Developing a robotic testbed for relative asteroid navigation

Florina Crina Mihalache

Delft University of Technology

Developing a robotic testbed for relative asteroid navigation

by

Florina Crina Mihalache
4827333

in compliance with the requirements for obtaining the title of Master of Science
at the Faculty of Aerospace Engineering of Delft University of Technology
publicly defended on 18th of July 2025 at 09:30 AM

Committee:	Dr. ir. E. Mooij	TU Delft Astrodynamics and Space Missions, Supervisor
	Dr. ir. B.C. Root	TU Delft Planetary Exploration, Chair
	Dr. J. Guo	TU Delft Space Systems Engineering, External Member

Preface

This project has been conducted during a very beautiful period of my life, and I'd like to take a few moments to show my gratitude for the amazing people who have been there with me along this journey.

I feel extremely lucky to have had such an amazing supervisor who took me up on this challenging project, who had had faith in me that I would pull through even in times I doubted myself, and who has been a role model of endless curiosity, determination, highest standards of work all being beautifully balanced by humour, kindness and empathy for everyone around himself. Thank you, Erwin, for everything. And now for something completely different, I move on with my life but I will try my best to incorporate these qualities I have seen into my next chapters as well.

To my mum, because our journey together began with a university graduation, and now, as I reach my own, I wonder more than ever how you managed it all. You are a force of nature and a constant source of inspiration. Despite everything, you made my world a beautiful place and filled my mind with joy and hope. Thank you for giving me the best of yourself, especially your sarcasm and drive to turn dreams into reality, because none would be possible without you dreaming in the first place.

Mamaie, tataie, multumesc frumos pentru toti anii in care mi-ati fost aproape, in care m-ati încurajat să învăt mai bine, in care mi-ati oferit timpul si iubirea voastră si mi-ati reamintit mereu că si dacă îmi e greu, pot oricând să vin acasă, chiar dacă Gigi se suie si pe avion dacă e nevoie. Vă iubesc mult!

There are a few who, for the longest, have been my family away from home, but more recently, I learned that they can make any place feel like home. Alex, Andrada, Nico, Tavi, Tudor, you all have a very special place in my heart. I admire you deeply because you're all pretty damn extraordinary people! I loved growing up and kinda figuring out life together. I'm so excited for the rest of our AeRO adventures. A special thank you to Flori, the absolute best first Delft person I could have come across, and Vlad, who brought us all together and guided along the way.

Chloe, thank you for all the love, for always matching my energy, for making the low moments feel less dramatic and happy ones just happier. Irmak, thank you for being the best housemate I could have ever asked for. If I ever have to pick who I'm taking with me on a trip to Ganymede, it will for sure be you two.

Emmy, thanks for being my double marathon buddy. This has been, at times, really hard, and it meant the world to have someone to share it with.

I would also like to thank the rude guy who pushed me into Angel at the start of this thesis. I guess he knew I would need some reminders that life still happens regardless of whether I have everything figured out or not.

A last round of thank you to Delft and to the people of Delft who have been kind and nice and warm, for the small little shared moments and for the friendships and memories made.

Florina Crina Mihalache
Delft, July 2025

Abstract

Navigation of spacecraft around asteroids remains a critical challenge due to communication delays, uncertain surface topography, and the absence of reliable gravitational models. Testing such technologies on the ground using robotic systems enables faster and more reliable development. However, publicly available information on how to create operational frameworks for such testbeds is limited, often due to inter-agency competition or safety concerns at military test sites. This thesis investigates how relative asteroid navigation can be supported and replicated in a ground-based robotic laboratory environment. The work is conducted within the newly inaugurated Guidance, Navigation and Control (GNC) laboratory at TU Delft and focuses on supporting the initial global characterisation phase of an asteroid mission specifically, landmark identification and the generation of circular motion profiles using the laboratory hardware.

A robotic testbed is configured using a 6 degrees of freedom (dof) UR16e robotic arm, a 3 dof omnidirectional RB-Kairos rover base, and a 1:10,000,000 scale model of asteroid 433-Eros. A Python software framework is developed to define reference frame transformations between hardware components and generate trajectory points. Control commands for the rover base, robotic arm, and camera are synchronised and executed via ROS nodes. To enhance the laboratory's demonstration capabilities, a Gazebo simulation environment is implemented to test the trajectories. Experimental evaluations assess the open-loop motion tracking accuracy of the system. For a circular trajectory of 1200 mm radius, deviations reached 370 mm (width) and 276 mm (length). Additional uncertainties in the robotic arm's mounting position introduced mean errors between -39 mm and 66 mm (width), -46 mm and 57 mm (length), and 0.76 mm to 12.96 mm (height). These results indicate that closed-loop control is required to mitigate trajectory errors and improve positioning accuracy.

