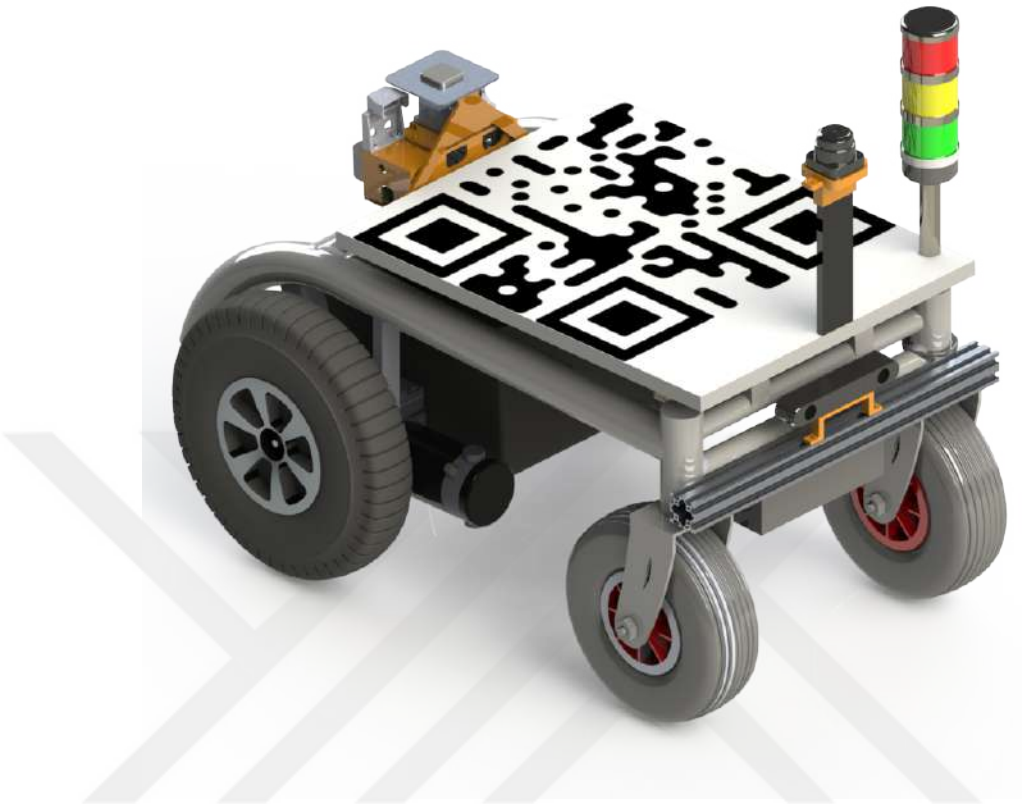


Figure 2.10. CAD model of the UGV.

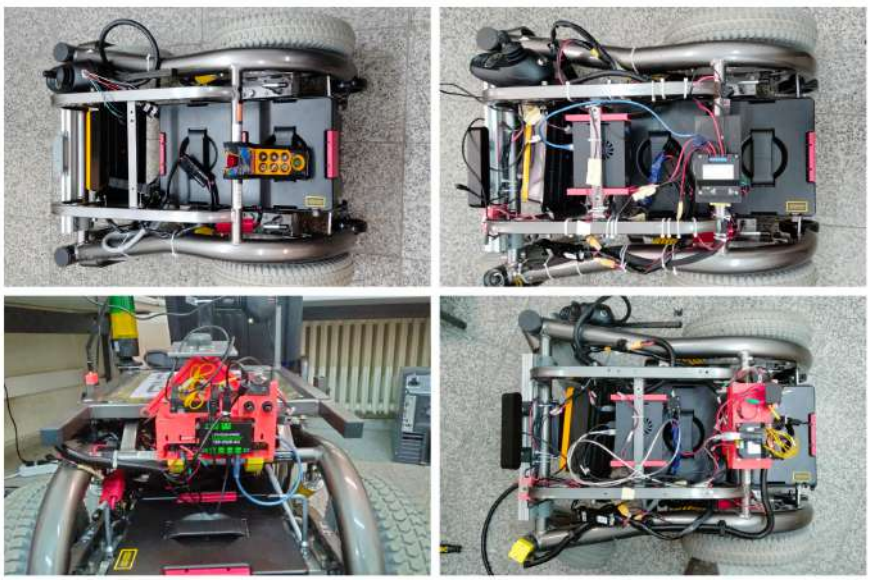


Figure 2.11. Behind the Scenes: Capturing Moments from the UGV Installation Process.

Additionally, a small UGV was constructed for indoor testing. Initially, it was used to test algorithms in laboratory environments using aluminum sigma profiles. Figure 2.12 below shows the small differential drive UGV and the completed assembly of the UGV.



Figure 2.12. The small UGV dor indoor tests(right one), and the completed assembly of the UGV (left one).

2.1.2.2. Electrical Design and Electronics of the UGV. The electronic design of the UGV encompasses essential components such as navigation, localization, and environmental sensors. The selection and integration of these components significantly affect the UGV's functionality and performance. Once the mechanical design and sensor set selection are finalized, brackets and mounting parts are designed and 3D printed to attach the components to the chassis. The power distribution and electronics scheme is designed as depicted in Figure 2.13. The components are outlined in Table 2.3.

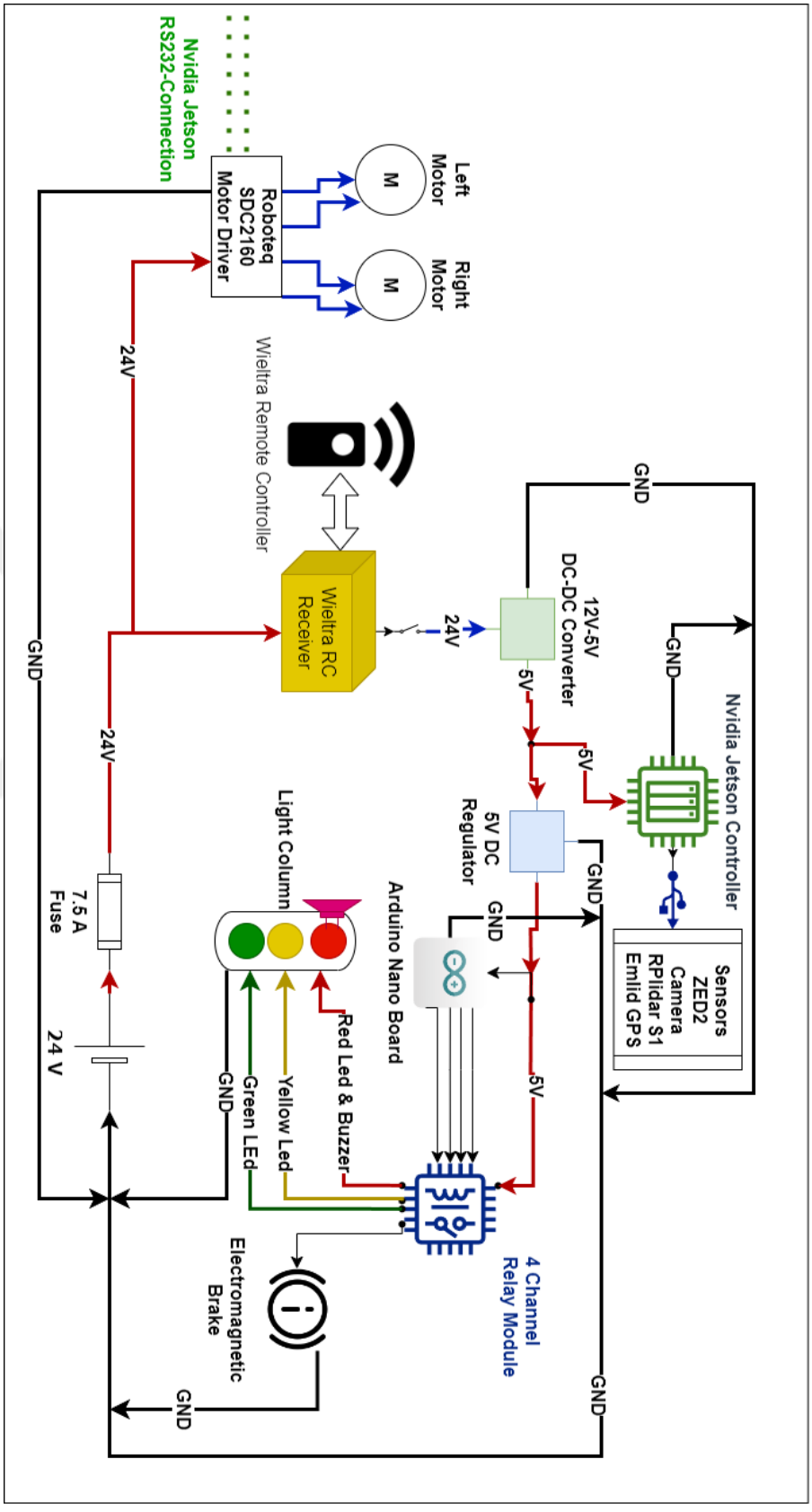


Figure 2.13. The electronic schematic of the UGV.

Table 2.3. Components of the UAV

No.	Component	Quantity
1	12V 40Ah Battery	2
2	4-Channel Relay Module	1
3	Adjustable 5V DC-DC Converter	1
4	Arduino Nano Board	1
5	Brushed DC Motor & 32:1 Gearbox	2
6	Emlid Reach M+ RTK-GNSS Module	1
7	Light Column	1
8	Nvidia Jetson Nano Controller Board	1
9	Roboteq SDC2160 Motor Driver	4
10	RPLidar S1	1
11	Wieltra RC Transmitter & Receiver	1
12	ZED2 Stereo Camera	1

The main components of the UGV are shown in Figure 2.14 below.



Figure 2.14. The main components of the UGV.

The main components of the UGV are described as follows:

- 12V 40Ah battery is used. The UGV operates on two serially-wired 12V 40Ah batteries, which collectively deliver 24V DC. To protect the electronics, the power output is linked to a 7.5A fuse. An RC transmitter is also connected to the power, allowing for an emergency stop if necessary.
- 4-Channel Relay Module is used to control the light column and the electromagnetic brake of the motors. These motors have an electromagnetic brake mounted on their shaft, which prevents the motor from turning when power is cut, enhancing safety. One channel controls the brake, while the other three channels respectively control the red LED and buzzer, yellow LED, and green LED of the light column.
- Adjustable 5V DC-DC converter is used to supply 5V DC power to electronics, such as the Jetson Nano Board and Arduino Board, which operate at this voltage. This converter steps down the power from 24V to 5V.
- Arduino Nano Board, based on the ATmega328P microcontroller, is a compact and versatile microcontroller board. Its smaller form factor makes it suitable for space-constrained projects. The Nano board includes a USB interface for easy programming and communication with computers, and digital and analog pins for interfacing with various sensors, actuators, and other electronic components. The UGV uses this board to communicate with the ROS master and control the electromagnetic brakes and light column over the relay module.
- Brushed DC motor & 30:1 gearbox drive motors are 24V brushed DC motors with 450W power. They feature an electromagnetic brake to keep the shaft fixed when the motor is not in use. The motors have a 32:1 reduction gearbox mounted to the shaft. The power is transferred to the wheels after the gearbox.
- Emlid Reach M+ RTK-GNSS Module is used. The UGV is equipped with an Emlid Reach M+ RTK-GNSS Module, similar to the UAV. This module operates in rover mode and receives corrections from the base module.
- Light Column-triple with an internal buzzer is used to indicate the UGV's status. It includes one buzzer and three LED lamps in red, yellow, and green. It's

connected to a 4-channel relay module, which controls it based on the status message received from the ROS master about the UGV's status. The green light signals that the UGV is performing the task. A blinking yellow light indicates that the UGV is launching, while a steady yellow light means the UGV is waiting for a command. A red light accompanied by the buzzer indicates that the UGV has failed.

- Roboteq SDC2160 motor driver is a 2-channel motor driver board. It can control brushed DC motors up to 60V and 20A per channel. This board is compatible with ROS and supports programming and monitoring over USB and RS232/RS485 protocols. It features integrated GPIO pins for adaptability across various applications and an internal trigger for maximum current and voltage. To ensure its protection, the Roboteq SDC2160 board is wired to a fusebox on the UGV.
- Nvidia Jetson Nano controller board is used. The UGV also has a Nvidia Jetson Nano Controller Board, similar to the one the UAV has.
- RPLidar S1 is a 360-degree Lidar sensor that's compatible with ROS. It features a ranging radius of up to 40m, 3cm precision, 0.391° angular precision, and is suitable for outdoor use with an enclosed cover. For this thesis, it has been mounted on the front of the UGV using a specially designed and 3D-printed bracket.
- Wieltra RC Transmitter & Receiver are integrated into the UGV for safety purposes. This device can remotely cut power in emergencies, acting as a remote-controlled relay. When the receiver gets a signal, it triggers the relay and cuts the power. To restore power, the RC transmitter must be restarted. Its replacement is depicted in Figure 2.15.
- ZED2 Stereo Camera, designed by Stereolabs, is ideal for applications such as robotics, augmented reality, and 3D mapping. This camera employs advanced depth-sensing technology to accurately perceive depth and understand spatial relationships in various settings, including both indoor and outdoor environments. Its compact design, lightweight, and ease of integration make the ZED 2 a popular choice among both developers and researchers. These features make it perfect

for creating intelligent systems and applications. The camera is equipped with a variety of internal sensors, including an IMU, gyroscope, magnetometer, accelerometer, barometer, and thermometers. With these tools, the ZED 2 can track position, and provide visual odometry and position data. For the purposes of this thesis, the ZED 2 is mounted to the UGV using a specially-designed and 3D-printed bracket.

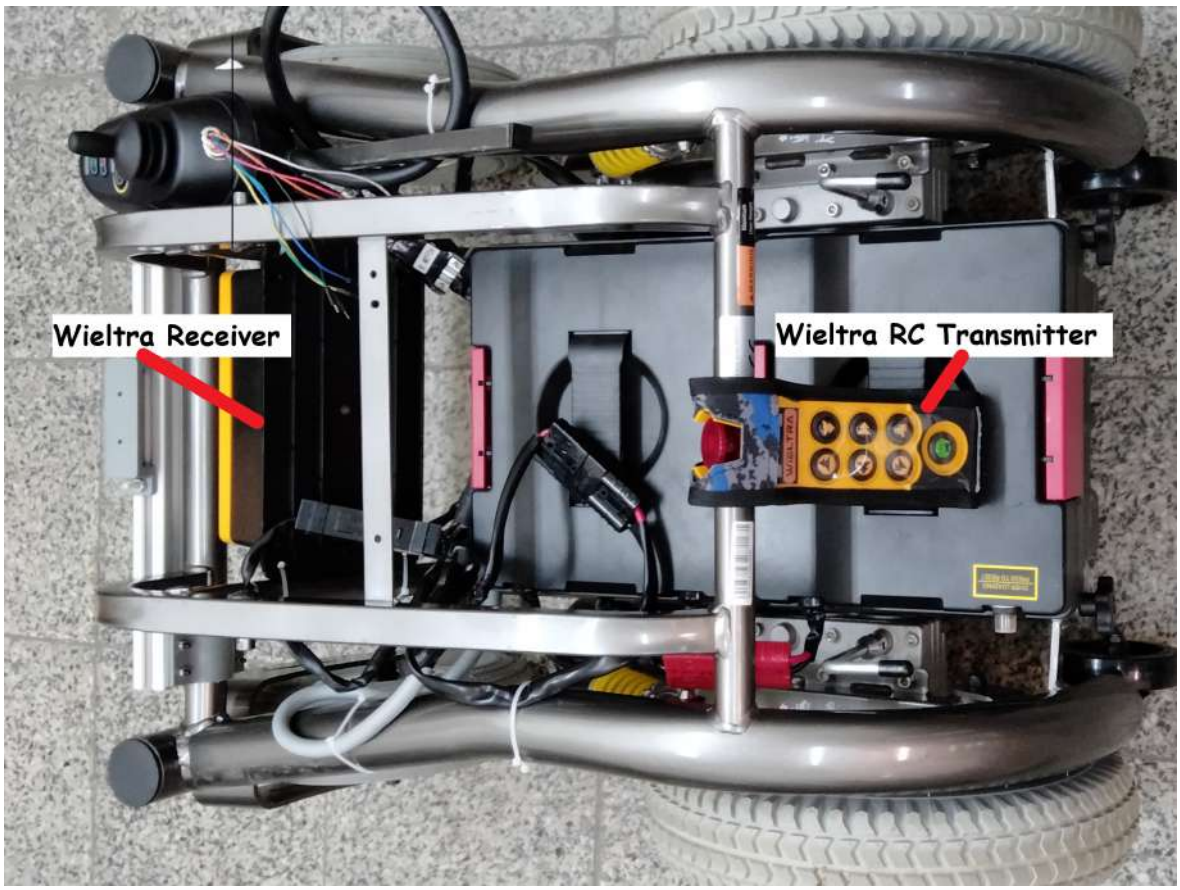


Figure 2.15. The Wieltra transmitter & receiver on the UGV.

2.2. UTOPIA's Navigation

2.2.1. Trajectory Generation

Trajectory generation, crucial for robotics motion planning, creates smooth, feasible paths considering position sequences, velocity, and acceleration while avoiding

obstacles. Its significance in this thesis is as a reference for localization comparison. The developed trajectory must be achievable by the tested robot to assess the localization improvement algorithms. The robot's localization is logged and compared, with decreasing discrepancies indicating improved localization.

2.2.1.1. Trajectory Generation Algorithm. Trajectory generation algorithms, from simple to advanced techniques, are essential for efficient and safe robot operation in scenarios like autonomous driving, robotic manipulation, and aerial navigation. They enhance the autonomy, accuracy, and performance of robotic systems by creating smooth, collision-free paths. Figure 2.16 below illustrates the trajectory generation algorithm, built using the point-to-point method in MATLAB, which requires inputs like trajectory v_T -velocity (m/s), d_e -Euclidean distance between points (m), T_s -time step (s), and r_T -radius of perpendicular corners (m). It divides the desired trajectory into smaller segments based on the provided Euclidean distance.

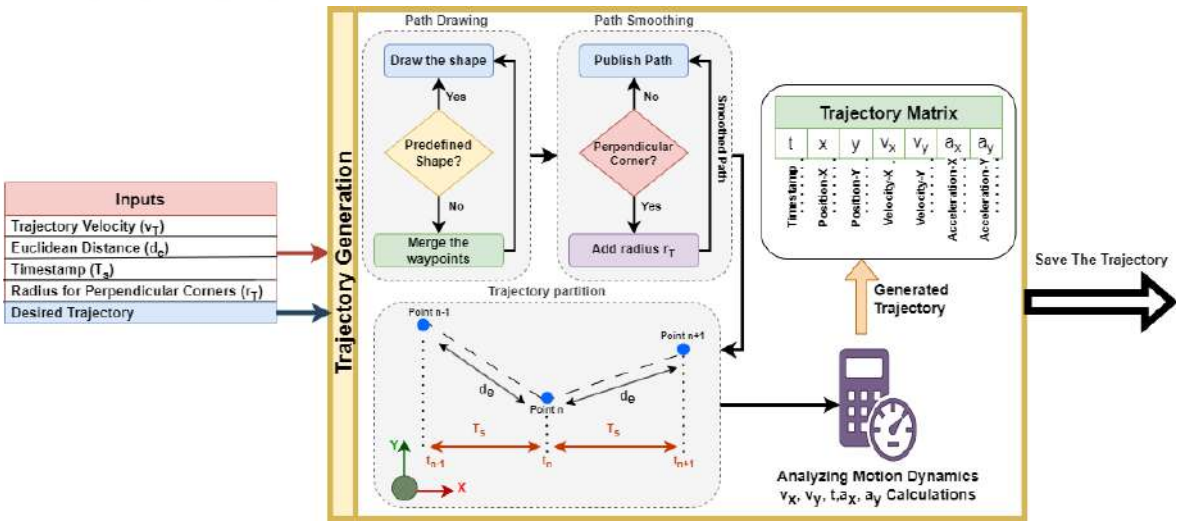


Figure 2.16. The diagram illustrating trajectory generation.

The trajectory can be a simple shape like a rectangle, circle, or a line, or a complex one from waypoints. If a predefined shape is chosen, the algorithm draws and divides it. For waypoints, it connects them, smoothens the path, and then divides it. The trajectory is represented by a 7-column matrix, with values for time, position, velocity, and acceleration in both x and y directions. Let $x(t)$ represents the robot's

x-coordinate at a given time t, with x_i and x_{i+1} indicating the starting and ending x-coordinates of the waypoints respectively. Similarly, y(t) represents the robot's y-coordinate at time t, with y_i and y_{i+1} indicating the starting and ending y-coordinates of the waypoints. t_i is the timestep for the starting waypoint, and t_{i+1} is the timestep for the ending waypoint. These equations facilitate the creation of a smooth trajectory between waypoints. Then the generation of the trajectory can be calculated as

$$x(t) = x_i + \frac{x_{i+1} - x_i}{t_{i+1} - t_i}, \tag{2.5}$$

$$y(t) = y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i}. \tag{2.6}$$

The algorithm discretizes data based on Euclidean distance using equations (2.5) and (2.6). Starting from an initial point, it moves along the trajectory until the sum of the Euclidean distances 'd_e' surpasses a set value. At that point, it documents the associated point and repeats the procedure. This method ensures that robots maintain an appropriate velocity while following a specified path. Derivatives of the positions are used to calculate $v(t)$ - the robot's velocity at a specific time, and $a(t)$ - the robot's acceleration at a specific time for x and y coordinates. The derivative of $x(t)$ gives v_x(t) - the x component of $v(t)$, while the derivative of $y(t)$ gives v_y(t) - the y component of $v(t)$ as

$$\begin{aligned} v_x(t) &= \frac{dx(t)}{dt},, \\ v_y(t) &= \frac{dy(t)}{dt}.. \end{aligned} \tag{2.7}$$

Similarly, the derivative of the velocities gives the acceleration $a(t)$ for the x and y axes can be calculated as

$$a_x(t) = \frac{dv_x(t)}{dt}, \tag{2.8}$$

$$a_y(t) = \frac{dv_y(t)}{dt}. \tag{2.9}$$

The trajectory generation algorithm is basically based on the equations (2.7), (2.7), (2.8) and (2.9).

2.2.1.2. Path Smoothing. Path smoothing is vital in trajectory generation, reducing high acceleration and deceleration rates from sharp corners, and enhancing robot stability and efficiency. It introduces curvature at corners, lessening sudden changes in

orientation and velocity. This provides smoother trajectories, reduces wear on mechanical parts, increases energy efficiency, and prevents loss of control.

This thesis addresses path smoothing at right-angle corners in robot trajectories, aiming for performance and safety optimization. By adjusting radius at corners, the planner can reduce acceleration peaks and provide smoother motion for the robot, improving navigation quality. A trajectory is an array of points with x and y coordinates. This thesis presents a three-function path-smoothing algorithm.The first function, "corner-mapping", identifies perpendicular corners in the trajectory. The second function, "getCT()", determines the corner's direction based on the neighboring points $p_{\text{in}}(i-1,:)$, $p_{\text{in}}(i,:)$, and $p_{\text{in}}(i+1,:)$. The third function, "generate-piece", applies a radius to corners and adjusts the trajectory. The corner mapping function takes an input matrix p_{in} of points and returns an output matrix p_{out} post-mapping. This is expressed as

$$p_{\text{out}}(i,3) = \begin{cases} 0 & \text{if } p_{\text{in}}(i,3) = 0, \\ \text{getCT}(p_{\text{in}}(i-1,:), p_{\text{in}}(i,:), p_{\text{in}}(i+1,:)) & \text{if } p_{\text{in}}(i,3) = 1. \end{cases} \tag{2.10}$$

where i is the index of the current point being processed. The result is defined by equation (2.10).

Let $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$, and $p_3 = (x_3, y_3)$ be three consecutive points in the trajectory. The differences in the x and y coordinates between the points are calculated as

$$\Delta x_1 = x_2 - x_1, \tag{2.11}$$

$$\Delta x_2 = x_3 - x_2, \tag{2.12}$$

$$\Delta y_1 = y_2 - y_1, \tag{2.13}$$

$$\Delta y_2 = y_3 - y_2. \tag{2.14}$$

The equations (2.11), (2.12), (2.13) and (2.14) are used to compute these differences. The corner type, denoted as *ctype*, is determined based on the signs of these differences.

This is described as

$$\text{ctype} = \begin{cases} 5 & \text{if } \Delta x_1 > 0 \text{ and } \Delta y_2 < 0 \text{ (from left to down)}, \\ 4 & \text{if } \Delta x_1 > 0 \text{ and } \Delta y_2 > 0 \text{ (from left to up)}, \\ 2 & \text{if } \Delta x_1 < 0 \text{ and } \Delta y_2 < 0 \text{ (from right to down)}, \\ 7 & \text{if } \Delta x_1 < 0 \text{ and } \Delta y_2 > 0 \text{ (from right to up)}, \\ 1 & \text{if } \Delta y_1 > 0 \text{ and } \Delta x_2 < 0 \text{ (from up to left)}, \\ 6 & \text{if } \Delta y_1 > 0 \text{ and } \Delta x_2 > 0 \text{ (from up to right)}, \\ 8 & \text{if } \Delta y_1 < 0 \text{ and } \Delta x_2 < 0 \text{ (from down to left)}, \\ 3 & \text{if } \Delta y_1 < 0 \text{ and } \Delta x_2 > 0 \text{ (from down to right)}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.15}$$

Equation (2.15) provides a clear classification based on the computed differences.

Upon identifying the corner type, a radius (r_T) is added in the corresponding direction for the generation of trajectory pieces. The equations for this process are provided below. The differences in the x and y coordinates are calculated as

$$\Delta x = p_2(1) - p_1(1), \tag{2.16}$$

$$\Delta y = p_2(2) - p_1(2). \tag{2.17}$$

For the generation of line pieces, the following conditions are applied as

$$\begin{aligned} \text{If } p_1(3) = 0: &\ \text{Generate straight line, segment,} \\ \text{Else if } p_2(3) = 0: &\ \text{Generate straight line, segment,} \\ \text{Else}: &\ \text{Generate line with circular arc at the end.} \end{aligned} \tag{2.18}$$

Finally, for the generation of circular turns, the smoothed corner is defined as

$$\text{smoothed-corner} = [x' \quad y' \quad \text{Circularity}(\theta)']. \tag{2.19}$$

Once the path smoothing process is complete using the equations (2.16), (2.17), (2.18), and (2.18) the trajectory is saved. Figure 2.17 compares unsmoothed and smoothed paths. The left side shows a smoothed path, resulting in rounded corners. Conversely, the right side displays an L-shaped path with unsmoothed edges, producing chamfered corners.

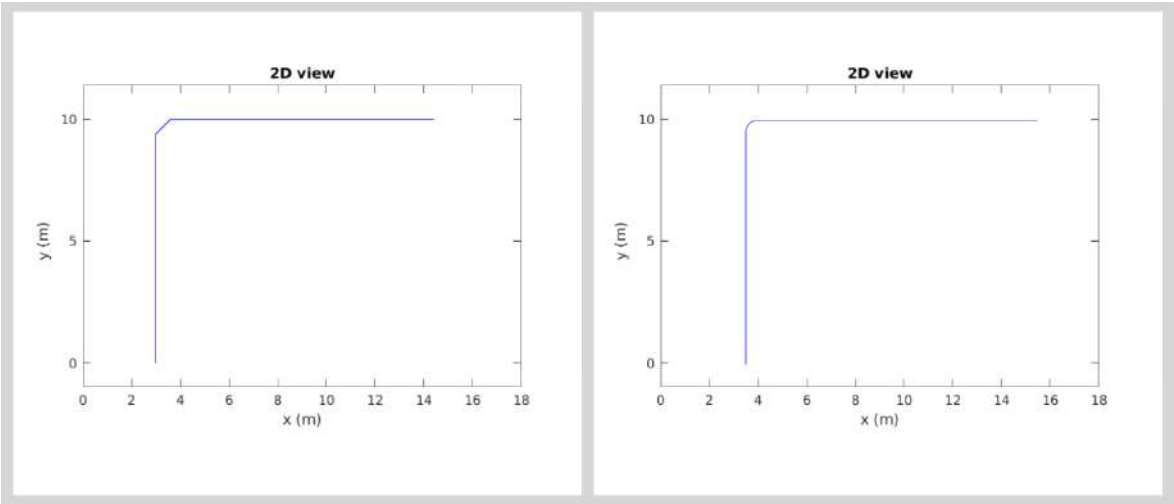


Figure 2.17. Comparison between unsmoothed (left) and smoothed (right) corners.

2.2.2. Sensor Fusion for Localization

This section explores the fundamental concepts and practical applications of sensor fusion using the EKF. It demonstrates how EKF-based sensor fusion enhances the localization of Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs). It details how EKF contributes to the control effectiveness in these systems by improving localization accuracy and reliability.

2.2.2.1. Sensor Fusion with Extended Kalman Filter.

(i) Sensor fusion merges data from multiple sensors like GPS, IMU, camera odometry, and gyroscope, improving the accuracy of a robot's position and orientation. It overcomes individual sensor limitations, enhancing localization performance. Techniques include Bayesian filters, Kalman filters, and particle filters.

(ii) Extended Kalman Filter (EKF) is used to enhance robot localization and cooperative localization. The EKF is a recursive algorithm handling nonlinearities in sensor measurements and dynamic systems, commonly used in robotics for data fusion. It maintains a robot's state probability distribution, merging past

estimates with current sensor data for precise localization. The algorithm effectively manages dynamic and noisy environments. The EKF includes two steps: prediction and correction.

(a) Prediction Step (Time Update): Let $\hat{x}_0$ and P_0 are initial estimates for state and covariance respectively. The time update step, also known as the prediction step, calculates the state estimate at time t. Let $\hat{x}t$ denote the state estimate at time t, Φ be the process model that describes how the state evolves over time, $\hat{x}t-1$ be the previous state estimate, and C_t the control input at time t. The calculation for the predicted state is given by

$$\hat{x}_t = \Phi(\hat{x}_{t-1}, C_t). \tag{2.20}$$

In the EKF, the covariance matrix P_t represents the uncertainty of the state estimate. The matrix A_t denotes the Jacobian matrix of the process model, P_{t-1} is the error covariance at the previous time step, and Q_t is the process noise covariance. During the time update step, the covariance matrix P_t and state estimate $\hat{x}_t$ are predicted together. The covariance matrix P_t is calculated as

$$P_t = A_t P_{t-1} A_t^\intercal + Q_t. \tag{2.21}$$

The expression in equation (2.20) provides the predicted state estimate and the expression in equation (2.21) captures the uncertainty in this predicted state. The covariance matrix tells us how much each element of the state estimate might deviate from its predicted value, and shows the correlations between different state variables. A higher covariance value implies a higher uncertainty in the relevant state variable. The covariance matrix is updated in the EKF based on the process model and measurement model during the measurement update step. This incorporates new sensor measurements and refines the state estimate further. Effective management of covariance is crucial, as it affects the trustworthiness of the filter's predictions and the overall performance of the estimation process.

(b) Correction Step (Measurement Update):In this step, the Kalman Gain (K_t) is computed. Let K_t represent the Kalman Gain at time t, H_t be the Jacobian matrix of the measurement model, N_t the measurement noise covariance, z_t the actual measurement at time t, and h the model that predicts the measurement based on the current state estimate. The updated state estimate at time t is expressed as $\hat{X}_t$, and the updated covariance estimate at time t is denoted as P_t, and the identity matrix is denoted as I. The Kalman Gain K_t, the updated state estimate $\hat{X}_t$, and the updated covariance estimate P_t are calculated as

$$K_t = P_t H_t^\intercal (H_t P_t H_t^\intercal + N_t)^{-1} \tag{2.22}$$

$$\hat{X}_t = \hat{x}_t + K_t(z_t - h(\hat{x}_t)) \tag{2.23}$$

$$P_t = (I - K_t H_t) P_t \tag{2.24}$$

Equations (2.22), (2.23), and (2.24) respectively provide the formal expressions for computing the Kalman Gain, updating the state estimate, and updating the covariance estimate.

Sensor fusion with EKF transforms disorganized data into accurate, reasonable information as shown in Figure 2.18.

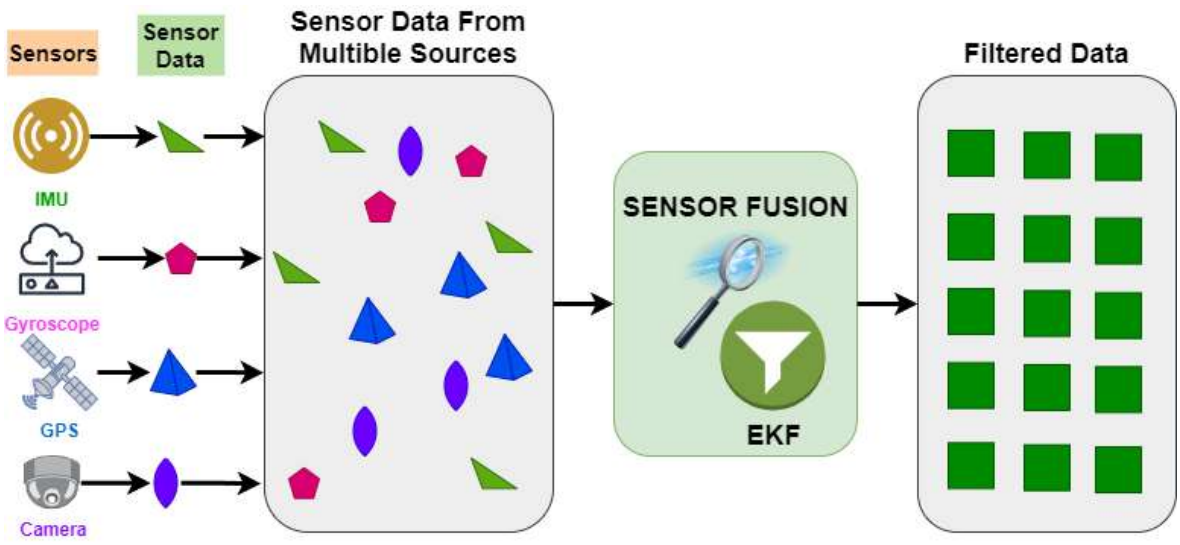


Figure 2.18. The diagram illustrating sensor fusion.

2.2.2.2. Robot Operating System (ROS). This thesis is conducted within the framework of ROS which is an open-source platform designed specifically for the development of robotic systems. It provides a comprehensive set of tools and libraries that simplify development, integration, and testing.

Its modular design is particularly useful for managing complex tasks such as sensor fusion and localization in robotics. This thesis explores these complexities, focusing on applying the EKF within the ROS platform using the 'Robot Localization' package.

Modeling the robot's kinematics and dynamics, as well as establishing the necessary frames are essential steps in the localization process. Below is a detailed breakdown of these steps.

(i) Unified Robotics Description Format (URDF): The URDF file outlines the robot's physical layout, including components such as links, joints, sensors, and their interconnections. Each rigid part of the robot is depicted by links, while the relationships between links are portrayed by joints. Various types of joints, such as revolute, prismatic, continuous, and fixed, are supported. Link elements possess visual, collision, and inertial properties. URDF permits the definition of reference frames relative to each link or joint, which can be utilized to stipulate sensor positions, controller reference frames, or other significant locations. In essence, URDF offers a standardized method to represent the physical structure of robots in ROS, which aids in the simulation, visualization, and control of robotic systems. Illustrations of URDF models for the UAV and the UGV are given in Figure 2.19.

(ii) Odometry: The robot model requires the implementation of odometry calculations based on sensor feedback. These calculations are used to estimate the robot's position and orientation relative to a reference frame. The odometry data

should then be published on the odometry topic to provide continuous updates about the robot's pose.

(iii) Robot Map Frame: The robot's map frame, which serves as the global reference for localization, needs to be defined. Other frames, including the base link frame, sensor frames, and any additional frames necessary for sensor fusion or navigation, should also be defined. These frames must align correctly and link in accordance with the robot's physical structure. As illustrated in Figure 2.19, each link of the robot should be transformed into the robot's base link (BFF). Following this, the base link should be transformed to the robot's map (IF) using odometry data for localization.

(iv) TF(Transform) Tree: The "TF" package in ROS manages coordinate transformations between different frames. To transform data between frames, you must publish it using the TF broadcaster. Maintaining a consistent and accurate TF tree is crucial for integrating sensor data and localization algorithms smoothly. The TF trees for the UAV and the UGV are provided below. As shown in the figures, each robot has a namespace - "quad1" for the UAV and "ryobi1" for the UGV. This prevents confusion and incorrect transformations of the links. In ROS multi-robot systems, messages, data, and transformations use the robot's namespace.

(v) Sensor Fusion and Localization: To enhance the accuracy of the robot's pose estimation, sensor fusion algorithms like EKF, UKF (Unscented Kalman Filter), or MCL should be implemented. These algorithms merge odometry data with data from other sensors, such as LiDAR, cameras, or GPS. The refined pose estimate can then be shared on the filtered odometry topic or another appropriate topic for localization information. It's also important to understand the data types used for sensor fusion.

There are two main categories of sensor data types for localization: Absolute and Relative, which are described below.

- Absolute sensor data indicates the robot's exact position or orientation within a global coordinate system, generally in relation to the Earth's sur-

face.

- Relative sensor data refers to the changes in position or orientation compared to a previous state or reference point. This data is typically gathered from onboard sensors that monitor alterations in motion, including velocity, acceleration, and angular rate.

Sensor data is published to ROS topics using various types of messages. These messages are used for localization and include odometry, twist, pose, and GPS-fix as follows [21]:

- Pose Message (nav_msgs/Pose Message) contains pose data in free space, represented by both a point position, indicating its location, and a quaternion, signifying its orientation.
- Twist Message (nav_msgs/Twist Message) contains velocity data in free space, represented by both a linear velocity, for its straight-line speed, and an angular velocity, for its rotational speed.
- Odometry Message (nav_msgs/Odometry Message) provides an estimate of position and velocity in free space. The pose should be specified in the coordinate frame defined by header.frame_id. Similarly, the twist should be defined in the coordinate frame given by child_frame_id.
- GPS-Fix Message (sensor_msgs/NavSatFix) contains GPS data, such as latitude, longitude, altitude, and satellite details. It can be used for corrections in localization.

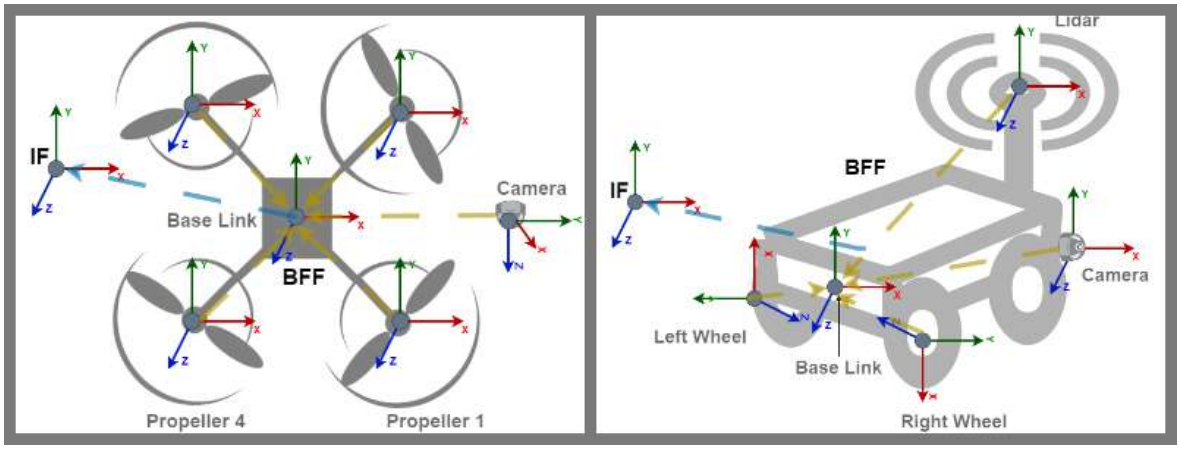


Figure 2.19. The diagram illustrating the URDF model for the robots.

2.2.3. Improvement Localization for The UAV

This section provides an in-depth guide on how to enhance the localization for the UAV. It covers essential steps such as the proper setting of the sensor set, which is of paramount importance for the accurate determination of the UAV's position and orientation. Furthermore, it discusses the crucial role of data diversity in ensuring that the UAV receives a wide range of information, thus allowing it to better understand its environment. In addition to that, this section offers detailed instructions on how to configure the EKF. EKF plays a vital role in estimating the state of a nonlinear system, and in this case, it is used for localization within a ROS environment. By following these guidelines, one can significantly improve the effectiveness of UAV localization.

2.2.3.1. Sensor Set of The UAV. The UAV is equipped with various sensors, as detailed in Table 2.4 below. Some sensors, such as the IMU, provide differential data, similar to velocity. Others, like the stereo camera, provide non-differential data, similar to position. Some, like the 3D Lidar (also known as a depth camera), provide both. The goal is to utilize a range of sensors from different sources and data types to enhance localization.

Table 2.4. Sensor types and sources

Sensor Type	**Differential**	**Source**
IMU	Yes	ZEDm Stereo Camera
Gyroscope	Yes	ZEDm Stereo Camera
Accelerometer	Yes	ZEDm Stereo Camera
Stereo Camera	No	ZEDm Stereo Camera
Depth Camera (3D Lidar)	Yes	ZEDm Stereo Camera
GPS (Base Corrected)	No	Emlid Reach M+ RTK GNSS
GPS	No	Pixhawk Flight Controller
Barometer	No	Pixhawk Flight Controller

If data is differential, then a measurement taken at time t from the sensor needs to be subtracted from the measurement at time $t-1$, and then converted to the resultant value. This configuration is particularly useful if the robot has two sources of absolute pose information, such as yaw measurements from odometry and an IMU. If the variances on the input sources are not correctly configured, these measurements may become out of sync with one another, causing oscillations in the filter. However, by integrating one or both of them differentially, this situation can be avoided [21].

2.2.3.2. ROS Configurations with EKF for UAV Localization. The UAV is equipped with the sensor set mentioned in the previous section. Each sensor operates within its own coordinate frame. Therefore, data from each sensor needs to be transformed to the base-link, which is the UAV's origin. This transformation is carried out using the TF package, with ROS nodes coded in Python. As shown in Figure 2.20 below, sensor data is initially collected according to the sensor's coordinate frame. It is then transformed to the UAV's base-link (BFF). As the UAV moves, its sensor data changes accordingly. These transformations enable the detection of both the distance and direction of movement relative to the IF.

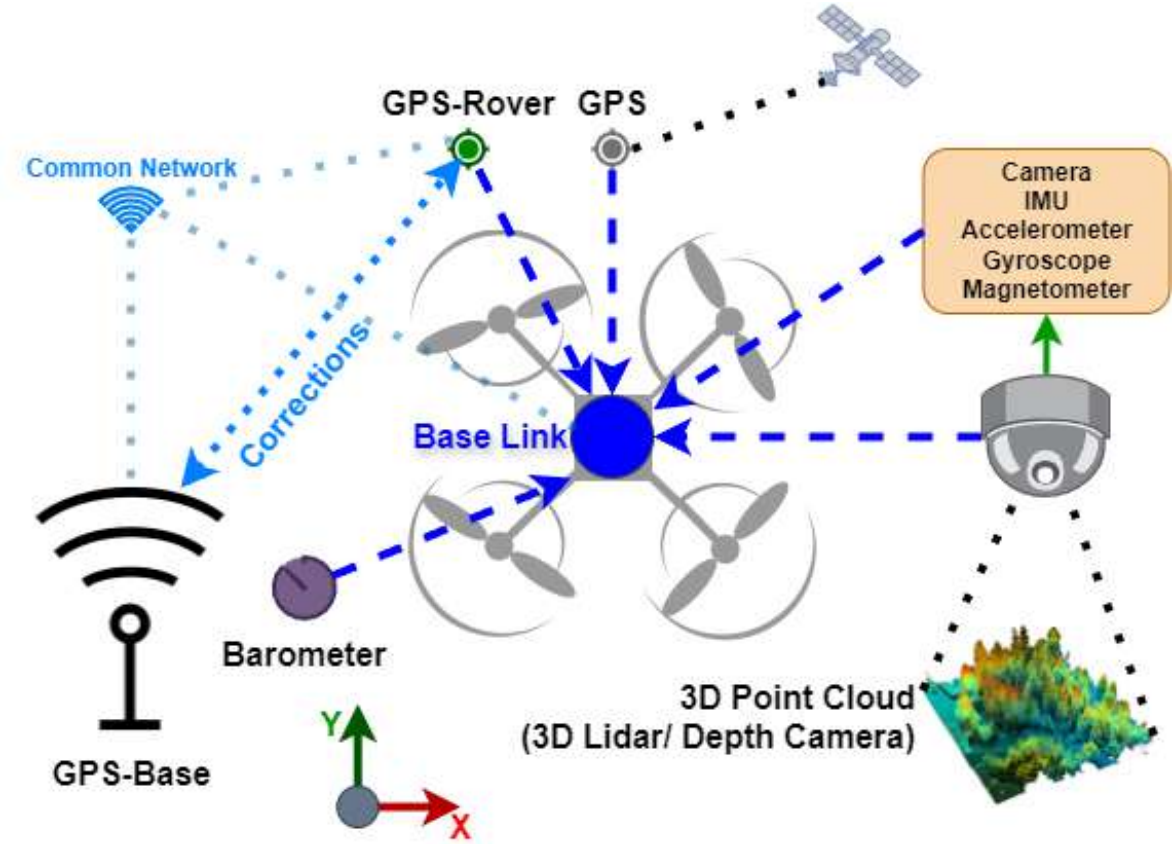


Figure 2.20. The illustration of sensor data transformations (TF) for the UAV.

The "Robot Localization" package's EKF localization node is used to apply EKF for localization within the ROS environment. Five ROS topics are generated from the sensors and used as data sources for EKF-based localization. These topics are detailed in Table 2.5 below. Because the magnetometer sensor is sensitive to magnetic fields, it is not included as a data source for EKF.

Table 2.5. EKF-Applied localization source topics for the UAV

Source Topic	**Source Sensor(s)**	**Differential**
\zedm\odom	ZEDm (Camera & 3D Lidar & Accelerometer)	Yes
\zedm\imu	ZEDm (IMU)	Yes
\zedm\pose	ZEDm (Camera & Gyroscope)	No
\tcp\fix	Emlid Reach M+ RTK GNSS (GPS)	No
\mavros\gps	Pixhawk Flight Controller (GPS)	No

In the Robot Localization ROS package, a configuration matrix must be created for each source topic, as illustrated in Table 2.6 below. This 15x3 matrix indicates position, velocity, and acceleration in 3D space for the topic. Elements in the matrix are assigned boolean values of true or false. If an element's value is to be used in the EKF, it should be set to true. Conversely, GPS data is only used in the EKF for correction to enhance localization. The configuration of the UAV's EKF for each source topic is detailed in Table 2.7.

Table 2.6. Robot localization package-source configuration matrix

Value/Content	**Matrix Elements**
Position	$x \mid y \mid z$
Orientation	$\phi \mid \theta \mid \psi$
Linear Velocity	$\dot{x} \mid \dot{y} \mid \dot{z}$
Angular Velocity	$\dot{\phi} \mid \dot{\theta} \mid \dot{\psi}$
Linear Acceleration	$\ddot{x} \mid \ddot{y} \mid \ddot{z}$

Table 2.7. Sensor data availability for the UAV's EKF localization

Element	**x**	**y**	**z**	ϕ	θ	ψ	$\dot{x}$	$\dot{y}$	$\dot{z}$	$\dot{\phi}$	$\dot{\theta}$	$\dot{\psi}$	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$
Odom	+	+	+	-	-	+	+	+	+	-	-	+	+	+	+
Pose	+	+	+	-	-	+	-	-	-	-	-	-	-	-	-
IMU	-	-	-	-	-	-	+	+	+	+	+	+	+	+	+

2.2.4. Improvement Localization for The UGV

This section offers a detailed guide on improving the localization of the UGV. It outlines critical steps such as the correct configuration of sensors, vital for accurately identifying the UGV's position and orientation. The guide emphasizes the need for data diversity, which gives the UGV a wide range of information, enhancing its environmental understanding. Moreover, it provides direction on configuring the EKF, a crucial tool for state estimation in nonlinear systems, used here for localization in a ROS environment. By following these guidelines, UGV localization can be significantly enhanced.

2.2.4.1. Sensor Set of The UGV. The UGV is similar to the UAV in that it is equipped with various sensors, as detailed in Table 2.8 below. The aim is to use a variety of sensors from different sources and data types to improve localization. Additionally, the UGV is equipped with a 2D Lidar sensor for safety, obstacle avoidance, and SLAM applications.