To support relative navigation, a photogrammetry pipeline is introduced to reconstruct the asteroid model from image data. Surface reconstructions using open-source software yield high-resolution meshes with up to 365,930 faces. Lighting configuration, image coverage, and environmental artefacts are identified as key limitations to reconstruction quality. The results demonstrate that the robotic laboratory can simulate circular arcs orbital motion. To further support relative navigation tasks alignment accuracy and feedback control have to be improved. Recommendations for future work include implementing closed-loop controllers, developing object-detection relative navigation, integrating adaptive lighting, and using the surface reconstruction models to estimate the gravity field of the asteroid model used in the laboratory.

Contents

Preface iii

Abstract v

List of Symbols ix

1 Introduction 1

2 Mission Heritage 5

2.1 Close-Proximity Operations to Asteroids 5

2.2 Robotics Labs for space technology testing 10

3 The Guidance, Navigation, and Control Robotics Lab 17

3.1 Overview GNC facility 17

3.2 RB-Kairos+ Rover Base 18

3.3 UR16e Robotic Arm 21

3.4 3D printed asteroid model 433-Eros 24

3.5 OptiTrack 25

3.6 Asteroid Tracking Camera 27

3.7 Setting up a simulation for the laboratory 29

4 Reference Frames 33

4.1 Coordinate Systems 33

4.2 Reference Frames 37

4.2.1 Orbital Reference Frames 37

4.2.2 Laboratory Reference Frames 40

4.3 Frame Transformations 44

4.4 Summary Frame Implementations 46

5 Modelling the Movement of the Robotic Setup 47

5.1 Software Used 47

5.2 Proposed Tests 48

5.3 Case 1: Mobile Base, Static Arm 52

5.3.1 Input Command Generation 53

5.3.2 Error Metrics for the Rover Base 54

5.3.3 Performance Evaluation Rover Base 55

5.4 Case 2: Static Base, Mobile Arm 66

5.4.1 Selection of the inverse kinematics solver 66

5.4.2 Creating the control input 68

5.4.3 Performance Evaluation Robotic Arm 73

5.5 Case 3: Coordinated Base and Arm Movement 81

5.6 Uncertainty Sources 85

6 Surface Reconstruction 89

6.1 Photogrammetry 89

6.2 Image Datasets 90

6.3 Open-source Photogrammetry Software 91

6.4 Discussion on 3D Surface Reconstruction Capabilities 97

7 Conclusions and Recommendations 101
7.1 Addressing Research Sub-Questions 102
7.2 Specification Document 103
7.3 Recommendations and Future Work 104
A Appendix A: Command-Line for Robot and Camera Operation 111
B Appendix B: Installing the GNC Lab Gazebo Simulation 113
C Appendix C: Human Robot Interface 115
D Appendix D: Joint Commands Visualisation 117

List of Symbols

Latin Symbols

a	Semi-major axis [m]
bL, bW, bH	Rover base length, width, height [m]
$C_x(\phi)$, $C_y(\theta)$, $C_z(\psi)$	Rotation matrices about body axes [-]
$\mathbf{C}_{B/A}$	Direction Cosine Matrix from A to B [-]
$\mathbf{I}_3$	3×3 identity matrix [-]
$\dot{x}$, $\dot{y}$, $\dot{z}$	Cartesian velocity components [m/s]
$\dot{\mathbf{r}} = [\dot{x}, \dot{y}, \dot{z}]^T$	Velocity vector [m/s]
e_1, e_2, e_3	Eigenaxis components [-]
$\bar{\mathbf{e}}$	Mean error over all waypoints [m]
$\mathbf{e}_i$	Pointwise error at waypoint i [m]
F	Gravitational force [N]
G	Gravitational constant [$m^3\ kg^{-1}\ s^{-2}$]
m_1, m_2	Masses of two bodies [kg]
mxT, mzT, hT	Marker offsets: longitudinal, lateral, vertical [m]
N	Number of waypoints [-]
$\mathbf{q} = [q_0, q_1, q_2, q_3]^T$	Quaternion [-]
q_0, q_1, q_2, q_3	Quaternion components [-]
r	Distance between centres of mass [m]
r_0	Circular orbital radius [m]
$\mathbf{r} = [x, y, z]^T$	Position vector [m]
s_i^{exp}, s_i^{meas}	Expected and measured states at waypoint i [m]
$\mathbf{T}$	Translation vector between frame origins [m]
v_0	Circular orbital speed [m/s]
v_x, v_y, v_z	Components of $\dot{\mathbf{r}}$ [m/s]
x, y, z	Cartesian coordinates [m]
x_C, y_C, z_C	Camera origin offsets from marker [m]
x_T, y_T, z_T	Target origin offsets from marker [m]