Table 2.8. Sensor types and sources

Sensor Type	**Differential**	**Source**
IMU	Yes	ZED2 Stereo Camera
Gyroscope	Yes	ZED2 Stereo Camera
Accelerometer	Yes	ZED2 Stereo Camera
Stereo Camera	No	ZED2 Stereo Camera
Magnetometer	Yes	ZED2 Stereo Camera
Depth Camera (3D Lidar)	Yes	ZED2 Stereo Camera
GPS (Base Corrected)	No	Emlid Reach M+ RTK GNSS
2D Lidar	No	RPlidar S1

2.2.4.2. ROS Configurations with EKF for UGV Localization. The sensor set of the UGV is discussed in the previous section. Each sensor operates within its distinct coordinate frame. Thus, the data from each sensor must be transformed to the base link, which serves as the UGV's origin (BFF). Like the UAV configuration, the UGV's sensor frame transformations are performed using similar ROS nodes as illustrated in Figure 2.21 below. The EKF configuration is also carried out, and the sensor topics are presented in Table 2.9. The EKF configuration for each sensor topic on the UGV is provided in Table 2.10.

Table 2.9. EKF-Applied localization source topics for the UGV

Source Topic	**Source Sensor(s)**	**Differential**
\zed2\odom	ZED2 (Camera & 3D Lidar & Accelerometer)	Yes
\zed2\imu	ZED2 (IMU)	Yes
\zed2\pose	ZED2 (Camera & Gyroscope)	No
\tcp\fix	Emlid Reach M+ RTK GNSS (GPS)	No

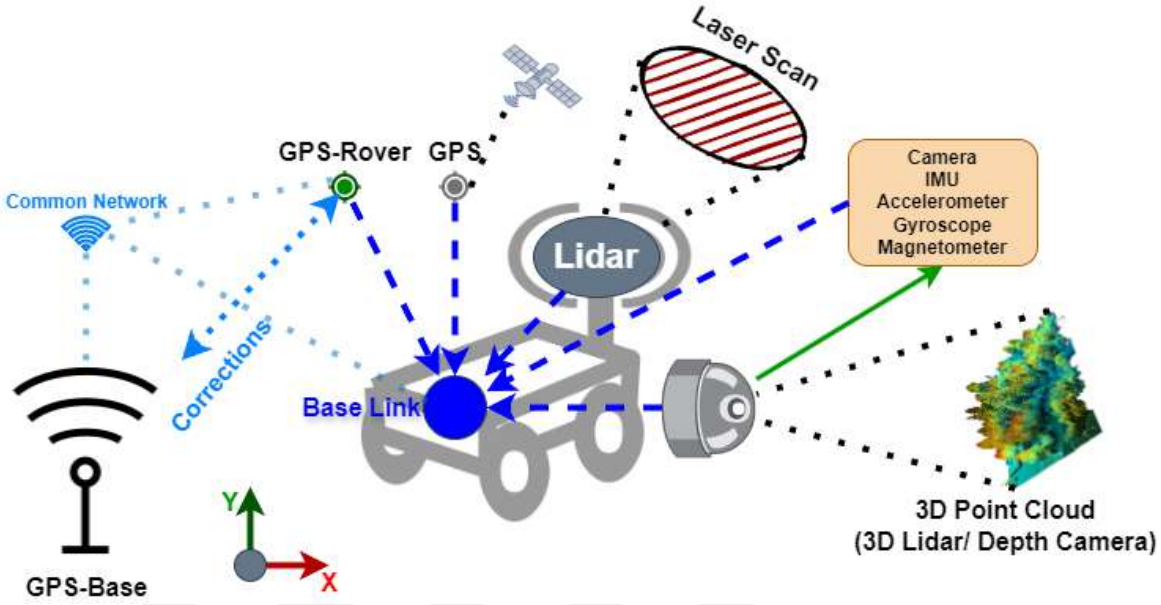


Figure 2.21. The illustration of sensor data transformations (TF) for the UGV.

Table 2.10. Sensor data availability for the UGV's EKF localization

Element	**x**	**y**	**z**	ϕ	θ	ψ	$\dot{x}$	$\dot{y}$	$\dot{z}$	$\dot{\phi}$	$\dot{\theta}$	$\dot{\psi}$	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$
Odom	+	+	+	-	-	+	+	+	+	-	-	+	+	+	+
Pose	+	+	+	-	-	+	-	-	-	-	-	-	-	-	-
IMU	-	-	-	-	-	-	+	+	+	+	+	+	+	+	+

2.2.5. Control Theory

This section delves into robot control. It uses a reference path to gauge the enhancement in localization accuracy by comparing it with the robot's actual trajectory. The validity of this comparison depends significantly on the robot's precision in adhering to the reference path. Accurate trajectory following is integral for dependable localization results. This section will explore the applications of PID and MPC control concepts in the trajectory following.

2.2.5.1. Proportional-Integral-Derivative (PID) Control. The PID controller is a straightforward control law. It considers the error variable (P for "proportional"), the integral (I for "integral"), and the derivative of the error variable (D for "derivative") [22]. The time-domain control law involves the following components:

- $U(t)$ represents the control input.
- K_p represents the proportional gain.
- K_i represents the integral gain.
- K_d represents the derivative gain.
- $e(t)$ represents the error signal at time t.
- $\int_0^t e(\tau)\,d\tau$ is the integral of the error signal from time 0 to t, where τ is the integration variable.
- $\frac{de(t)}{dt}$ denotes the derivative of the error with respect to time t.

The time-domain control law is represented as

$$U(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{de(t)}{dt}, \tag{2.25}$$

where equation (2.25) combines these components to form the control input.

In this thesis, the PID control approach is applied to both the UAV and UGV for trajectory following. The trajectory following requires three PID controllers for the robot's longitudinal, lateral, and angular motion. The Tracking-pid ROS package is used to control the robot to follow the desired trajectory. K_p, K_i, and K_d gains for each controller are fine-tuned on MATLAB for the simulation environment. During experimental tests, these gains are adjusted empirically for each controller. The illustration of the PID controllers in the trajectory following for a robot is given in Figure 2.22.

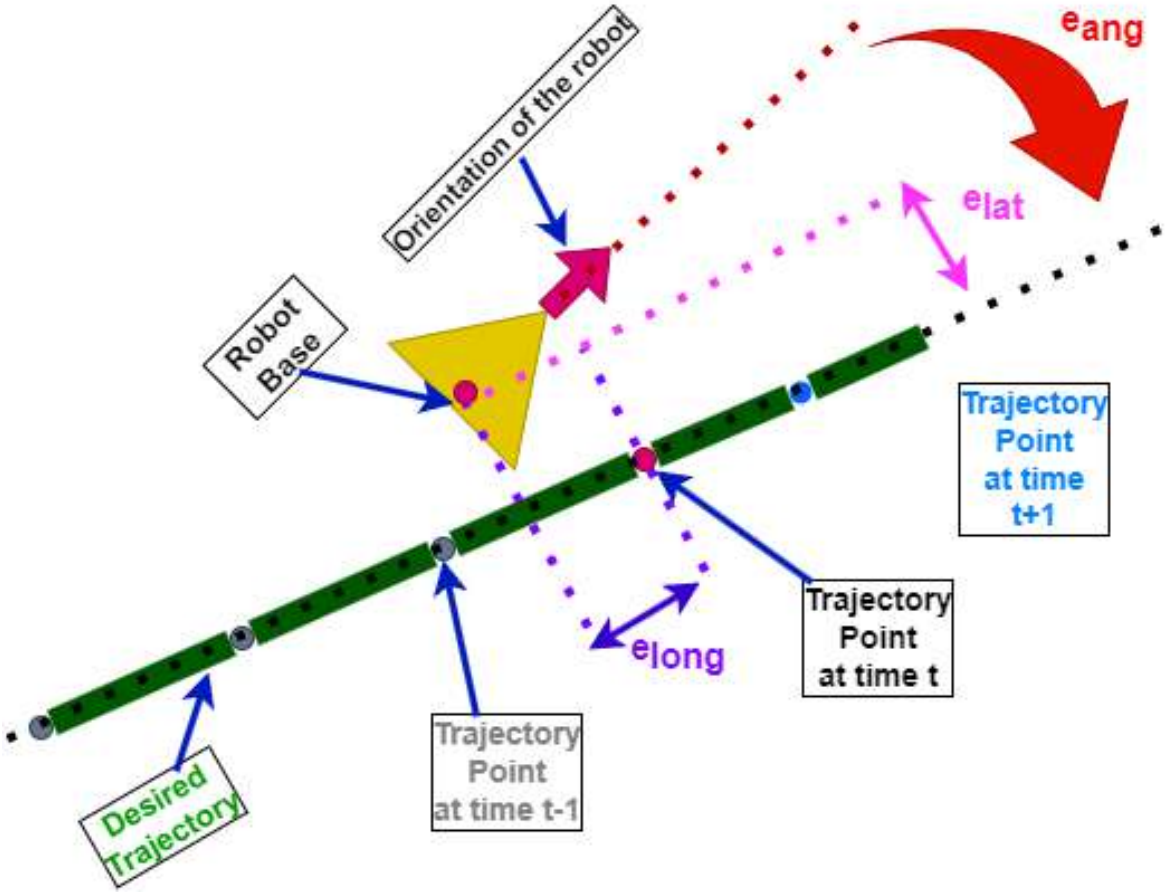


Figure 2.22. The illustration of the PID controllers.

- Longitudinal PID Controller: This controller mainly handles the robot's forward and backward movements. The longitudinal error, denoted as e_{long}, quantifies the discrepancy between the desired and actual positions along the robot's longitudinal axis. The longitudinal control effort, denoted as U_{long}, is calculated as

$$U_{long}(t) = K_p e_{long}(t) + K_i \int_0^t e_{long}(\tau)\, d\tau + K_d \frac{de_{long}(t)}{dt}. \tag{2.26}$$

- Lateral PID Controller: This controller mainly handles the robot's lateral movement. Lateral error, expressed as e_{lat}, represents the difference between the desired and actual positions along the robot's lateral axis. The longitudinal control

effort, denoted as U_{lat}, is calculated as

$$U_{lat}(t) = K_p e_{lat}(t) + K_i \int_0^t e_{lat}(\tau)\, d\tau + K_d \frac{de_{lat}(t)}{dt}. \tag{2.27}$$

- Angular PID Controller: This controller is primarily used to manage the robot's orientation or heading. The angular error, denoted as e_{ang}, measures the difference between the desired and actual orientation of the robot. The angular control effort, denoted as U_{ang}, is calculated as

$$U_{ang}(t) = K_p e_{ang}(t) + K_i \int_0^t e_{ang}(\tau)\, d\tau + K_d \frac{de_{ang}(t)}{dt}. \tag{2.28}$$

These three PID controllers given in the equations (2.26), (2.27), and (2.28) create a twist message for the ROS environment to define the UGV's velocity for longitudinal on the x axis, lateral on the y axis, and angular (yaw) around the z axis. The Roboteq motor controller's ROS-compatible driver package subscribes to this twist message and uses the UGV's kinematics to control the motors and achieve the desired movement. The longitudinal controller provides output for linear movement, while the lateral and angular PID controllers provide output for angular movement.

2.2.5.2. Model Predictive Control (MPC). Model Predictive Control (MPC) is a more advanced control concept compared to the PID controller. In MPC, an internal model is used at each time step to predict system behavior over a predefined horizon. Then, an optimal control input sequence that minimizes a certain cost function is generated. This method allows for the consideration of different types of constraints for states and inputs, which is a significant advantage of MPC [22].

However, the primary limitation of MPC is its high computational complexity, which restricts its use in real-time applications. It also requires an explicit model for prediction and precise advanced knowledge of external inputs and disturbances for the entire prediction horizon. Below is a simplified representation of an MPC formulation. In this formulation:

- N_c is the control horizon,

- N_p is the prediction horizon, where $N_c \leq N_p$,
- $x_{\min}$ and $x_{\max}$ are the constraints for the states,
- $u_{\min}$ and $u_{\max}$ are the constraints for the inputs,
- The subscript $k+1|t$ denotes the value of the variable $k+1$ steps ahead of the current time t [22].

The MPC formulation is given by

$$\min_{u_{0|t},\ldots,u_{N_c-1|t}} \sum_{k=0}^{N_p-1} x_{k|t}^T Q x_{k|t} + \sum_{k=0}^{N_c-1} u_{k|t}^T R_w u_{k|t},$$

subject to

$$\begin{aligned} & x_{k+1|t} = Ax_{k|t} + Bu_{k|t}, \quad k = 0, 1, \ldots, N_p - 1, \\ & x_{\min} \leq x_{k+1|t} \leq x_{\max}, \quad k = 0, 1, \ldots, N_p - 1, \\ & u_{\min} \leq u_{k|t} \leq u_{\max}, \quad k = 0, 1, \ldots, N_c - 1, \\ & u_k = 0, \quad N_c \leq k \leq N_p. \end{aligned} \tag{2.29}$$

In this thesis, the MPC approach based on equation (2.29), implemented by the MPC-Tracker ROS package, is solely applied for altitude control to ensure safe flight for the UAV.

2.2.5.3. Kinematics and Dynamics Analysis of the UAV. A quadcopter, also known as a quadrotor helicopter or simply a quadrotor, is a type of UAV or drone. It is driven and controlled by four rotors. The quadcopter's design is a symmetrical cross-shaped frame, with each arm ending in a rotor. These rotors spin rapidly in opposite directions to generate lift and thrust. The quadcopter can maneuver in the air by independently adjusting the speed and rotation direction of each rotor. Figure 2.23 below provides a general schematic of a X-configured quadcopter's kinematics.

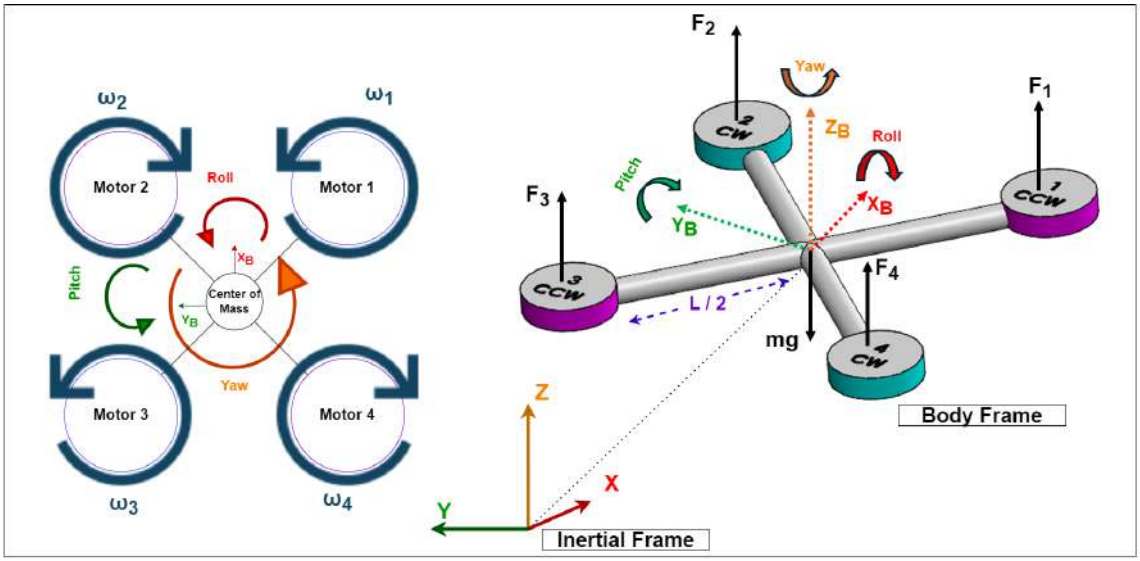


Figure 2.23. Schematic diagram for the kinematics of a quadcopter.

The Inertial Frame (IF) is denoted by the [X,Y,Z] axis system, while the Body Fixed Frame (BFF) is represented as [X_B,Y_B,Z_B]. The dynamic equations can be simplified using Newton-Euler formalism, based on certain assumptions that help model the dynamics.

- The quadcopter has a rigid and symmetrical mechanical structure, with all its connections firm.
- The quadcopter's center of mass is at the origin of the BFF.
- The propellers are of the same size and pitch, and they are rigid.
- Each propeller-motor pair delivers thrust and moment proportional to the square of the propeller's speed.
- The aerodynamic effects of air friction, except for those contributing to thrust generation, and the gyroscopic effects caused by the relative angular velocity of the propellers to the quadcopter's body, are neglected.

The IF is the coordinate frame where the quadcopter moves. The quadcopter's position and orientation can be represented by mathematical expressions. Specifically, let $\xi \in \mathbb{R}^3$ denote the position and $\eta \in \mathbb{R}^3$ denote the orientation in the IF. The position

ξ is given by

$$\xi = [x \quad y \quad z]^T, \tag{2.30}$$

and the orientation η is given by

$$\eta = [\phi \quad \theta \quad \psi]^T. \tag{2.31}$$

The quadcopter's orientation in the BFF with respect to the IF is defined using the Euler angles (ϕ, θ, ψ), which represent the roll-rotation on the X_{B} axis, pitch-rotation on the Y_{B} axis, and yaw-rotation on the Z_{B} axis, respectively. The translational and rotational velocities of the quadcopter in the BFF are denoted as $V_{\mathrm{B}} \in \mathbb{R}^3$ and $\omega \in \mathbb{R}^3$, respectively. The translational velocity V_{B} and rotational velocity ω can be found by taking the derivative of the positions given as

$$V_{\mathrm{B}} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{2.32}$$

$$\omega = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{2.33}$$

The dynamics of the quadcopter are defined in the BFF, while its motion occurs in the IF. To perform transformations and express vectors from the IF to the BFF, a rotation matrix R_{BI} is needed. This matrix R_{BI} can be represented by the sequence of rotations: roll (ϕ), pitch (θ), and yaw (ψ), with the corresponding rotation matrices R_ϕ, R_θ, and R_ψ for each rotation. These matrices combine as

$$\begin{aligned} R_{\mathrm{BI}} &= R_\phi R_\theta R_\psi, \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & s(\phi) \\ 0 & -s(\phi) & c(\phi) \end{pmatrix} \begin{pmatrix} c(\theta) & 0 & -s(\theta) \\ 0 & 1 & 0 \\ s(\theta) & 0 & c(\theta) \end{pmatrix} \begin{pmatrix} c(\psi) & s(\psi) & 0 \\ -s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}, \\ &= \begin{pmatrix} c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\ s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & s(\phi)c(\theta) \\ c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)c(\theta) \end{pmatrix}. \end{aligned} \tag{2.34}$$

The rotation matrix is denoted as R_{BI} in equation (2.34), where $c(\cdot)$ and $s(\cdot)$ represent cosine and sine functions, respectively.

On the other hand, the translational velocity in the IF (V_{I}), and in the BFF (V_{B}) for the quadcopter can be written as

$$V_{\mathrm{I}} = R_{\mathrm{BI}} V_{\mathrm{B}}, \tag{2.35}$$

$$R_{\mathrm{IB}} = R_{\mathrm{BI}}^{-1} = (R_{\mathrm{BI}})^{T}, \tag{2.36}$$

$$V_{\mathrm{B}} = R_{\mathrm{IB}} V_{\mathrm{I}}. \tag{2.37}$$

The rotation matrices can be used to express the angular velocity in the BFF as

$$\begin{aligned} \omega = \begin{pmatrix} p \\ q \\ r \end{pmatrix} &= R_{\phi} R_{\theta} R_{\psi} \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} + R_{\phi} R_{\theta} \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + R_{\phi} \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} \\ &= R_{\mathrm{r}} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & s(\phi)c(\theta) \\ 0 & -s(\phi) & c(\phi)c(\theta) \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}. \end{aligned} \tag{2.38}$$

The dynamic equations outlined above are crucial in both the mechanical design phase and the control algorithms' development for the quadcopter. Equations (2.30) and (2.31) define the position and orientation of the quadcopter, respectively. Equations (2.32) and (2.33) define the translational and rotational velocities of the quadcopter, respectively. Equations (2.35), (2.36), and (2.37) are used to calculate the translational velocities in IF and BFF for the quadcopter. Equation (2.38) illustrates the expression for the angular velocity in the BFF using the rotation matrices.

2.2.5.4. MPC Settings for Altitude Controller.

- System Matrices: The continuous-time system matrix A_c and input matrix B_c are defined as

$$A_c = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2.39}$$

$$B_c = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{2.40}$$

- Sampling Time: The sampling time T_{samp} is set to 0.1 seconds.
- Discrete-Time Matrices: The discrete system matrix A_d and input matrix B_d are calculated as

$$A_{dis} = \sum_{i=0}^{\infty} \frac{(A_c \cdot T_{samp})^i}{i!}, \tag{2.41}$$

$$B_{dis} = \begin{bmatrix} \frac{T_{samp}^3}{6} \\ \frac{T_{samp}^2}{2} \\ T_{samp} \end{bmatrix}. \tag{2.42}$$

- Horizon Length: The prediction horizon N_c is set to 20 steps.
- Weighting Matrices: The control effort weighting R_e is 1, and the state error weighting matrix Q_e is defined as

$$Q_e = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.43}$$

- Control Law: Let F_g is the LQR(Linear Quadratic Regulator) gain matrix calculated using the discrete-time algebraic Riccati equation. The control input u can be calculated as

$$u = -F_g \Delta z. \tag{2.44}$$

The altitude control for the quadcopter is obtained applying MPC to the thrust using equations (2.39), (2.40), (2.41), (2.42), (2.43), and (2.44) expressed above.

2.2.5.5. Kinematics and Dynamics Analysis of the UGV. Differential drive kinematics is a favored configuration for UGVs. It offers simple control and maneuverability. In this system, two wheels on a common axis are independently powered and controlled. This allows the UGV to rotate in place by spinning one wheel while the other remains stationary. However, due to kinematic constraints, a differential drive vehicle cannot move laterally, resulting in a constant lateral velocity of zero. For simplicity in modeling, it is also assumed that the wheels are always in contact with the ground and the vehicle moves without slipping. This design is simple and easy to control, making it suitable for various applications. By adjusting the speeds and directions of the two wheels, the UGV can perform a variety of movements, including forward and backward motion, and precise turns. This flexibility is particularly helpful in navigating tight spaces and conducting complex tasks in challenging environments. Figure 2.24 below provides a general schematic of a differential drive robot's kinematics. When the robot moves along a curve, the center of the curve is the Instantaneous Center Of Curvature (ICR). X and Y represent the IF coordinate axes, and x, and y represent the position coordinates with respect to IF, while X_B and Y_B denote the BFF coordinate axes. S is the distance between the wheel centers, and R_{ICR} is the distance from ICR. ψ represents the robot's orientation with respect to the IF. ω indicates rotation around the Z axis of the BFF (yaw). v_{left} and v_{right} refer to the wheels' translational velocities, V_B refers to the translational velocity vector of the robot in BFF, v_{ugv} refers the translational velocity of the UGV, and ω_{ugv} denotes the robot's angular velocity [23,24].

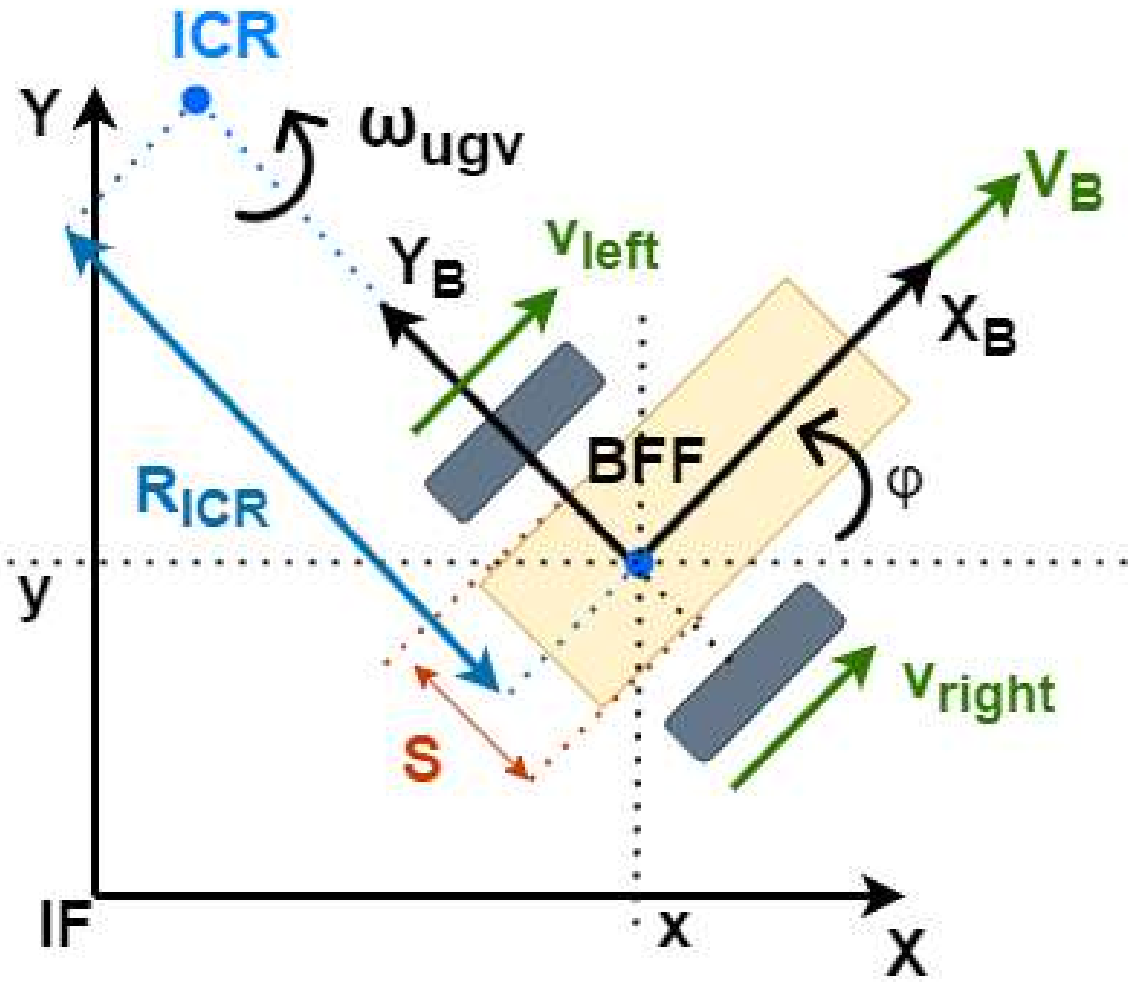


Figure 2.24. Schematic diagram for the kinematics of a differential drive robot.

The translational velocities of both wheels can be calculated as follows. Let v_{left} and v_{right} represent the velocities of the left and right wheels, respectively. These velocities can be determined as

$$v_{\text{left}} = \omega_{\text{ugv}} \left(R_{\text{ICR}} - \frac{S}{2} \right), \tag{2.45}$$

$$v_{\text{right}} = \omega_{\text{ugv}} \left(R_{\text{ICR}} + \frac{S}{2} \right). \tag{2.46}$$

The angular velocity ω_{ugv}, which is defined as the difference between the right and left wheel velocities divided by the wheel separation, is calculated as

$$\omega_{\text{ugv}} = \frac{v_{\text{right}} - v_{\text{left}}}{S}. \tag{2.47}$$

The speed of the robot, v_{ugv}, which is the average of the left and right wheel velocities, is calculated as

$$v_{\text{ugv}} = \frac{v_{\text{right}} + v_{\text{left}}}{2}. \tag{2.48}$$

The rotation matrix R_ψ used for the transformation is described as

$$R_\psi = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.49}$$

The velocity matrices for the robot, V_B relative to BFF and V_I relative to IF can be determined using equations (2.45), (2.46), (2.47), and (2.48) as

$$V_I = \begin{bmatrix} v_{\text{ugv}} \\ 0 \\ \omega_{\text{ugv}} \end{bmatrix}, \tag{2.50}$$

$$V_B = \begin{bmatrix} v_{\text{ugv}} \cos(\psi) \\ v_{\text{ugv}} \sin(\psi) \\ \omega_{\text{ugv}} \end{bmatrix}, \tag{2.51}$$

$$V_I = R_\psi V_B. \tag{2.52}$$

while equations (2.50) and (2.51) express the velocity matrices relative to BFF and IF respectively, equation (2.52) expresses the relationship between these matrices using the rotation matrix that defined in equation (2.49).

The distance from ICR R_{ICR} can be written as

$$R_{\text{ICR}} = \frac{1}{2}\left(\frac{v_{\text{right}}+v_{\text{left}}}{v_{\text{right}}-v_{\text{left}}}\right). \tag{2.53}$$

The position of the ICR can be written as

$$\text{ICR} = \begin{bmatrix} x - R_{\text{ICR}} \sin(\psi) \\ y + R_{\text{ICR}} \cos(\psi) \end{bmatrix}, \tag{2.54}$$

The position of the UGV and position vectors can be written as

$$\begin{aligned} \xi &= \begin{bmatrix} x & y & \psi \end{bmatrix}^T, \\ &= \begin{pmatrix} x \\ y \\ \psi \end{pmatrix}, \\ &= \begin{pmatrix} \cos(\omega) & -\sin(\omega) & 0 \\ \sin(\omega) & \cos(\omega) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x - \text{ICR}_x \\ y - \text{ICR}_y \\ \psi \end{pmatrix} + \begin{pmatrix} \text{ICR}_x \\ \text{ICR}_y \\ \omega_{\text{ugv}} \end{pmatrix}. \end{aligned} \tag{2.55}$$

The equations expressed above (2.53), (2.54), and (2.55) are used to control the motors of the UGV in position control during trajectory tracking.

2.3. UTOPIA Framework And Coordinate Transformations

This section explores the creation of the UTOPIA system's global framework and coordinate transformations for heterogeneous mobile robots. The goal is to achieve cooperative localization and control, enabling the robots to carry out cooperative missions. As mentioned earlier, each robot is equipped with a set of sensors. Each sensor has its own coordinate frame, and the sensor data is transformed to the robot's base-link (BFF) using coordinate transformations. For localization data creation, an odometry frame needs to be established for each robot. The transformed sensor data is then used to apply sensor fusion with EKF to each robot, creating odometry data which is subsequently transformed to each robot's map frame. This allows tracking of a robot's velocity and position data in its map frame as it moves. The sensor data can also be visualized in its map with visualization software such as Rviz. To execute multi-robot tasks with cooperative localization and control, creating a global map is necessary. This process involves transforming each robot's individual map into the global map frame, as shown in Figure 2.25 below. Therefore, when a robot moves within its map,

its localization and sensor data are also transformed to the global map. This allows it to be tracked and recognized by the site manager and the other robots on the global map.

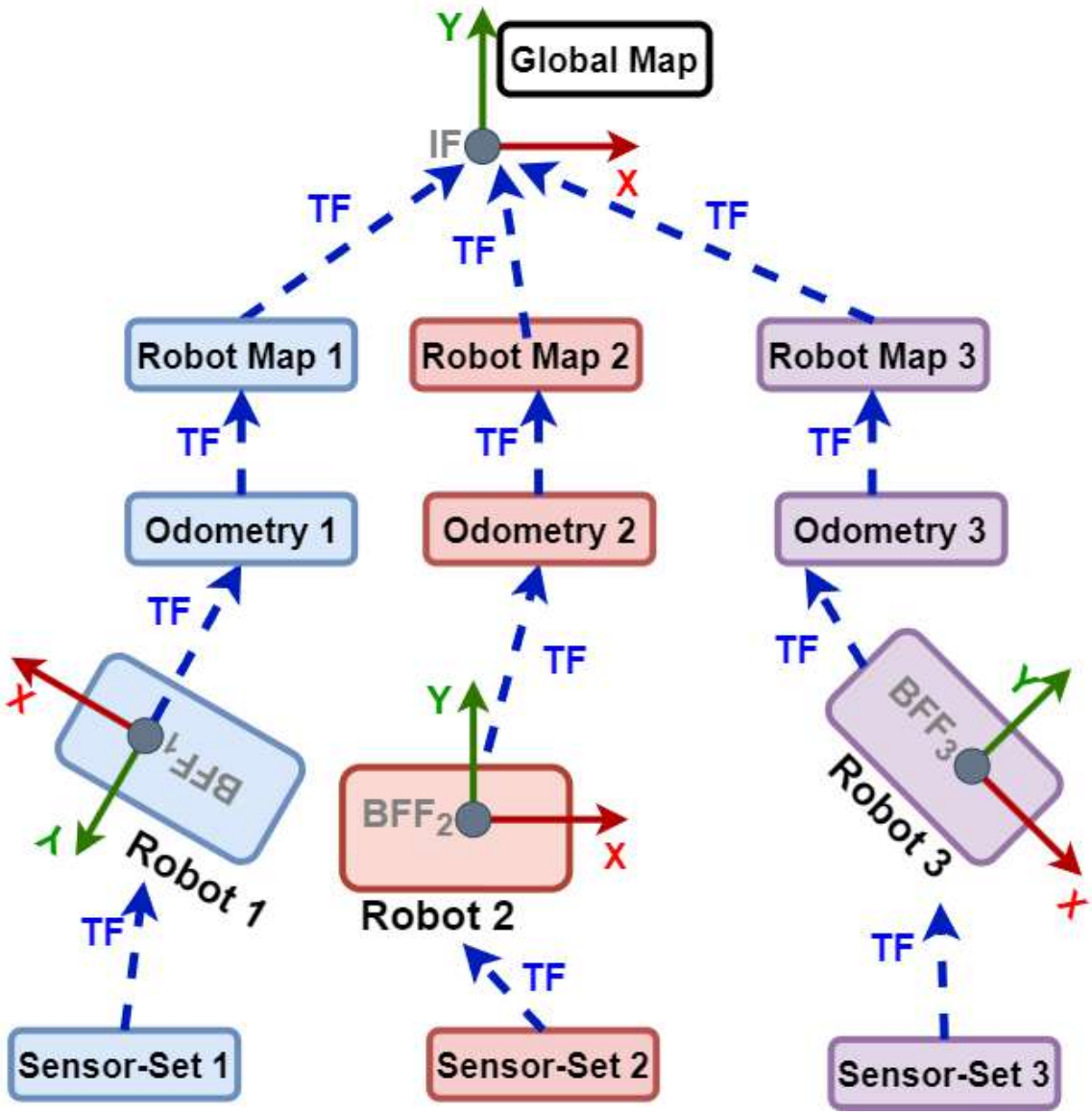


Figure 2.25. The illustration of a global framework for a multi-robot system.

Similarly, odometry and robot map frames for the UAV and UGV are set up. A global map frame is then established, and the map frames of the robots are transformed into this global frame using Python-coded ROS nodes. The coordinate transformations of the UAVs and UGVs, along with the visualization of the global framework on the

Rviz software, are illustrated in Figure 2.26.

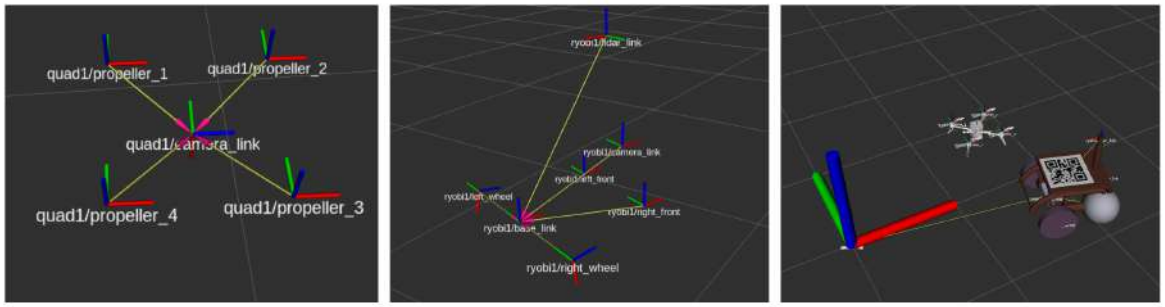


Figure 2.26. The Rviz visualization represents the coordinate transformations for the UAV (on the right), the UGV (in the middle), and the global framework (on the left), respectively.

2.4. UTOPIA's Cooperative Missions For Heterogeneous Mobile Robots

This section outlines the cooperative missions of the UAV and the UGV. It showcases their enhanced cooperative localization and control capabilities, which are the primary focus of this thesis. In order to enhance these capabilities for heterogeneous mobile robots, designing cooperative missions is necessary. These missions provide a platform to test control strategies and localization enhancements while the robots follow their prescribed trajectories.

2.4.1. Cooperative Mission 1: Trajectory Generation from Waypoints for Another Robot to Follow

The first mission illustrated in Figure 2.27 aims to demonstrate the UGV's trajectory following capability and the UAV's precision localization in collaboration. The UAV can be manually controlled via its RC, or it can follow a set trajectory. During its flight, a human actor selects the waypoints for the UGV to follow. Once waypoint selection is complete, the site manager creates a trajectory passing through the collected waypoints using equations (2.2-2.11). The UGV then follows this trajectory. This

mission utilizes the capabilities of heterogeneous mobile robots in cooperative tasks. This can be beneficial for agricultural operations and multi-robot operations in various fields such as the military, naval, and industry.

UAVs can fly much faster than UGVs and have a wider vision range. Instead of an UGV monitoring the entire field, a UAV can quickly scan the field and the operator can mark the areas that need to be monitored by the UGV. This scenario can be reversed, with the UGV marking waypoints, and then the UAV following them to provide bird's-eye monitoring.

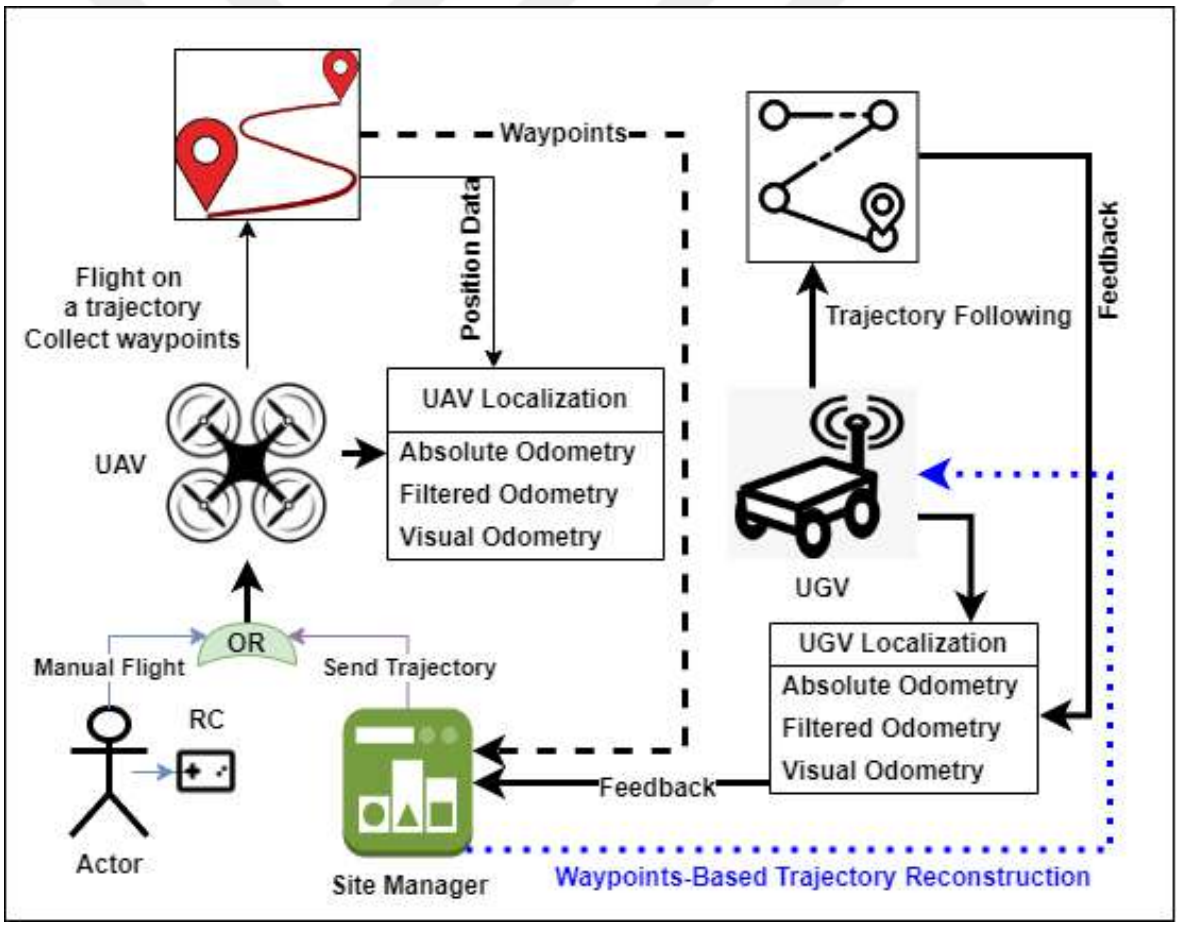


Figure 2.27. The illustration of the cooperative mission 1.

2.4.2. Cooperative Mission 2: Detection and Estimation of Localization for a Lost Robot Using Onboard Sensors and Cooperative Localization

The second mission, as illustrated in Figure 2.28, is designed to estimate another robot's location using one robot's sensor set. The goal of this scenario is to demonstrate that cooperative localization for heterogeneous mobile robots can be achieved.

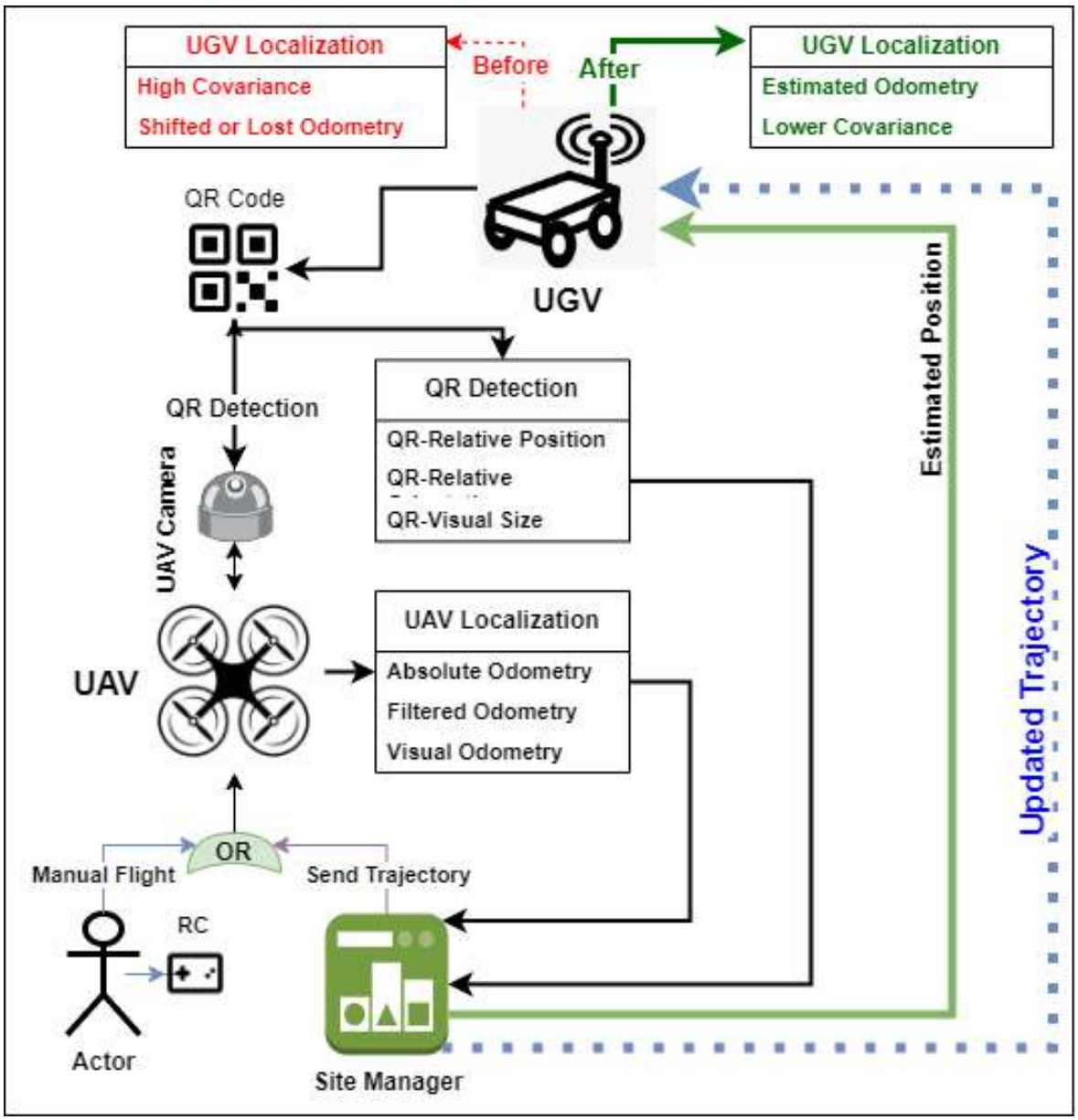


Figure 2.28. The illustration of the cooperative mission 2.

The UAV's camera is used for QR code detection, and the UAV can follow either a manual or autonomous trajectory. The UGV, due to the challenges of the agricultural environment, can have high covariance for localization or may lose sensor data such as GPS or visual odometry.

Issues like tree coverage can cause a GPS antenna to lose signal, creating dead zones. Similarly, due to the monotonous vision in an agricultural environment or changes in sunlight outdoors, visual odometry may shift or fail. In these cases, the UGV's localization is not reliable due to high covariance.

However, a QR code is placed on the base chassis of the UGV. The UAV can detect this QR code and calculate its orientation and distance. Subsequently, the site manager can estimate the UGV's location and manage the UGV accordingly. As a result, the UGV's estimated localization becomes more reliable and has a lower covariance.

2.4.2.1. QR Detection. The Pyzbar Python library and the Zbar ROS package are used to detect QR codes through the camera view. The algorithm first identifies the QR code and then compares its value. If the value matches the UGV's name, it publishes "the UGV is found" in the ROS environment. Subsequently, it calculates the UGV's position and orientation using the approach provided below.

2.4.2.2. Vector Calculation:. The QR code on the UGV is rectangular, with black squares at three corners and one empty corner. The algorithm developed for this thesis determines the UGV's orientation by finding the QR code's relative orientation to the UAV. It then calculates the UGV's absolute orientation to the global map frame (IFF). The first step involves calculating the vector for the QR code. The algorithm identifies the QR code's border and assigns numbers to the corners, beginning with the top left corner, labeled as x_0, y_0 in a clockwise direction to x_3, y_3 which are depicted in Figure 2.29 below. It then draws a vector with the components $vector_x$ and $vector_y$

which represent x and y coordinates respectively and calculated as

$$\text{vector_x} \quad = x_3 - x_0, , \tag{2.56}$$

$$\text{vector_y} \quad = y_3 - y_0. \tag{2.57}$$

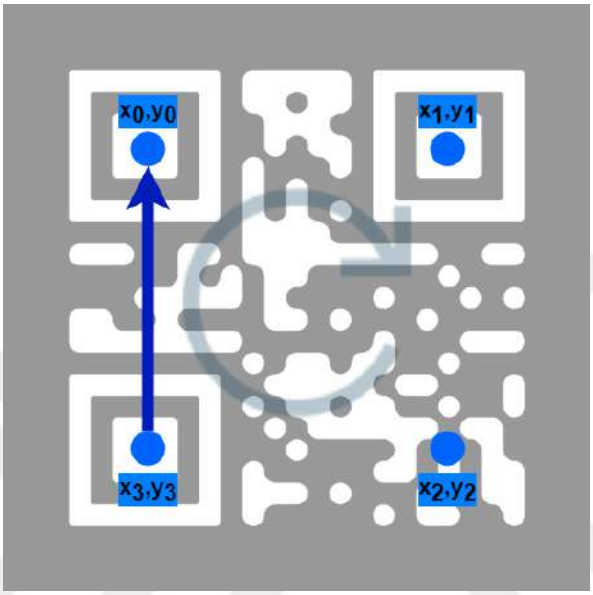


Figure 2.29. The illustration of the vector calculation.

2.4.2.3. Angle Calculation:. After determining the QR Code's coordinates using equations (2.56) and (2.57), a vector is drawn. Then the drawn vector's angle, which represents the slope, indicates the QR code's relative orientation to the UGV's camera frame. The vector's angle is calculated as

$$\text{QR's orientation-radians} = \arctan\left(\frac{\text{vector}_y}{\text{vector}_x}\right), \tag{2.58}$$

$$\text{QR's orientation-degree} = \text{QR's orientation-radians} \times \left(\frac{180}{\pi}\right). \tag{2.59}$$

2.4.2.4. Estimation of Localization:. To simplify calculations during QR code detection, the UAV pauses its trajectory and switches to "Hold Mode". This keeps the QR code at the center of the camera view, aligning the x and y positions of the UGV with the UAV's. The UGV's absolute orientation, concerning the global map frame, is then determined by subtracting the QR code's orientation from the UAV's orientation. The

equations for these calculations are provided as

$$\text{UGV's Position} = \Big(\text{UAV's position}_x,\ \ \text{UAV's position}_y,\ \ 0\Big), \tag{2.60}$$

$$\text{UGV's Orientation} = \Big(0,\ \ 0,\ \ \text{UAV's yaw} - \text{QR's orientation-radians}\Big). \tag{2.61}$$

After determining the position and orientation vectors for the UGV using equations (2.59), (2.58), (2.60), and (2.61) the site manager initiates a new localization process using these initial vectors.

2.4.3. Cooperative Mission 3: Ensuring Compliance with Safe Zone through Limited Movement of Robots at a Relative Distance

The third mission as illustrated in Figure 2.30 below aims to maintain robots within a pre-defined safe zone based on their relative locations.