Greek Symbols

Δt	Time increment [s]
ϕ	Roll angle about the x-axis [rad]
θ	Pitch angle or true anomaly [rad]
ψ	Yaw angle about the z-axis [rad]
ρ	Radial distance in spherical coordinates [m]
ρ	Density [kgm^{-1}]
λ	Longitude in spherical coordinates [rad]
μ	Gravitational parameter, G M [m^3s^{-2}]
Ω	Right ascension of ascending node [rad]
ω	Argument of periapsis [rad]

Reference Frames

I	Inertial reference frame centred at the asteroid’s mass centre [-]
A	Asteroid body-fixed frame, principal axes [-]
B	Spacecraft body-fixed frame [-]
L	Laboratory inertial frame [-]
O	OptiTrack global calibration frame [-]
OBC	OptiTrack frame of the rover base chaser markers [-]
OC	OptiTrack frame of the chaser (camera) markers [-]
OT	OptiTrack frame of the target (asteroid) markers [-]
BC	Rover base chaser frame (body-fixed) [-]
AMC	Arm mount on the chaser rover base [-]
C	Chaser (camera) frame at the end-effector [-]
T	Target (asteroid) body-fixed frame [-]

Abbreviations

AFC	Asteroid Framing Camera
APL	Johns Hopkins University Applied Physics Laboratory
ATV	Automated Transfer Vehicle
CUDA	Compute Unified Device Architecture
DART	Double Asteroid Redirection Test
DCM	Direction Cosine Matrix
DLR	Deutsche Forschungsanstalt für Luft- und Raumfahrt
dof	degrees of freedom
DRACO	Didymos Reconnaissance and Asteroid Camera for Optical Navigation
DSN	Deep Space Network
DTM	Digital Terrain Map
EKF	Extended Kalman Filter
EPOS	European Proximity Operations Simulator
ESA	European Space Agency
FOV	Field Of View
fps	frames per second
GCP	Ground Control Points
GN	Gauss-Newton
GNC	Guidance, Navigation, and Control
GRALS	GNC Rendezvous, Approach, and Landing Simulator
GPU	Graphics Processing Unit
HRI	Human Robot Interface
HTV	H-II Transfer Vehicle
ICT	Information Communication and Technology
IK	Inverse Kinematics
ISS	International Space Station
JAXA	Japan Aerospace Exploration Agency
JPL	Jet Propulsion Laboratory
LEO	Low Earth Orbit
lidar	Light Detection and Ranging
LM	Levenberg-Marquardt
MSI	Multispectral Imager

MVS	Multi-Video Stereo
NASA	National Aeronautics and Space Administration
NAVCAM	Navigation Camera
NEAR	Near Earth Asteroid Rendezvous
NEO	Near Earth Object
NFT	Natural Feature Tracking
NR	Newton-Raphson
OSIRIS	Optical, Spectroscopic, and Infrared Remote Imaging System Camera
OSIRIS-REx	Origins, Spectral Interpretation, Resource Identification, and Security Regolith Explorer
RAAN	Right Ascension of the Ascending Node
RMSE	Root Mean Square Error
ROS	Robot Operating System
RPO	Rendezvous and Proximity Operations
SCOM	Satellite Communications and Orbital Mechanics
SfM	Structure-from-Motion
SLAM	Simultaneous Localization and Mapping
SMART Nav	Small-body Maneuvering Autonomous Real-Time Navigation
SPC	Stereophotoclinometry
SRP	Solar Radiation Pressure
SSH	secure shell
TMT	Target Marker Tracking
TRON-DLR	Testbed for Robotic Optical Navigation at DLR
TRON-Stanford	Robotic Testbed for Rendezvous and Optical Navigation at Stanford
TRL	Technology Readiness Level
URDF	Unified Robotics Description Format

1 Introduction

Exploring the surface of celestial bodies is no trivial task, whether the observations are performed from the ground or through means of a spacecraft deployed in space for a mission. The latter comes with its unique set of challenges, as in addition to performing the measurements of interest, there is the additional engineering challenge of correctly evaluating the location of the spacecraft, orientation relative to its target, and commanding its trajectory to follow the desired path such that the measurements can be taken at the key-point locations. Larger celestial bodies have been observed in the past by spacecraft, and multiple missions have successfully landed on the surface of planets. More recently, scientific interest in asteroids has been growing as understanding their origins may reveal insights into solar system formation. Additionally, certain asteroids, classified as Near Earth Object (NEO), pose potential collision threats to Earth, making their study crucial for planetary defence. Moreover, asteroids are rich in various metals, making them a significant resource for future mining endeavours.

Despite the increasing focus on asteroid missions, current navigation methods still rely heavily on ground-based supervision, underlining the need for more advanced autonomous techniques. The challenges include communication delays, limited data transmission rates, and incomplete knowledge of asteroids' physical properties, such as exact shape models, composition, and gravity fields.