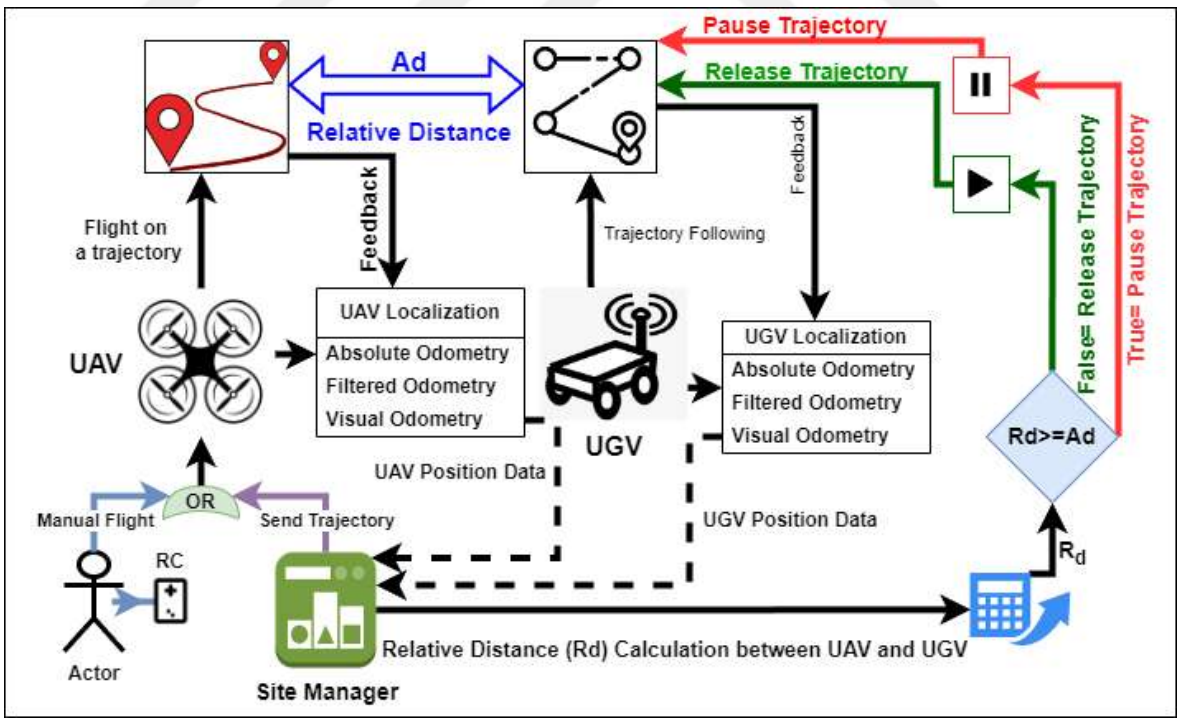


Figure 2.30. The illustration of the cooperative mission 3.

Each robot follows its trajectory, but the relative distance R_d between them is

kept within a set limit A_d, which can vary depending on factors like the UAV's battery level. Let UAV_x and UAV_y represent the UAV's position in IFF, while UGV_x and UGV_y represent the UGV's position in IFF as illustrated in Figure 2.31. Them R_d is calculated as

$$R_d = \sqrt{(UAV_x - UGV_x)^2 + (UAV_y - UGV_y)^2}. \tag{2.62}$$

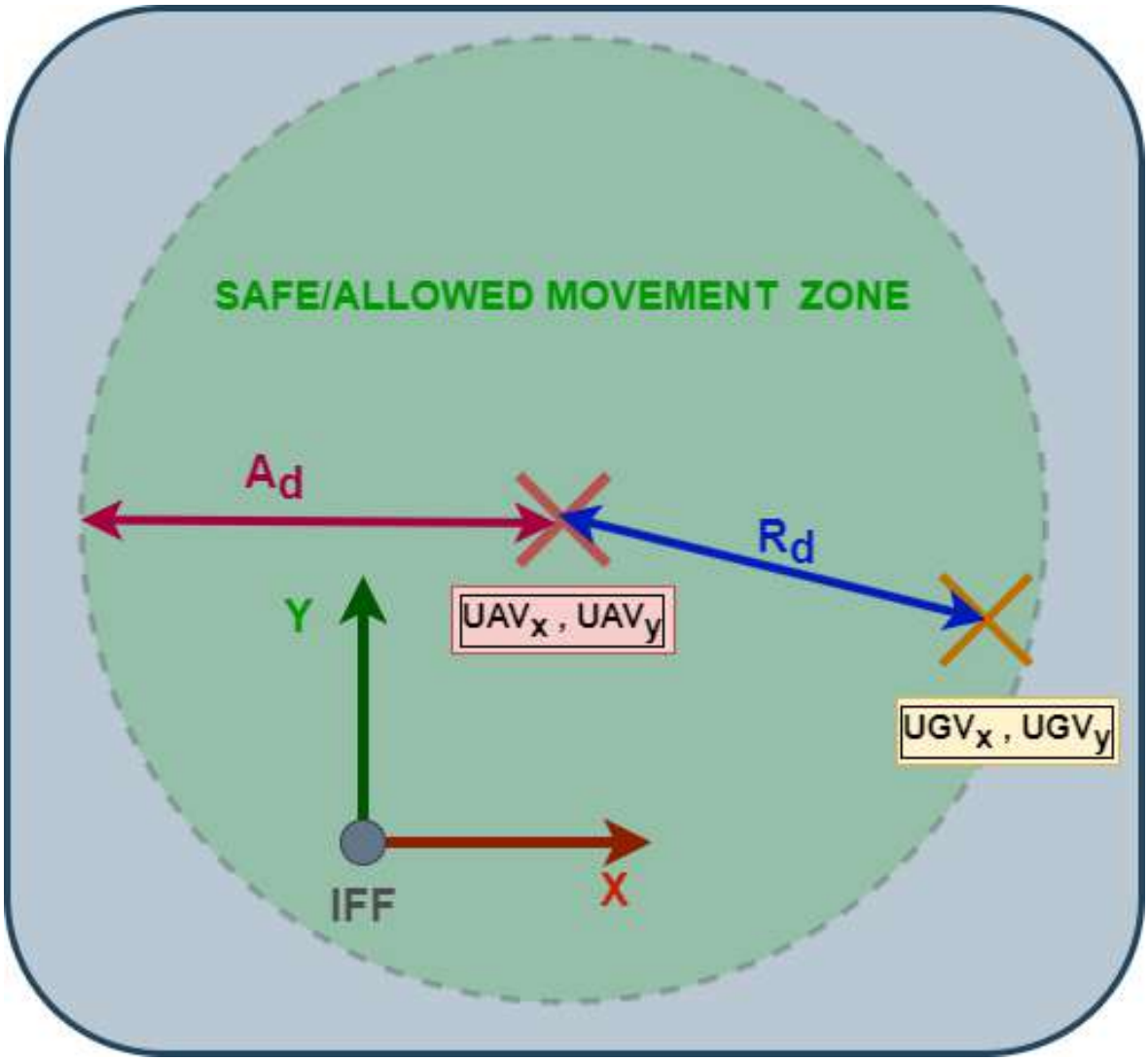


Figure 2.31. The illustration of calculation for R_d.

The UAV is capable of moving faster and navigating areas with fewer obstacles compared to the UGV. Therefore, the UGV halts while the UAV continues. However, the UAV's battery life is limited due to weight constraints and its dynamics. This

scenario is valuable for demonstrating how to keep aerial robots safe in heterogeneous applications. If the UAV's battery reaches a critical level or an emergency landing is required, the nearby UGV can provide a safe landing spot.

3. SIMULATION WORKS AND RESULTS

3.1. UTOPIA's Digital Twin

This chapter discusses the simulation aspects of the thesis. The purpose of these simulations is to fine-tune the code for real-world testing. First, each robot's localization is improved using its URDF model, the Gazebo simulation environment, Gazebo sensor plugins, and sensor fusion through the application of the EKF. Next, each robot's trajectory is enhanced using PID and MPC methods. Then, a global framework is established, and a digital twin of the UTOPIA project's environment is created, as shown in Figure 3.1. Using the improved algorithms, cooperative missions are conducted. The results of these missions are graphically represented. During the tests, ground truth is obtained from the simulation environment on Gazebo.

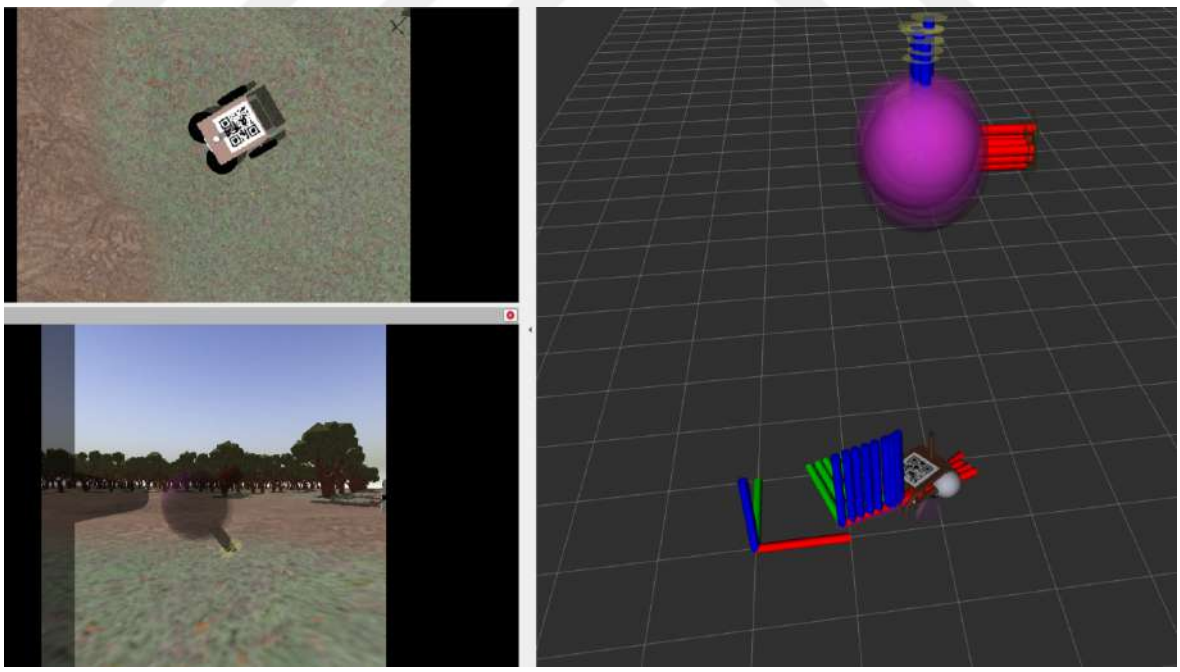


Figure 3.1. The digital twin of the UTOPIA project's environment.

The simulation test parameters for trajectories are given in Table 3.1. The trajectories are created in MATLAB and tracked using the Tracking_pid ROS package.

Table 3.1. Simulation test parameters for trajectories

Parameter Name	**Value**	**Unit**
Velocity	0.5	meter/second
Time step	0.05	second
Euclidian distance	0.1	meter
Radius of corners	1	meter

3.1.1. Gazebo Plugins

Gazebo plugins enable the simulation of the robot and its functions within a simulation environment. The plugin types that can be referenced through the robot's URDF model include model plugins, sensor plugins, and visual plugins [25, 26].

3.1.1.1. Gazebo Model Plugins. Gazebo model plugins are unique code segments that enhance the functionality of simulated models in the Gazebo environment. These plugins can control the model's behavior, simulate sensors, and interact with the simulation in multiple ways. The velocity controller is used for the UAV's rotors, and the differential drive controller is used for the UGV's wheels to simulate their movements.

3.1.1.2. Gazebo Sensor Plugins. In Gazebo, sensor plugins simulate data from various sensors often used in robotics. They generate realistic sensor data by mimicking the physical phenomena that actual sensors would measure.

(i) IMU Sensor Plugin: Simulates an IMU, providing data on linear acceleration and angular velocity, necessary for estimating robot's orientation and motion.

(ii) GPS Sensor Plugin: Emulates a GPS receiver, offering the robot's global coordinates, crucial for outdoor navigation.

(iii) Magnetometer Sensor Plugin: Simulates a magnetic field sensor helping in determining the robot's orientation relative to the Earth's magnetic field.

(iv) Mavros Odometry Sensor Plugin: Works with MAVROS package, supplying odometry data as estimated by the autopilot.

(v) Wheel Odometry Sensor Plugin: Simulates wheel encoders, providing data on wheel rotation to estimate the robot's position and velocity.

(vi) Laser Sensor Plugin: Emulates a laser rangefinder or LiDAR sensor, aiding in mapping surroundings, detecting obstacles, and navigation.

(vii) Camera Sensor Plugin:Mimics cameras, generating image data from the simulated environment, essential for vision-based applications.

3.2. Test Trajectories

The first trajectory used for simulation tests is a straight line measuring 5 meters. The second trajectory used in simulation tests is an L-shaped figure. The third trajectory used in simulation tests is an ellipse, which presents more difficulty for lateral and angular trajectory following and localization. The last trajectory used in simulation tests is an 8-shaped figure. This presents more challenges for lateral and angular trajectory tracking and localization compared to other figures. The details of the trajectories are shown in Figure 3.2.

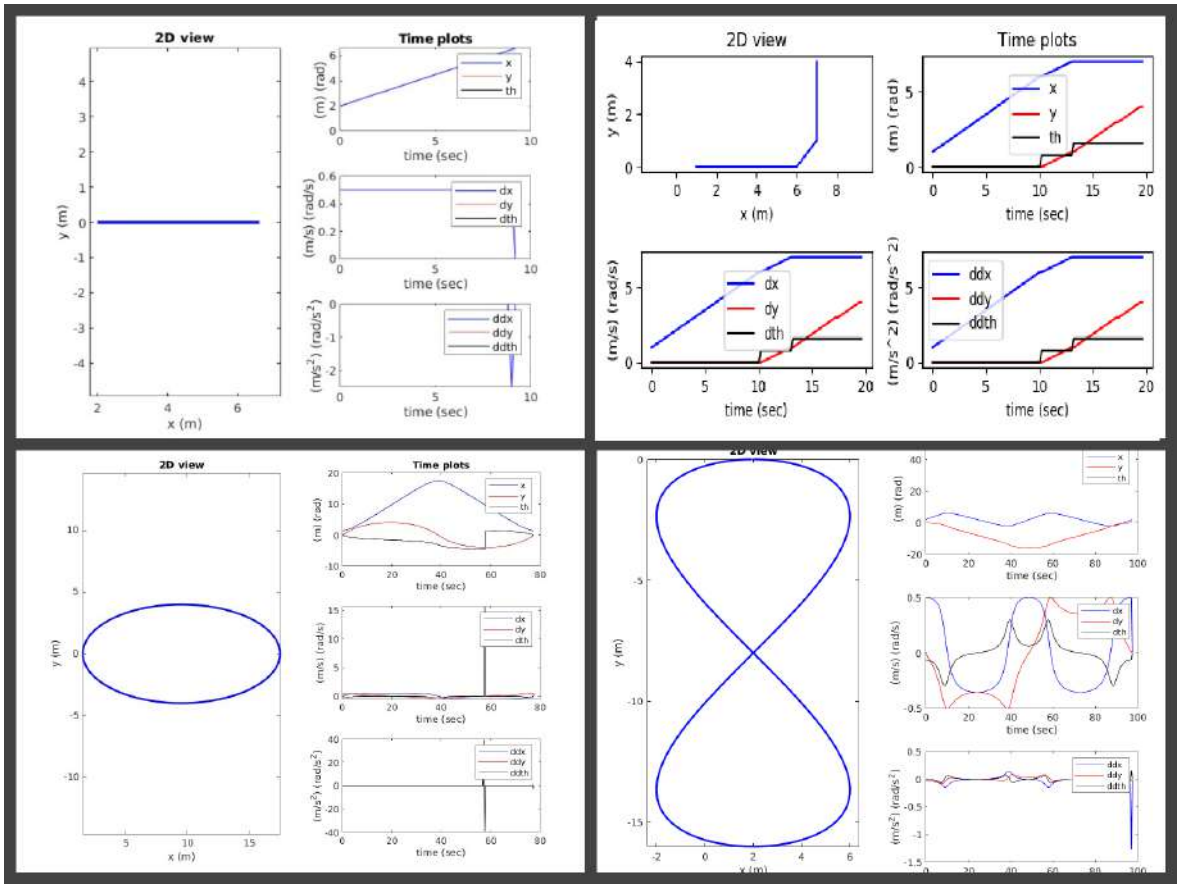


Figure 3.2. The detailed demonstration of the test trajectories.

The errors are calculated using the Root Mean Square Error (RMSE) equation given below, where X_i and Y_i are i_{th} position coordinates for the selected localization configuration, and X_{ref} and Y_{ref} are the reference position coordinates, and n is the sample number. The RMSE is calculated as

$$RMSE = \frac{1}{2}\left(\sqrt{\frac{\sum_{i=1}^{n}(X_i - X_{\text{ref}})^2}{n}} + \sqrt{\frac{\sum_{i=1}^{n}(Y_i - Y_{\text{ref}})^2}{n}}\right). \tag{3.1}$$

3.3. Simulation Works for The UAV

Improvements to control and localization of the UAV are performed in the Gazebo simulation environment, with data visualized in the Rviz interface. The enhancement of

localization is demonstrated through the trajectory following. Thus, control capability is also crucial to test. The tests range from a basic straight-line trajectory to more complex ones, like an 8-shaped figure as shown in Figure 3.3.

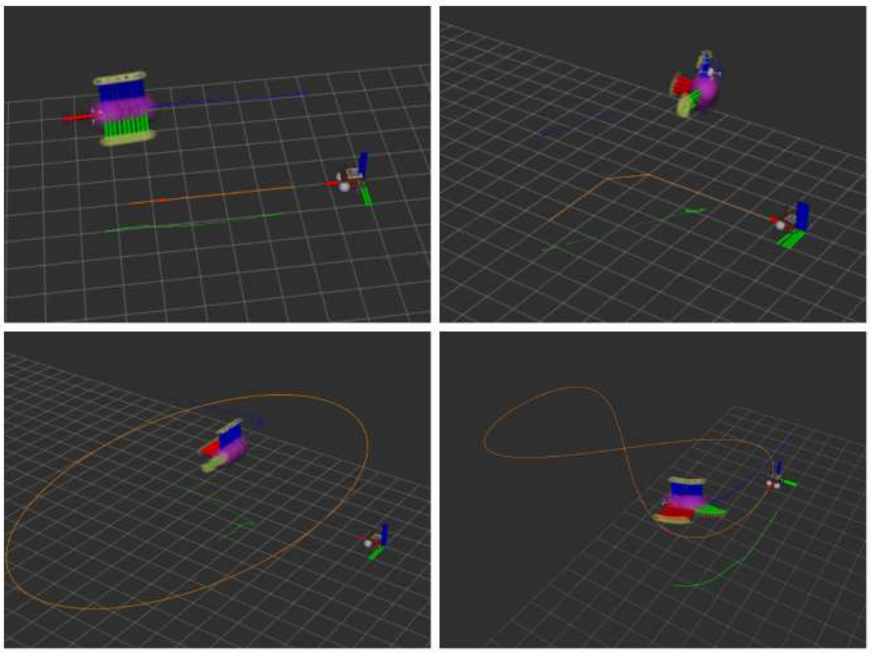


Figure 3.3. Simulation test scenes for the UAV.

The PID configuration parameters for the UAV are given in Table 3.2 below.

Table 3.2. PID parameters for the UAV on simulation

Parameter	**Longitudinal**	**Lateral**	**Angular**
K_p	1.0	-3	1.5
K_i	0.1	-0.008	0.1
K_d	0.2	-1.0	0.05

During tests, configurations demonstrating EKF and GPS correction impacts on localization are created and recorded, as detailed in Table 3.3. Designed to mimic real robot sensors, these configurations may differ in actual tests due to Gazebo sensor

variations.

Table 3.3. Configurations for localization-UAV

Number	**Configuration**
Config 1	Mavros Odometry-No EKF
Config 2	GPS, IMU and Magnetometer
Config 3	IMU, and Magnetometer
Config 4	GPS and IMU
Config 5	GPS and Magnetometer

The configurations recorded for localization, along with the reference trajectories (the straight line and the L-shaped figure), are depicted in Figure 3.4.

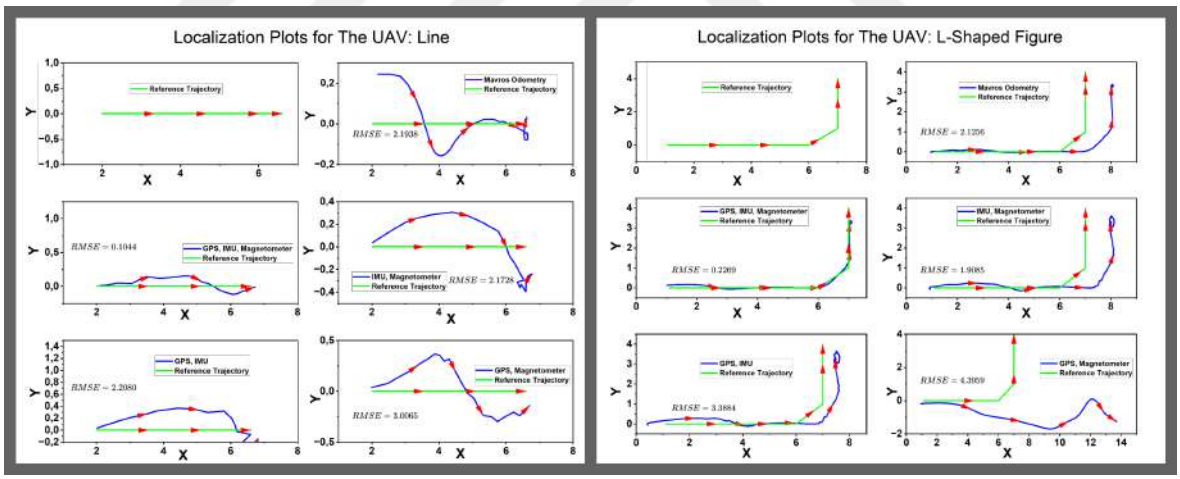


Figure 3.4. The plotting of recorded data for the straight line and the L-shaped trajectories of the UAV.

The configurations recorded for localization, along with the reference trajectories (the ellipse figure and the 8-shaped figure), are depicted in Figure 3.5.

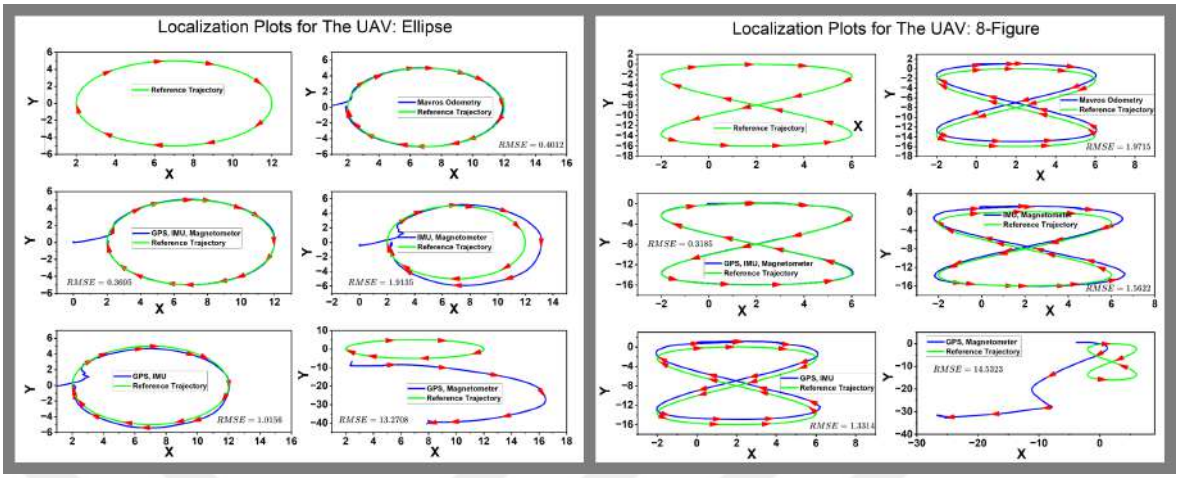


Figure 3.5. The plotting of recorded data for the ellipse and 8-figure trajectories of the UAV.

The RMSE values are calculated for each trajectory and localization configuration. The results are presented in Table 3.4. The results indicate that the second configuration, which includes GPS correction with the IMU, Mavros odometry, and magnetometer, produces the lowest RMSE value. The localization plots support this conclusion. On the other hand, the final configuration, which only utilizes GPS and the magnetometer, exhibits the worst performance.

Table 3.4. RMSE for the UAV's configurations and trajectories

Trajectory	GPS	IMU	Magnetometer	Mavros Odometry	RMSE
Line	-	-	-	X	2,1938
	X	**X**	**X**	**X**	**0,1044**
	-	X	X	-	2,1728
	-	X	X	X	2,2080
	X	-	X	-	3,0065
L-Shape	-	-	-	X	2,1256
	X	**X**	**X**	**X**	**0,2269**
	-	X	X	-	1,9085
	-	X	X	X	3,3884
	X	-	X	-	4,3959
Ellipse	-	-	-	X	0,4012
	X	**X**	**X**	**X**	**0,3695**
	-	X	X	-	1,9135
	-	X	X	X	1,0156
	X	-	X	-	13,2708
8-Figure	-	-	-	X	1,9715
	X	**X**	**X**	**X**	**0,3185**
	-	X	X	-	1,5622
	-	X	X	X	1,3314
	X	-	X	-	14,5323

3.4. Simulation Works for The UGV

The tests for the UGV on simulation are performed using the same trajectories as the UAV's tests. Trajectory parameters are the same as given in 3.1. The PID configuration parameters for the UGV are given in Table 3.5 below, and simulation test scenes for the UGV are shown in Figure 3.6.

Table 3.5. PID parameters for the UGV on simulation

Parameter	**Longitudinal**	**Lateral**	**Angular**
K_p	0.3	-0.8	-0.05
K_i	0.01	-0.001	-0.001
K_d	0.2	-0.01	-0.01

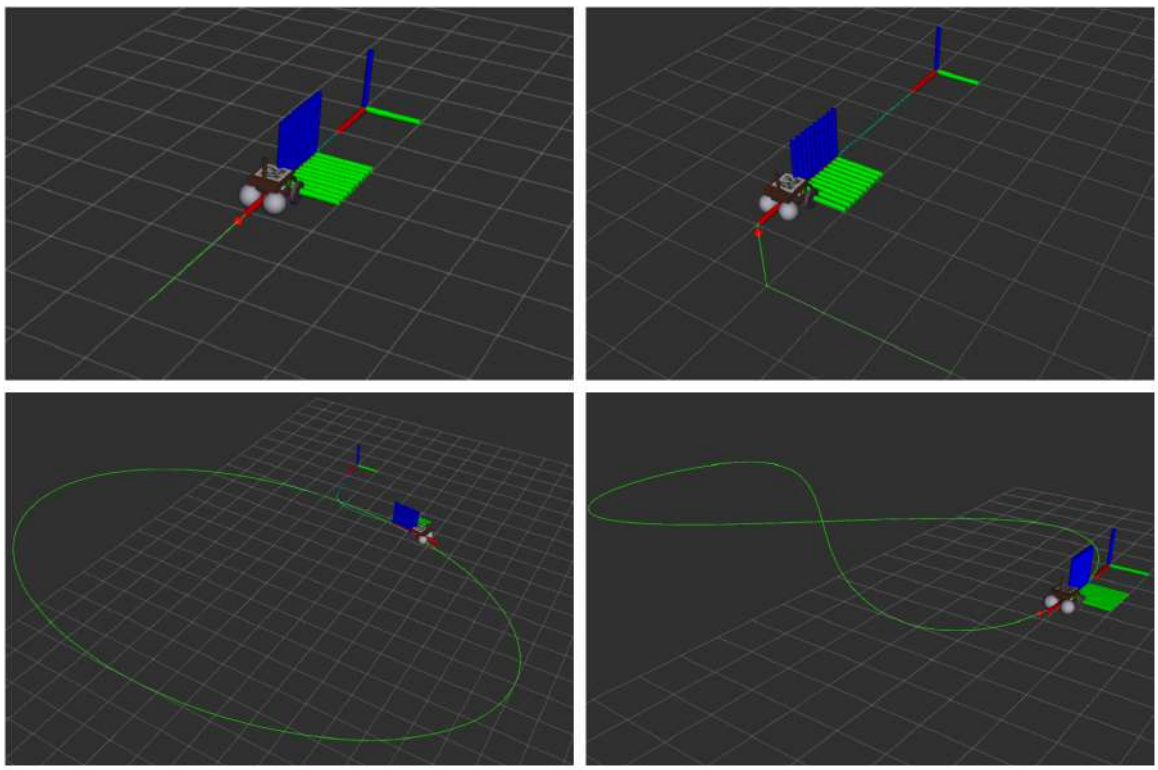


Figure 3.6. Simulation test scenes for the UGV.

The localization configurations for the UGV differ slightly from those of the UAV. In the UGV's EKF-applied localization configurations, the wheel odometry plugin is used as the odometry source instead of the MAVROS plugin as given in Table 3.6.

Table 3.6. Configurations for localization-UGV

Number	**Configuration**
Config 1	Wheel Odometry- No EKF
Config 2	Wheel Odometry, IMU, and Magnetometer
Config 3	GPS, Wheel Odometry, IMU, and Magnetometer
Config 4	GPS and IMU
Config 5	GPS and Magnetometer

The configurations recorded for localization, along with the reference trajectories (the straight line and the L-shaped figure), are depicted in Figure 3.7.

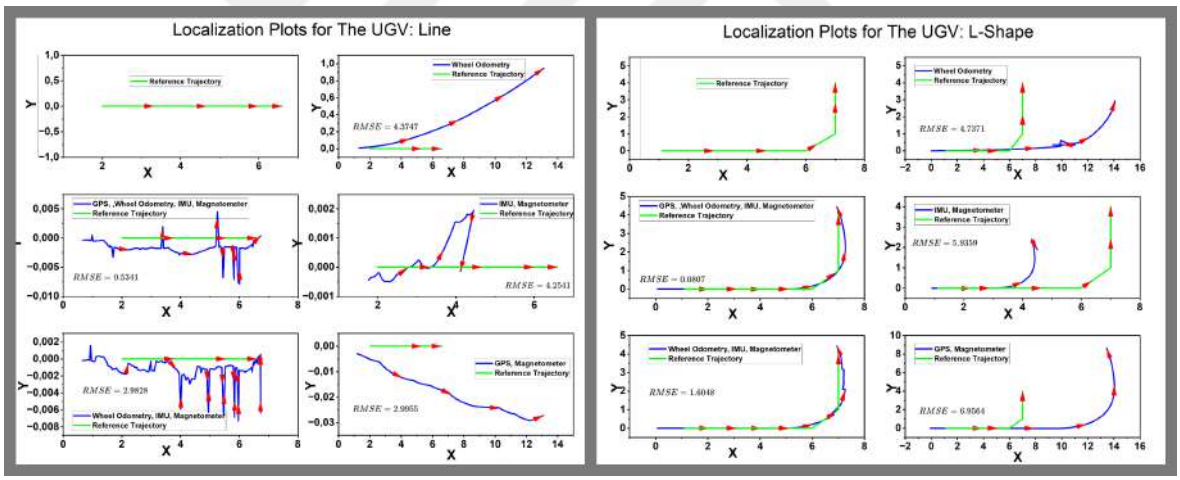


Figure 3.7. The plotting of recorded data for the straight line and the L-shaped trajectories of the UGV.

The configurations recorded for localization, along with the reference trajectories (the ellipse figure and the 8-shaped figure), are depicted in Figure 3.8

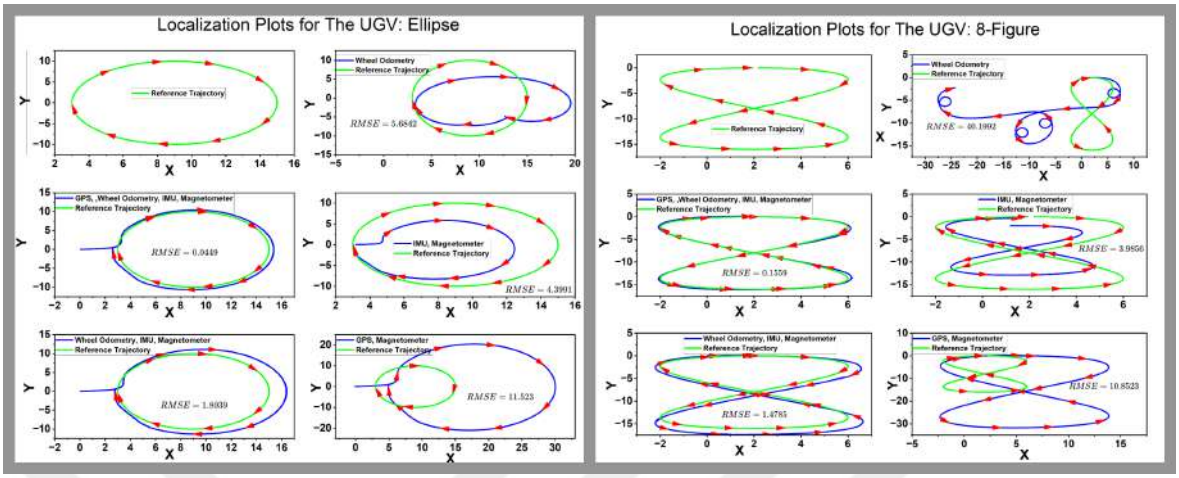


Figure 3.8. The plotting of recorded data for the ellipse and the 8-figure trajectories of the UGV.

The RMSE values are calculated for each trajectory and localization configuration. The results are presented in Table 3.7. The results indicate that the second configuration, which includes GPS correction with the IMU, wheel odometry, and magnetometer, produces the lowest RMSE value. The localization plots support this conclusion. On the other hand, the final configuration, which only utilizes GPS and the magnetometer, exhibits the worst performance.

Table 3.7. RMSE for the UGV's configurations and trajectories

Trajectory	GPS	IMU	Magnetometer	Wheel Odometry	RMSE
Line	-	-	-	X	4,3747
	X	**X**	**X**	**X**	**0,5341**
	-	X	X	-	4,2541
	-	X	X	X	2,9828
	X	-	X	-	2,9955
L-Shape	-	-	-	X	4,7371
	X	**X**	**X**	**X**	**0,0807**
	-	X	X	-	5,9359
	-	X	X	X	1,6048
	X	-	X	-	6,9564
Ellipse	-	-	-	X	5,6842
	X	**X**	**X**	**X**	**0,0449**
	-	X	X	-	4,3991
	-	X	X	X	1,8039
	X	-	X	-	11,523
8-Figure	-	-	-	X	5,6655
	X	**X**	**X**	**X**	**0,1559**
	-	X	X	-	3,9856
	-	X	X	X	1,4785
	X	-	X	-	10,8523

3.5. Cooperative Localization and Control-Mission 1

UTOPIA's Cooperative Mission 1 is depicted in Figure 2.27. In the UTOPIA system's digital twin, the UAV is operated and waypoints are recorded, as shown in Figure 3.9. These waypoints are then used to create the trajectories, illustrated in Figure 3.10 below. The UGV subsequently follows these trajectories. The resulting plots are provided in Figure 3.11. Table 3.8 presents the RMSE values for each localization

configuration.

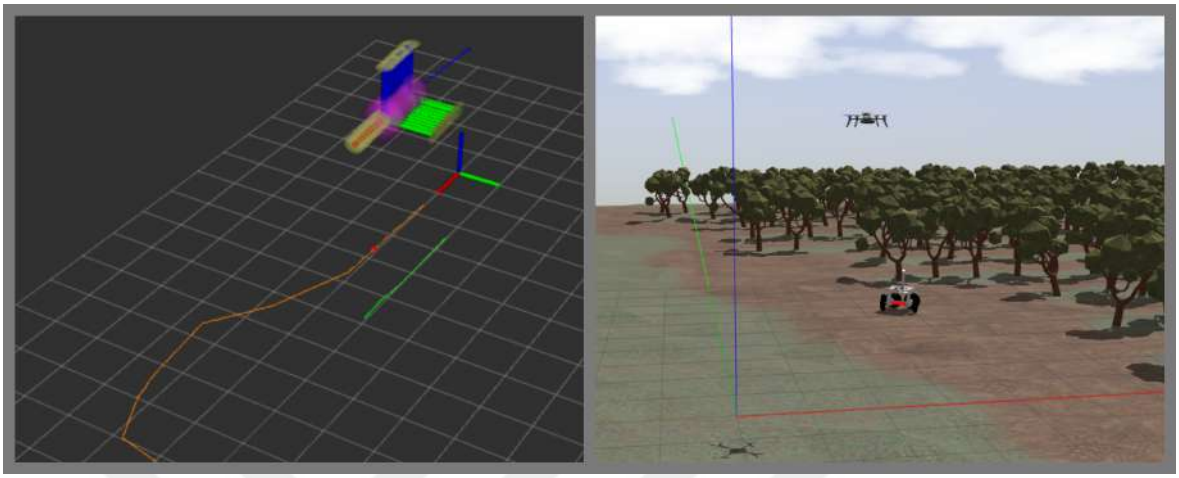


Figure 3.9. The scenes of performing cooperative mission 1 on UTOPIA's digital twin.

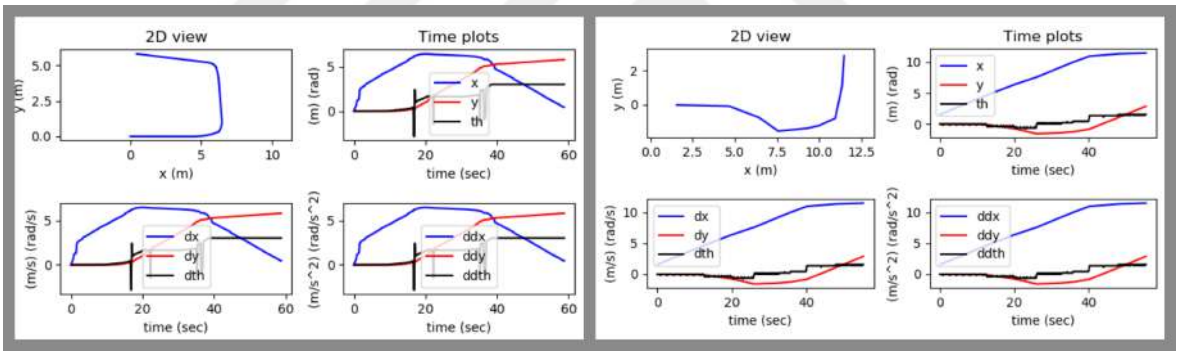


Figure 3.10. The trajectory of cooperative mission 1 on UTOPIA's digital twin.

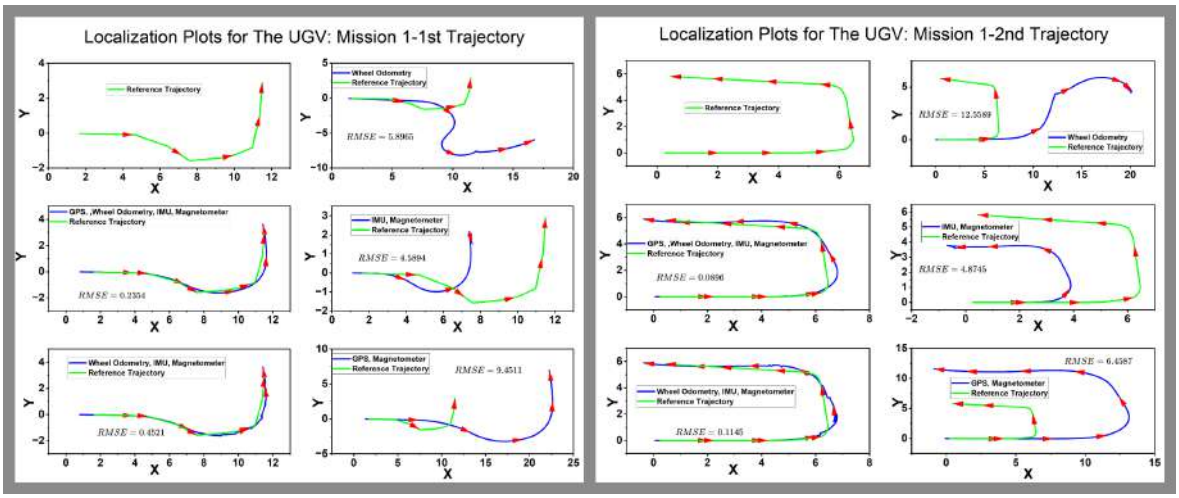


Figure 3.11. Localization plots of cooperative mission 1 on UTOPIA's digital twin.

Table 3.8. RMSE values for cooperative mission 1 on simulation tests

Trajectory	**GPS**	**IMU**	**Magnetometer**	**Wheel Odometry**	**RMSE**
1	-	-	-	X	5,8965
	X	**X**	**X**	**X**	**0,2354**
	-	X	X	-	4,5894
	-	X	X	X	0,4521
	X	-	X	-	9,4511
2	-	-	-	X	12,5589
	X	**X**	**X**	**X**	**0,0896**
	-	X	X	-	4,8745
	-	X	X	X	0,1145
	X	-	X	-	6,4587

3.6. Cooperative Localization and Control-Mission 2

UTOPIA's Cooperative Mission 2 is illustrated in Figure 2.28. In the UTOPIA system's digital twin, GPS and wheel odometry plugins are aborted to simulate a lost UGV due to high covariance in its localization. Then, the UAV follows its trajectory to scan the field. When it detects the UGV's QR code, the site manager calculates the UGV's localization using the UAV's localization and the orientation and position of the QR code, as depicted in Figure 3.12. Then, the UGV's estimated localization is launched with much lower covariance, as shown in Figure 3.13.

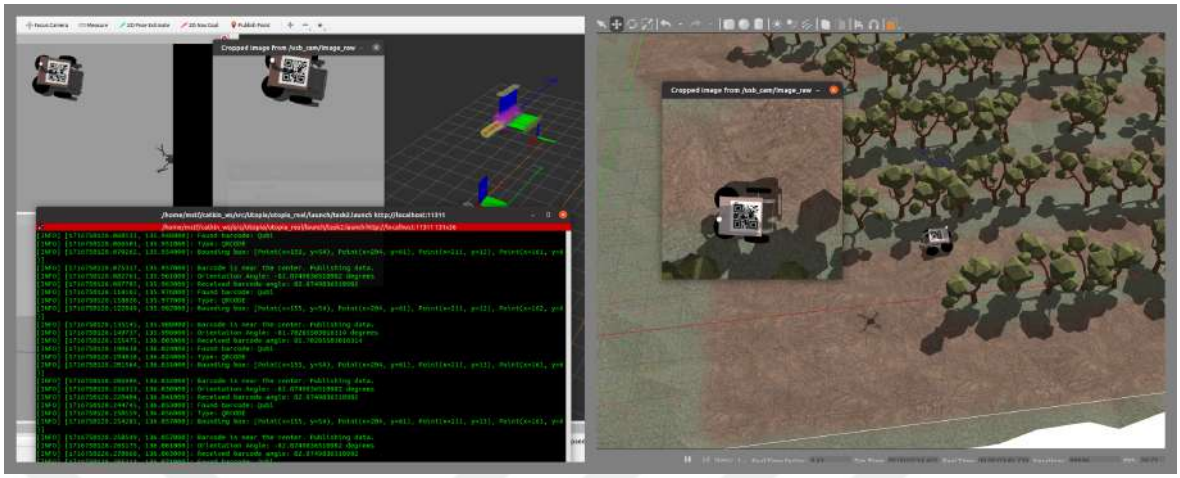


Figure 3.12. The plotting of recorded data for the straight line trajectory of the UGV.

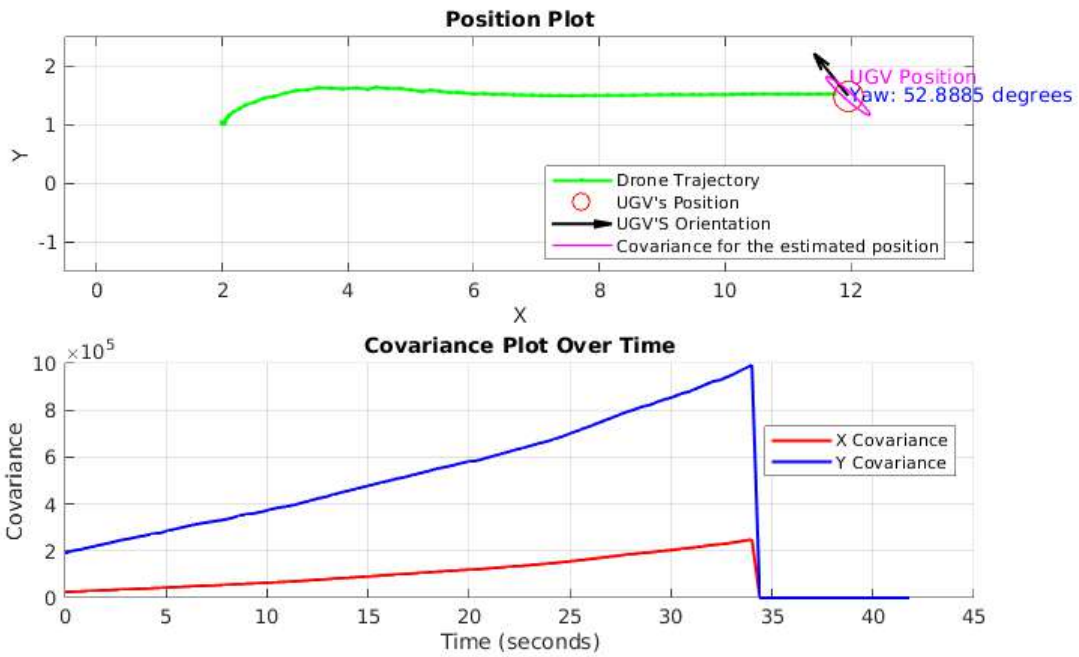


Figure 3.13. The plotting of recorded data for the straight line trajectory of the UGV.

3.7. Cooperative Localization and Control-Mission 3

The UAV is set to stand on hover, and a trajectory is sent to the UGV. The allowed distance is set to 0.5 m, 2m, and 5m respectively with random duration. The UGV is paused its trajectory when the relative distance is reached to the allowed

distance. Figure 3.13 demonstrates the change in the allowed distance and the position record of the UGV.

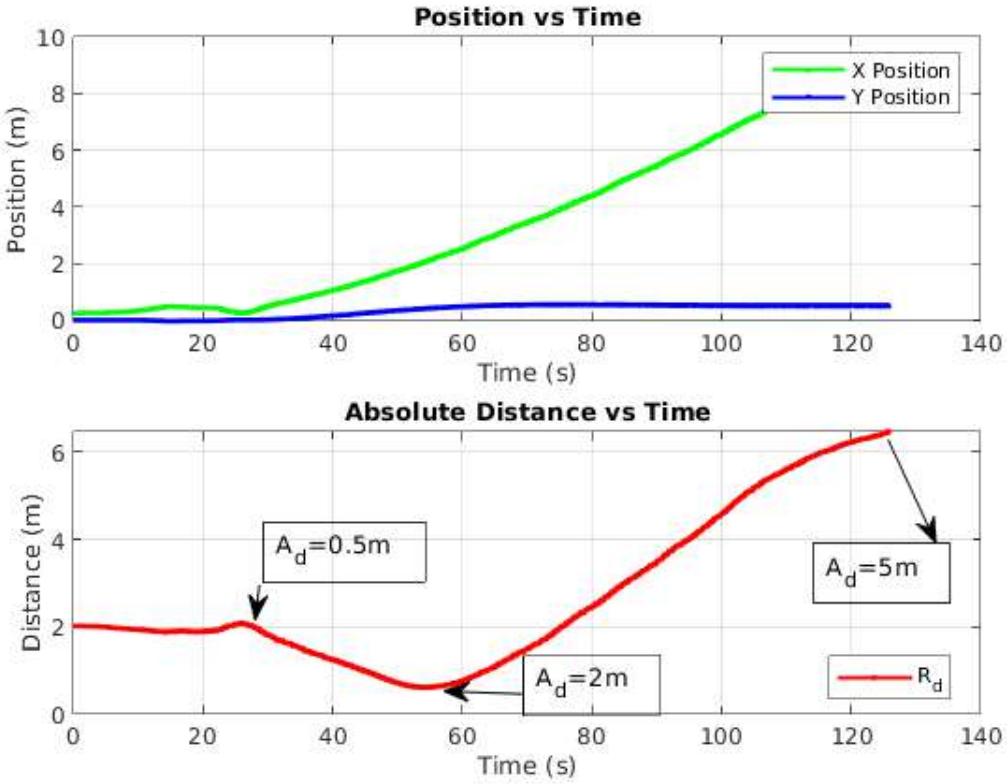


Figure 3.14. The plotting of recorded data for the straight line trajectory of the UGV.

4. EXPERIMENTAL WORKS AND RESULTS

This chapter presents the preliminary work necessary for conducting experiments as well as the results. After the simulation configurations are completed and tests are conducted, improved algorithms should be implemented in the robots for experimental testing. Therefore, some initial groundwork is required.

4.1. Preliminary Works for Experimental Tests

4.1.1. Setup for The UTOPIA's Global Network

Once the calibrations for both the UAV and the UGV are complete, a global network is established to fulfill the cooperative missions. This network is created using a wireless modem with an access point. Subsequently, the Emlid GPS modules, including both base and rovers, are configured for this network. Figure 4.1 shows the UTOPIA's global network components.

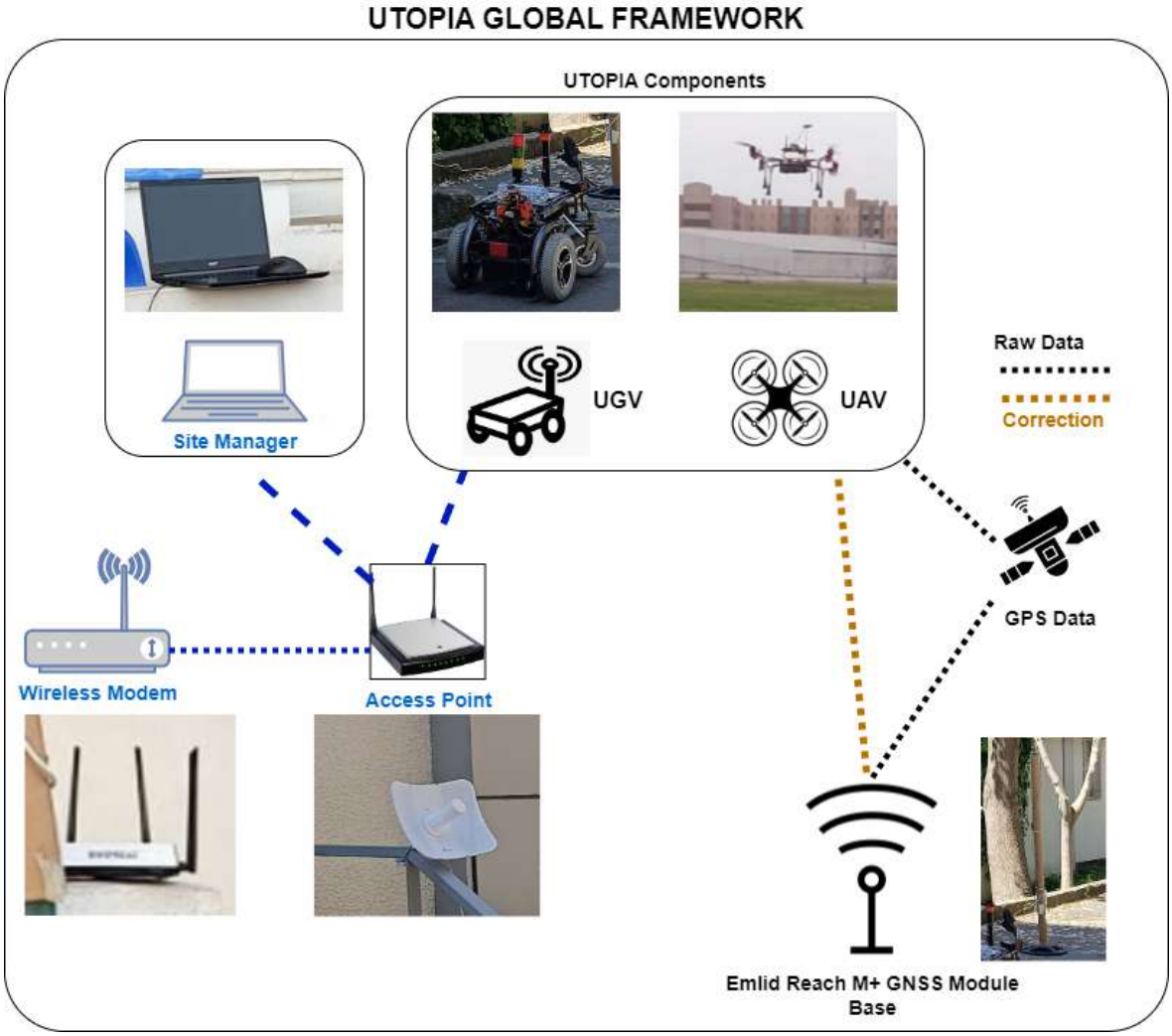


Figure 4.1. UTOPIA's global network.

Table 4.1 provides the configurations for UTOPIA components' localization. These configurations are created after the calibrations described in the following sections.

Table 4.1. Configurations for localization for UTOPIA components

Number	**Configuration**
Config 1	Visual Odometry- No EKF
Config 2	GPS, Visual Odometry, IMU, and Pose
Config 3	GPS, IMU, and Pose
Config 4	GPS and IMU
Config 5	Visual Odometry, IMU, and Pose

The ground truth is obtained from Gazebo during simulation tests. However, it is more challenging in experimental work. Reference trajectories are marked on the ground using signs after measurements. In cooperative missions, key positions of the trajectories are marked or referenced for verification. It is important to note that this ground truth method is valid and reliable only for pre-defined reference trajectories. For more precise ground truth, advanced techniques are necessary to achieve accurate localization. Here, the aim is to get the RMSE error for trajectory following, not localization.

4.1.2. SITL Tests of The UAV

Software In The Loop (SITL) tests for the UAV are conducted on Gazebo using PX4 Autopilot software packages. This setup simulates tasks and facilitates configurations. The PX4 Autopilot software and MAVROS packages enable communication with the QGround Control interface during these simulated tests. Consequently, it allows for the simulation of real tests and configuration adjustments before actual testing, thus reducing risk.

4.1.3. HITL Tests of The UAV

Hardware-in-the-loop (HITL) tests are conducted using MAVROS, PX4 Autopilot, Gazebo, and the Qground Control interface to command the UAV's propellers over the Pixhawk board during simulation tests. HITL tests provide an opportunity to observe and verify configurations for real flight. A screenshot from the HITL test's position track is provided in Figure 4.2.



Figure 4.2. The position track record of the HITL test.

4.1.4. Calibration of The UAV

After completing the SITL and HITL configurations and tests, the UAV is tested for manual flight. Configurations for the RC transmitter and flight modes are also made. The necessary flight modes are assigned to the RC transmitter and the actuator settings are shown in Figure 4.3 below.

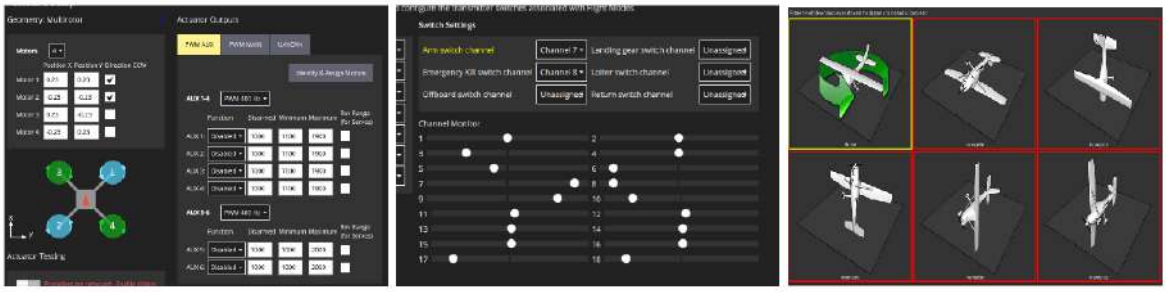


Figure 4.3. The UAV's actuator settings are on the left, and the RC settings are on the middle, and the sensor calibrations are on the right.

Several flight modes are available in the QGround Control software. However, only a select few are used for this thesis. These include:

- Hold mode: The UAV maintains its current altitude and hovers.
- Position mode: The UAV uses GPS corrections to maintain its position.
- Stabilized Mode: In this mode, the UAV attempts to correct its location using

the internal IMU fusion function from the Pixhawk board.

- Altitude mode: The UAV only maintains its altitude, disregarding its position.
- Land: The UAV lands autonomously.
- Arm: The UAV's motors start rotating.
- Kill: This mode stops the motors immediately.
- Takeoff: The UAV takes off autonomously.

The test flights are performed as shown in Figure 4.4 and the camera and image processing algorithm tests' scenes are shown in Figure 4.5 below.



Figure 4.4. The scenes from the UAV's calibration test flights.

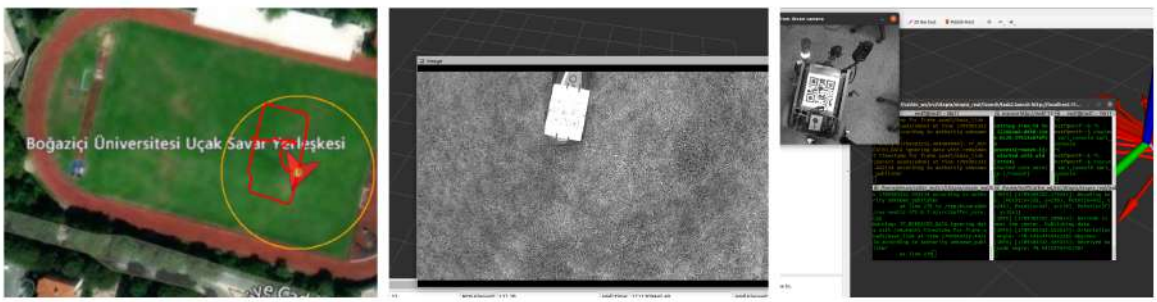


Figure 4.5. The scenes from the UAV's camera and algorithm tests.

4.1.5. Calibration of The UGV

Before conducting cooperative mission experiments, enhancements are made to the UGV's indoor and outdoor localization, and its autonomous trajectory-following capabilities are tested. Initially, the UGV's PID parameters and the power output of the motor driver are configured through testing. The Roboteq motor controller is set up as an open loop internally since the motors lack internal sensors for position tracking. However, the trajectory following occurs in a closed loop, which necessitates configuring the power output of the motor controller through testing. The PID parameters of the UGV for the experimental tests are given in Table 4.2. Figure 4.6 shows the calibration scenes of the UGV.

Table 4.2. PID parameters for the UGV on experimental tests

Parameter	**Longitudinal**	**Lateral**	**Angular**
K_p	0.4	-0.05	-0.2
K_i	0.01	-0.01	-0.01
K_d	0.05	-0.01	-0.08

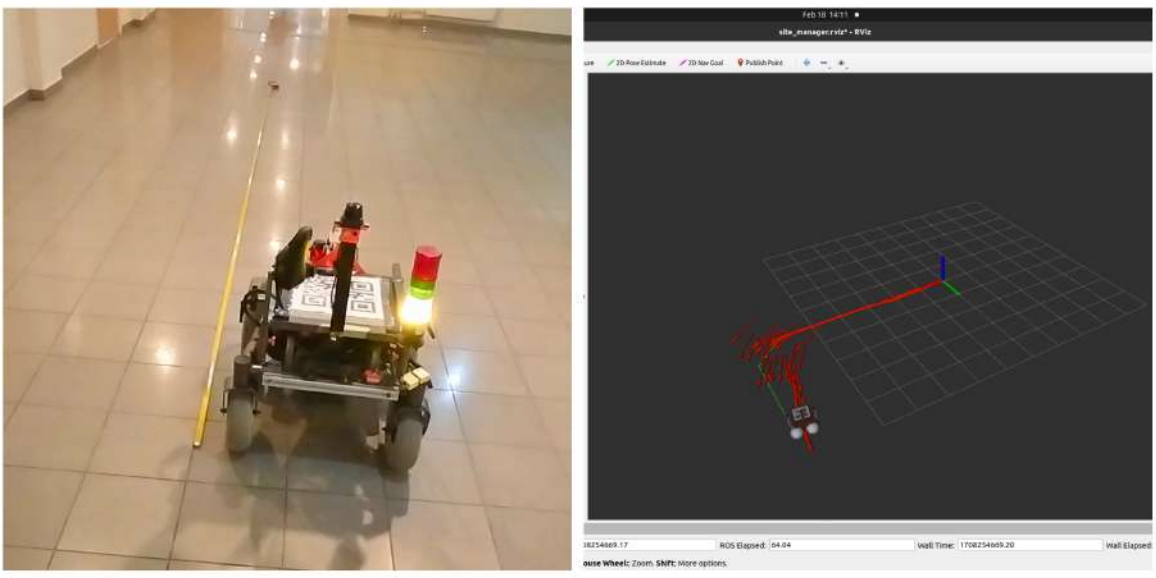


Figure 4.6. The scenes from the UGV's indoor calibration.

4.1.6. Indoor Tests of The UGV

The UGV is initially tested indoors to enhance its localization. Indoor odometry is simpler compared to outdoor, as it has fewer disturbances and sensor noises. Indoor localization doesn't include GPS correction. Any calibrations and improvements are first implemented indoors. Figure 4.7 shows the scenes from the indoor tests for the UGV.



Figure 4.7. The scenes from the UGV's indoor tests.

In indoor calibrations, GPS was not used. Only IMU, Visual Odometry, and Pose data are used in indoor environments from the UGV's sensor set. The configurations based on these data are given in Table 4.3.

Table 4.3. Configurations for localization of the UGV for indoor tests

Number	**Configuration**
Config 1	Visual Odometry and IMU
Config 2	Visual Odometry, IMU, and Pose
Config 3	IMU
Config 4	IMU and Pose
Config 5	Visual Odometry - No EKF

The line and L-shaped trajectories are used for indoor configurations. Once calibration is complete, the calibrated test data is recorded and plotted, as shown in Figure 4.8 below. The calculated RMSE values are in Table 4.4.

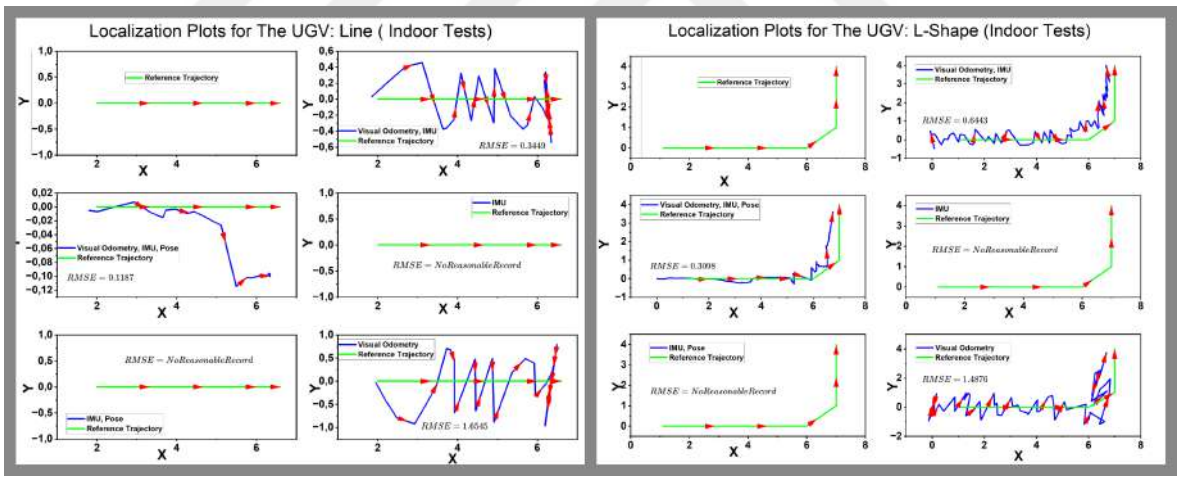


Figure 4.8. The plots from the UGV's indoor tests.

Table 4.4. RMSE values for indoor calibration tests of the UGV

Trajectory	GPS	IMU	Pose	Visual Odometry	RMSE
Line	-	-	-	X	0.3449
	X	**X**	**X**	**X**	**0.1187**
	X	X	X	-	**NONE**
	X	X	**-**	**-**	**NONE**
	-	X	X	X	1.6545
L-Shape	-	-	-	X	0.6443
	X	**X**	**X**	**X**	**0.3098**
	X	X	X	-	**NONE**
	X	X	**-**	**-**	**NONE**
	-	X	X	X	1.4876

During the calibration tests, it was observed that IMU data alone or IMU data with Pose did not provide reasonable results. However, combining visual odometry and IMU yielded better results compared to using only visual odometry. The best result was achieved when all sensors were used as sources in the EKF.

4.1.7. Outdoor Tests of The UGV

After completing indoor tests for the UGV, the system proceeds to outdoor testing during both day and night. First, the sensors are configured for outdoor localization and tested without GPS. Then, GPS is incorporated. Figure 4.9 shows the scenes from the outdoor tests for the UGV. During the tests, the configurations for localization are created based on the indoor calibration results as given in 4.3 and 4.8. These configurations are listed in Table 4.5.

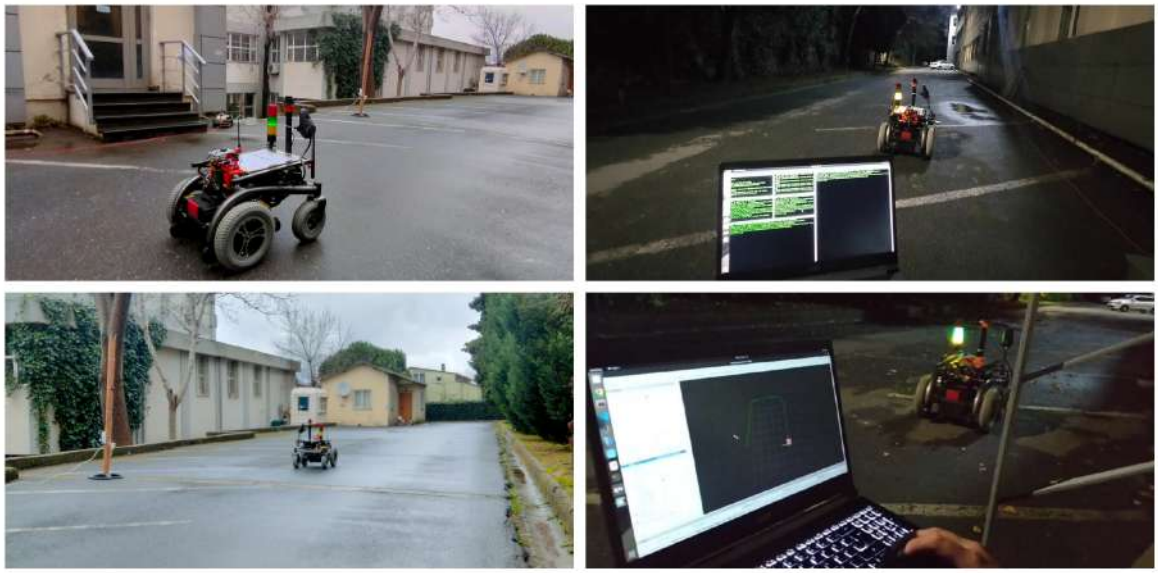


Figure 4.9. The scenes from the UGV's outdoor tests.

Table 4.5. Localization configurations of the UGV on outdoor tests

Number	**Configuration**
Config 1	Visual Odometry- No EKF
Config 2	GPS, Visual Odometry, IMU, and Pose
Config 3	GPS, IMU
Config 4	GPS, IMU, and Pose
Config 5	Visual Odometry, IMU, and Pose

The outdoor tests are conducted using the configurations in Table 4.5. Once the calibrations are done, the line and L-shaped trajectories are tested. The results are plotted as shown in Figure 4.10, and the calculated RMSE errors are shown in Table 4.6.

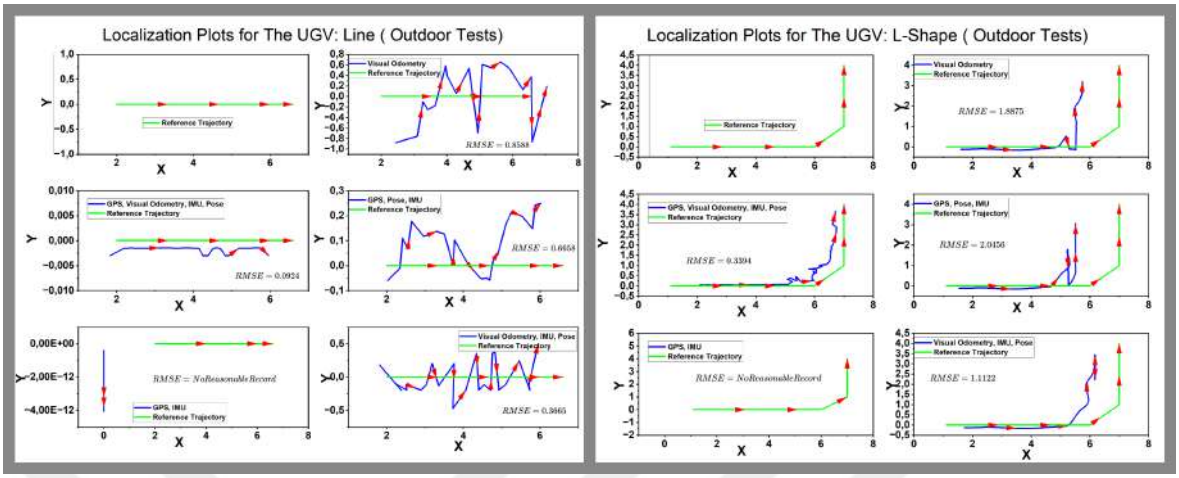


Figure 4.10. The plots from the UGV's outdoor tests.

Table 4.6. RMSE values for outdoor calibration tests of the UGV

Trajectory	**GPS**	**IMU**	**Pose**	**Visual Odometry**	**RMSE**
Line	-	-	-	X	0.8588
	X	**X**	**X**	**X**	**0.0924**
	X	X	X	-	0.6658
	X	X	**-**	**-**	**NONE**
	-	X	X	X	0.3665
L-Shape	-	-	-	X	1.8875
	X	**X**	**X**	**X**	**0.3394**
	X	X	X	-	2.0456
	X	X	**-**	**-**	**NONE**
	-	X	X	X	1.1122

During calibrations, it was found that the IMU alone does not provide reasonable results. Visual odometry performs better indoors but becomes unstable outdoors due to light intensity, moving objects, and unexpected disturbances. Indoor environments are more stable, so visual odometry works better there. Optical flow is simpler and requires less computational power. Combining visual odometry with the IMU yields better results after the EKF. Adding Pose data to this combination gives more reason-

able results. Outdoors, GPS significantly corrects this configuration. Based on these findings, the configurations for localizing the UTOPIA components are created as given in Table 4.1.

4.2. Cooperative Localization and Control-Mission 1

The first mission is performed similarly to the simulation test. The vineyard use-case of UTOPIA is shown in Figure 4.11. The UAV is flown and waypoints are manually selected. Once the waypoint selection is complete, the site manager creates the trajectories depicted in Figure 4.12. The UGV then follows these trajectories, and localization configurations are recorded. These records are illustrated in Figure 4.13. The RMSE values are calculated and presented in Table 4.7.



Figure 4.11. The vineyard use-case of UTOPIA.

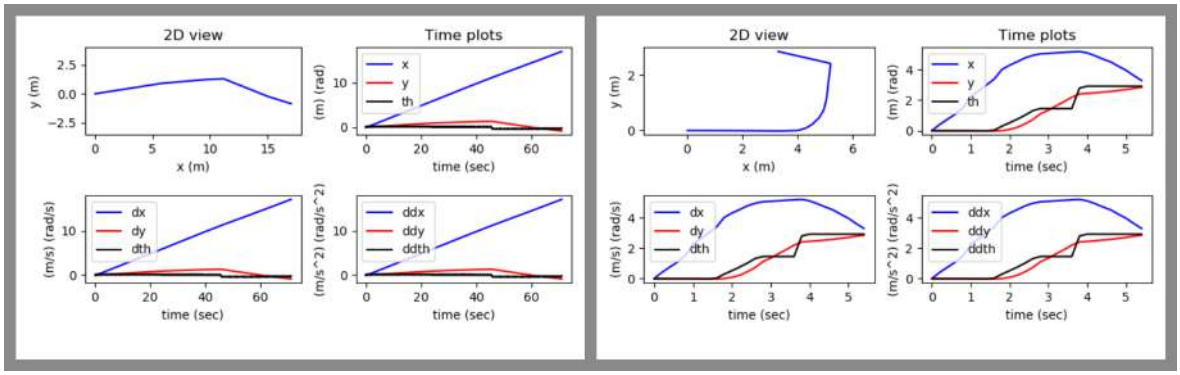


Figure 4.12. The records for cooperative mission 1.

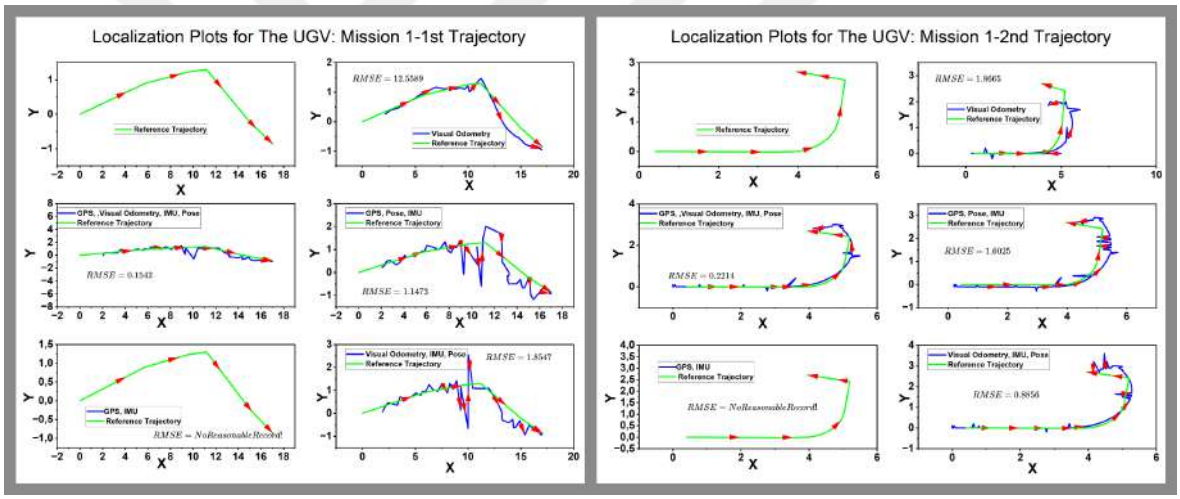


Figure 4.13. The records for cooperative mission 1.

Table 4.7. RMSE-Cooperative mission 1 on experimental tests

Trajectory	**GPS**	**IMU**	**Pose**	**Visual Odometry**	**RMSE**
1	-	-	-	X	12,5589
	X	**X**	**X**	**X**	**0,1542**
	X	X	X	-	1,1473
	X	X	**-**	**-**	**NONE**
	-	X	X	X	1,8547
2	-	-	-	X	1,8665
	X	**X**	**X**	**X**	**0,2214**
	X	X	X	-	1,0025
	X	X	**-**	**-**	**NONE**
	-	X	X	X	0,8856

The results presented in Table 4.7 above indicate that the second configuration, which includes all sensors, performs best, yielding the lowest RMSE value. In contrast, the fourth configuration, using only GPS and IMU data, performs the worst, as it does not output reasonable data.

4.3. Cooperative Localization and Control-Mission 2

The second mission is executed similarly to the simulation tests. The GPS and camera of the UGV are unplugged to increase covariance for localization. The UAV is then flown, and the UGV's QR code is scanned using the UAV's camera. Once the UGV is detected, the site manager computes the UGV's position and orientation using the relative localization data and camera view. This results in the UGV being handled with significantly reduced covariance on localization. Two experimental tests are conducted. Figure 4.14 displays the scene view of the tests on the site manager screen. Figure 4.15 and Figure 4.16 show the UAV's trajectory and the UGV's covariance plots for test 1 and test 2 respectively. The position and orientation of the UGV are marked upon its location.

The estimated localization and the recorded covariance information for the UGV are provided in Table 4.8. The results show that the UGV has a high covariance on its localization until it is detected by the UAV and its relocalization node is initiated by the site manager. After launching the relocalization node using the relative localization between the UAV and UGV, along with the QR calculation results, the UGV's new localization has a reasonable covariance and more reliable localization.

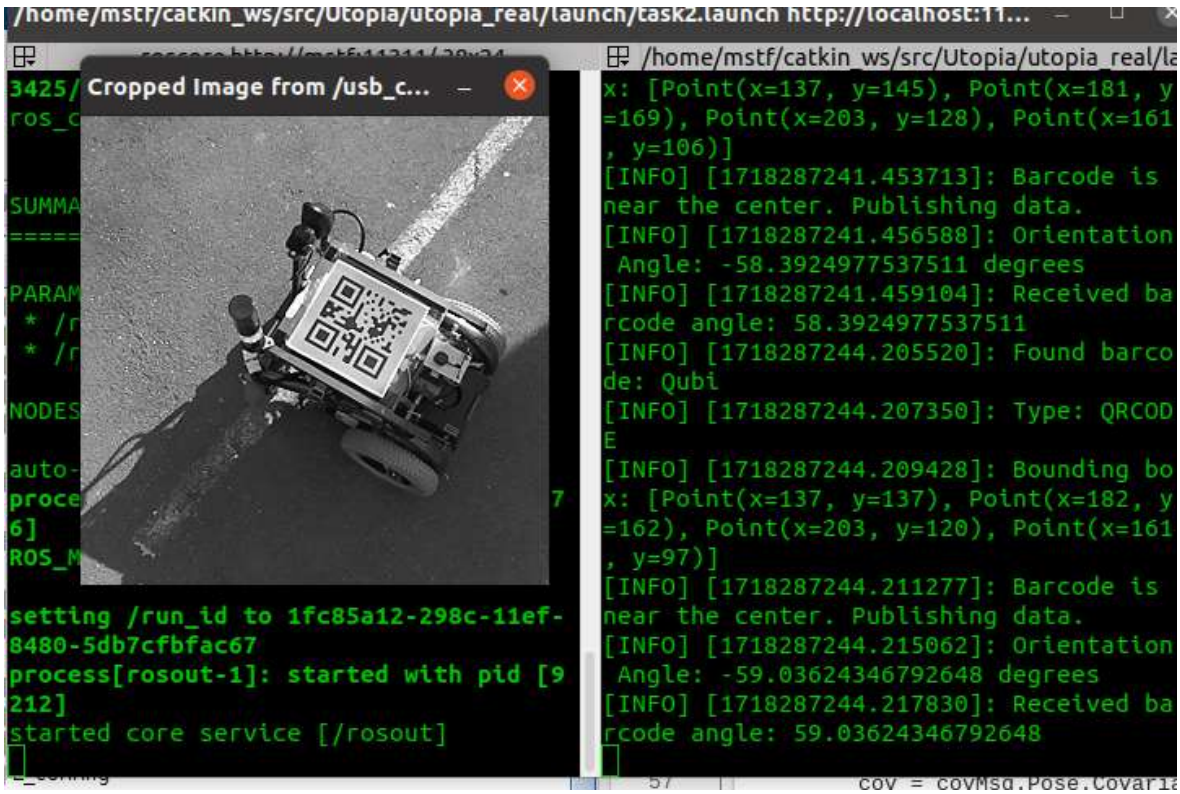


Figure 4.14. The scene view of the tests for cooperative mission 2 on the site manager screen.

Table 4.8. Estimated UGV localization and recorded covariance

Test	Estimated Position		Estimated Orientation	
	X	**Y**	**Yaw (Rad.)**	**Angle (Deg.)**
1	4,70970479	-0,5258043	1,05723771	60,5752589
2	8,81632224	0,01362048	0,40375257	23,1333182
Test	**Covariance Before Correction**		**Covariance After Correction**	
	X	**Y**	**X**	**Y**
1	**1282,77629**	3,62E-16	**0,00299949**	-6,45E-13
2	**718,536498**	1,93E-17	**0,0044209**	2,43E-05

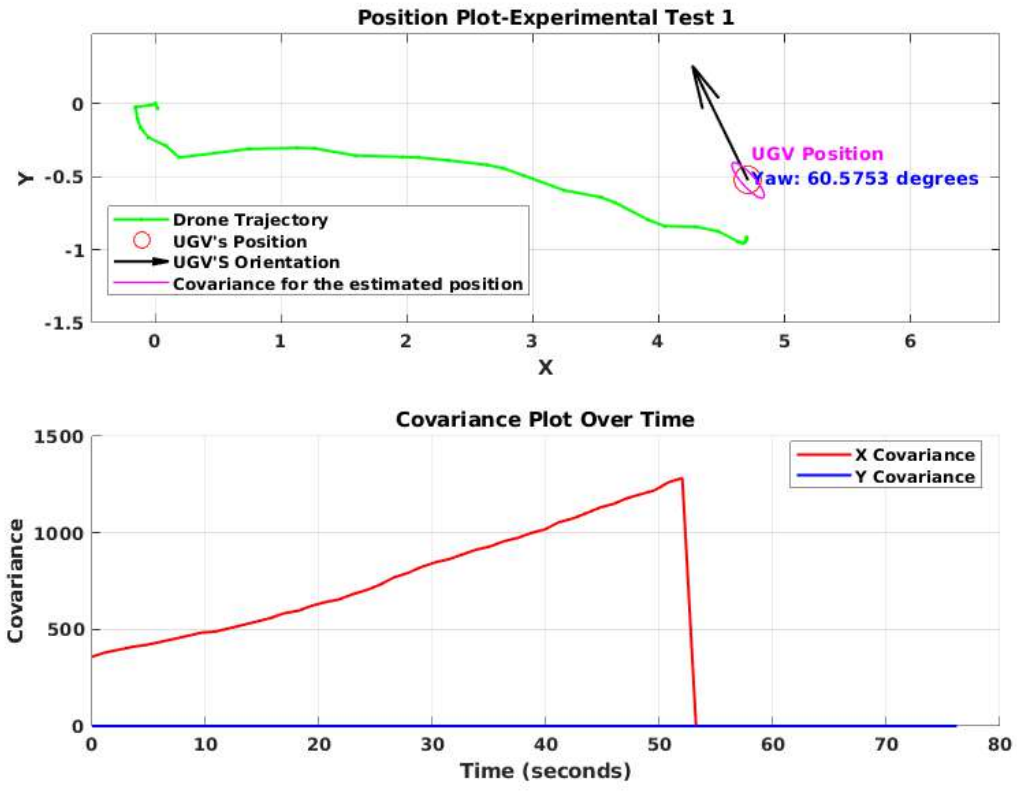


Figure 4.15. The records of the first experimental work for cooperative mission 2.

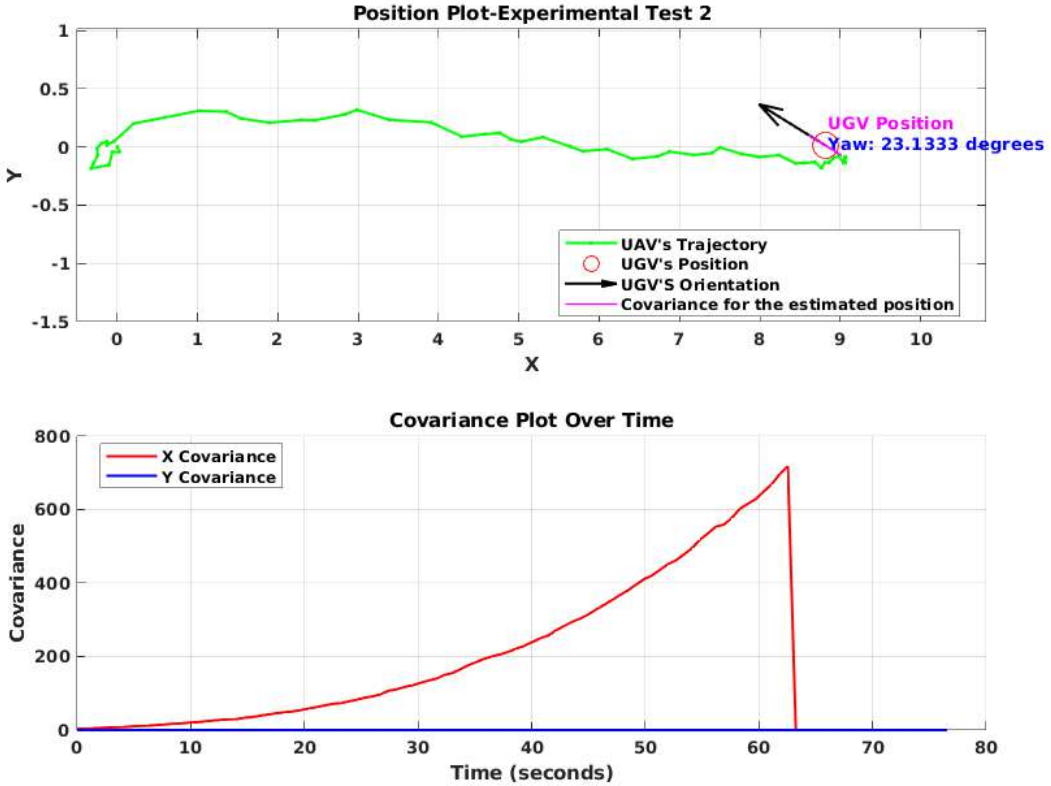


Figure 4.16. The records of the second experimental work for cooperative mission 2.

4.4. Cooperative Localization and Control-Mission 3

The test for this mission is performed to demonstrate the viability of these applications as a safety option for heterogeneous mobile robot systems like UTOPIA. During the test, the UAV remained in hold mode, while the UGV followed a predetermined trajectory. The limiting distance, referred to as A_d, was adjusted regularly. The UGV only adhered to its trajectory if A_d exceeded the relative distance between the UAV and the UGV. If not, the UGV suspended its trajectory. The records of A_d adjustments and the time-related position of the UGV are depicted in Figure 4.17.

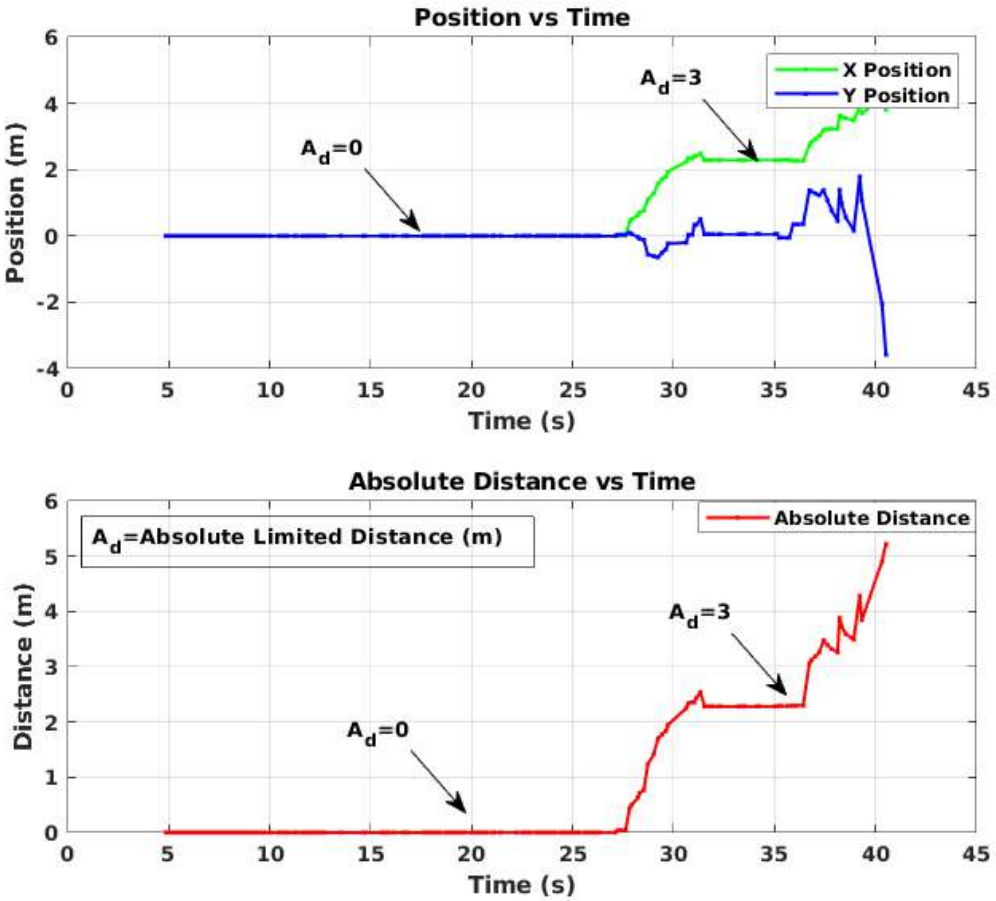


Figure 4.17. The records (UGV's Position and A_d) of the experimental work for cooperative mission 3.

5. CONCLUSION

This thesis contributes to the UTOPIA project, which focuses on cooperative localization and control for diverse mobile robots in challenging environments, such as agricultural settings. To achieve the project's objectives, a UAV and a UGV were developed. Each robot has a unique set of sensors and different maneuvering capabilities, each with its strengths and weaknesses.

Given these differences, robust localization is essential. EKF is tested with various configurations to achieve reliable localization. The localization accuracy of each robot was first improved through simulations and then confirmed in experimental tests. Then three cooperative missions are tested in both simulations and real-world experiments.

Simulation tests are performed using Gazebo sensor plugins to mimic real-world scenarios. However, only a few plugins are available to replicate the sensor sets used in experimental work. Thus, after pre-tests, reasonable configurations are created for each robot, as shown in Table 3.3 for the UAV and Table 3.6 for the UGV. The tests are then performed based on these configurations.

Based on the simulation tests, the algorithms are improved and modified for experimental work. It is observed that IMU alone, and IMU with Pose data, do not provide reasonable results. Visual odometry performs better indoors but worsens outdoors. Therefore, integrating IMU with visual odometry performs better. Outdoor localization is more challenging, and GPS has a correcting impact on localization.

The results reveal that incorporating more sensor data into the EKF reduces the RMSE, thereby improving localization. In homogeneous mobile robot systems, where each robot has similar sensors, localization can be achieved with a comparable RMSE. However, in heterogeneous systems, the higher variances in sensor capabilities and performance need careful management to ensure robust localization for cooperative

tasks.

For example, UAVs move faster and have a larger field of vision, but they have limited load capacity and battery life. In contrast, UGVs move slower but can carry more load and have longer battery life. These differences highlight the need for a robust localization system for effective task coordination.

One issue observed during experimental tests was time synchronization between the robots. In homogeneous mobile robot systems, this can be tolerated because each robot has similar components and similar moving capacity. However, in heterogeneous systems, it clearly affects task performance. For example, late messages and unmatched time steps during QR scanning led to incorrect estimations of the UGV's position in cooperative mission 2. To address this, we installed the "chrony sync time" package for Ubuntu and configured the ROS environment to eliminate time synchronization problems.

Another problem identified during the tests was the high computational costs and delayed messages due to outdoor wireless communication. The diversity of sensors and large amounts of high-frequency data caused the controller boards to work much slower in outdoor tests, leading to localization issues. We mitigated this by setting the robots to work headless, reducing GPU use, and optimizing each robot's EKF nodes to run on the robot's controller board. We also canceled the cameras' raw topics and unnecessary image topics, narrowed the camera's scope during image processing, and isolated each sensor from magnetic effects.

In conclusion, this research underscores the importance of integrating diverse sensor data to improve localization accuracy in heterogeneous robotic systems. The insights gained contribute to the development of robust cooperative localization and control strategies, enhancing the capabilities of mobile robots in complex environments. Future work should aim to refine these strategies further to improve the performance and reliability of diverse robotic systems in different applications.

REFERENCES

1. S. Bach, P. K. and S. Yi, "Application of QR Code for Localization and Navigation of Indoor Mobile Robot", *IEEE Access, Multidisciplinary*, Vol. 11, No. 1, pp. 28384–28390, 2023.

2. G. Atali, D. K., Z. Garip and S. Ozkan, *Localization of Mobile Robot Using Odometry, Camera Images and Extended Kalman Filter*, Polish Academy of Sciences, Institute of Physics, Warsaw, Polonia, 2017.

3. A. Yarovoi, Y. C., "Review of Simultaneous Localization and Mapping (SLAM) for Construction Robotics Applications", *Automation in Construction*, Vol. 162, No. 105344, pp. 1–11, 2024.

4. L. Li, L. S. and K. Kalavadia, "Promising SLAM Methods for Automated Guided Vehicles and Autonomous Mobile Robots", *5th International Conference on Industry 4.0 and Smart Manufacturing*, pp. 2867–2874, Prague, Czech Republic, 2024.

5. Zhang, S., S. Zhao and D. An, "Visual SLAM for Underwater Vehicles: A Survey", *Computer Science Review*, Vol. 46, No. 100510, pp. 1–16, 2022.

6. A. Mohseni, V. D. and T. Wong, "Improvement in Monte Carlo Localization Using Information Theory and Statistical Approaches", *Engineering Applications of Artificial Intelligence*, Vol. 131, No. 107897, pp. 1–14, 2024.

7. D. Vieria, M. S., R. Orjuela and M. Basset, "Positioning and Attitude Determination for Precision Agriculture Robots Based on IMU and Two RTK GPSs Sensor Fusion", *IFAC PapersOnLine*, Vol. 55, No. 32, pp. 60–65, 2022.

8. J. Lemke, R. S. and T. Oksanen, "Highly Accurate Obstacle Localization Using Fused Inertial, RTK-GNSS, and Lidar Positioning for Agricultural Field Operations", *IFAC PapersOnLine*, Vol. 55, No. 2, pp. 9269–9275, 2023.

9. N. Shalal, C. M., T. Low and N. Hancock, "Orchard Mapping and Mobile robot Localisation Using On-Board Camera and Laser Scanner Data Fusion – Part B: Mapping and Localisation", *Computers and Electronics in Agriculture*, Vol. 119, No. 1, pp. 267–278, 2015.

10. A. Leanza, A. U. E. C., R. Galati and G. Reina, "Where Am I Heading? A Robust Approach for Orientation Estimation of Autonomous Agricultural Robots", *Computers and Electronics in Agriculture*, Vol. 210, No. 107888, pp. 1–14, 2023.

11. J. Xiong, Y. Z. D. H., J. Liang and Z. Yang, "Real-time Localization and 3D Semantic Map Reconstruction for Unstructured Citrus Orchards", *Computers and Electronics in Agriculture*, Vol. 213, No. 108217, pp. 1–15, 2023.

12. M. GhaemiDizaji, C. D. and H. Leung, "Efficient Robot Localization and SLAM Algorithms Using Opposition Based High Dimensional Optimization Algorithm", *Engineering Applications of Artificial Intelligence*, Vol. 104, No. 104308, pp. 1–17, 2021.

13. L. Du, X. H., L. Chen and Y. Chen, "Cooperative Vehicle Localization Base on Extended Kalman Filter In Intelligent Transportation System", *The 28th Wireless and Optical Communication Conference (WOCC 2019)*, pp. 1–5, Beijing,China, 2019.

14. Park, S. and G. Lee, "Mapping and Localization of Cooperative Robots by ROS and SLAM in Unknown Working Area", *the SICE Annual Conference 2017*, pp. 858–861, Kanazawa, Japan, 2017.

15. H. Ebel, M. R. and P. Eberhald, "Cooperative Object Transportation With Differential-Drive Mobile Robots:Control and Experimentation", *Robotics and Autonomous Systems*, Vol. 173, No. 104612, pp. 1–10, 2024.

16. Y. Bergeon, V. K. J. M., R. Doskocil and A. Stefek, "UAV Assisted Landing on

Moving UGV", *International Conference on Military Technologies*, pp. 1–5, Brno, Czech Republic, 2015.

17. C. Kim, D. K., H. Yang and D. Lee, "2-D Cooperative Localization With Omni-Directional Mobile Robots", *The 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2015)*, pp. 425–426, Goyang city, Korea, 2015.

18. Isogai K., N. H., Inohara R. and H. Okazaki, "Modeling and Simulation of Motion of a Quadcopter", *2016 International Symposium on Nonlinear Theory and Its Applications*, pp. 165–168, Yugawara, Japan, 2017.

19. O. Tatale, S. P., N. Anekar and S. Sarkale, "Quadcopter: Design, Construction and Testing", *International Journal for Research in Engineering Application Management (IJREAM)*, Vol. 4, No. Special Issue, pp. 1–7, 2018.

20. T-Motor, `https://store.tmotor.com/product/u3-motor-u-power.html`, accessed on May 03, 2024.

21. ROS, "Robot Localization Documentation", `https://docs.ros.org/en/melodic/api/robot_localization/html/index.html`, accessed on May 08, 2024.

22. D., W. and M. Horn, *Automated Driving: Safer and More Efficient Future Driving*, Springer, Graz/Austria, 2016.

23. G., D. and J. M., *Computational Principles of Mobile Robotics*, Cambridge University Press, United Kingdom, 2012.

24. Hellström, T., *Kinematics Equations for Differential Drive and Articulated Steering*, Umea University, Umea/Sweeden, 2011.

25. Gazebo Plugins, `https://classic.gazebosim.org/tutorials?tut=ros_`

`gzplugins`, accessed on May 21, 2024.

26. ROS Wiki-Gazebo Plugins, `https://wiki.ros.org/gazebo_plugins`, accessed on May 21, 2024.

27. Salman, K., *Design And Implementation of a Model Predictive Trajectory Tracking Controller for a Quadcopter*, Master's Thesis, Boğaziçi University, 2018.

28. K., C. C. and G. Chen, *Kalman Filtering With Real-Time Applications*, Springer, Cham/Switzerland, 2017.