To address these issues, the research community has focused on developing autonomous navigation methods, with optical navigation through landmark tracking emerging as a promising approach (Olds et al., 2022). Several recent missions, such as NASA's Near Earth Asteroid Rendezvous (NEAR) Shoemaker, DART, and ESA's upcoming Hera mission, have demonstrated advances in this area. The Hera mission, in particular, aims for full autonomy, using data fusion and an Asteroid Framing Camera (AFC) to model the asteroid and navigate relative to it simultaneously (Volpe et al., 2020).

Ground-based testing is essential to the success of asteroid navigation missions. However, replicating real mission conditions in a laboratory environment poses inherent challenges. Full-scale testing of asteroid navigation algorithms is impractical due to the vast space required. This limitation makes it necessary to determine which aspects of the system can be tested within the constraints of available facilities, such as those at the Aerospace Engineering Faculty of Delft University of Technology. This study aims to lay the groundwork for establishing a framework and organisational structure for the department's new Guidance, Navigation, and Control (GNC) facility, dedicated to research on relative asteroid navigation. The work builds on the recommendations of the most recent study, in which manual experiments were conducted to produce experimental navigation measurements (Terwindt, 2025). The main objective is to evaluate how the hardware currently available within the facility can be utilised to automate the execution of navigation measurements. Previous research within the department has primarily focused on more advanced navigation techniques, such as surface scanning (Razgus, 2017) and gravity field estimation (Munuera Vilalta, 2024), both requiring more complex orbital configurations. Consequently, the framework developed in this study is designed to be modular, enabling future expansion and integration of different orbital parameters.

The limited availability of ground-based testing facilities and simulators for asteroid missions stems from factors such as safety restrictions, competition between space agencies, and the military affiliations of certain test sites. Consequently, publicly accessible information on the testing protocols and simulators for asteroid navigation is scarce. Given the high costs, extended mission durations, and significant risks associated with space missions, the development of terrestrial simulators that can accurately replicate the conditions of asteroid navigation is critical. In the academic context, where theses and research projects cannot be directly translated into space missions, such simulators offer an additional layer of technical rigour, allowing students and researchers to test, verify, and validate navigation techniques before their real-world application.

This thesis aims to contribute to this effort by developing a hardware-in-the-loop simulation method to support the testing of control and navigation techniques for close-proximity operations around asteroids. Although creating a laboratory environment that accurately mirrors real-world conditions poses significant challenges, overcoming these obstacles is essential to advance asteroid exploration and ensure mission success. This leads to the central research question:

Research Question

How can the development of autonomous asteroid navigation be supported in a robotic-lab environment?

Research subquestions

1. **Experiment Design and Validation**
   (a) How can a robotic system be designed and utilised to simulate the navigation around a 3D-printed model of Eros?
   (b) What parameters need to be replicated in the laboratory trajectory to be representative of the space environment?
   (c) What are the requirements for the external monitoring of the setup using a tracking system like OptiTrack?

2. **Model Accuracy and Detail**
   What level of detail and accuracy is required in the 3D-printed model of Eros to ensure reliable navigation and tracking?

3. **Experiment Evaluation**
   (a) To what extent can the robotic system accurately replicate the trajectory profiles required for relative asteroid navigation?
   (b) Which features of autonomous navigation around an asteroid can be tracked using the available set-up?

Thesis Structure

To address the research question, the thesis work presented in this report is structured as follows.

Chapter 2 summarises the literature review conducted to better understand the types of missions and manoeuvres in which spacecraft operate in close proximity to asteroids. It also explores the setup of research laboratories dedicated to on-ground testing of such missions.

Chapter 3 provides an overview of the hardware available within the newly inaugurated Guidance, Navigation, and Control (GNC) Laboratory of the Faculty of Aerospace Engineering. It details how the hardware has been configured and used for this work, and includes additional information on the development of a software simulation intended to expand the current demonstration capabilities of the laboratory.

In Chapter 4, the methodology used to establish equivalence between points along a spacecraft's orbital trajectory around an asteroid and corresponding points within the Cartesian space of the laboratory is discussed.

Chapter 5 outlines the modelling and assessment of the movement of the robotics system, with a focus on the accuracy and repeatability of the generated trajectories.

Chapter 6 discusses how surface reconstruction of the asteroid model can be performed from experimental images and how this approach can contribute to relative asteroid navigation in the laboratory.

Finally, this research's conclusions and recommendations are presented in Chapter 7. This chapter answers the research questions and introduces a specification document defining the derived requirements for hardware operation for future experiments in the GNC laboratory. Lastly, it introduced recommendations and suggestions for future research.

2 Mission Heritage

This study's research question and objectives are divided into two primary components. The first relates to the increasing interest in asteroid missions and the advancements in autonomous navigation technologies for such missions into the broader space research community, specifically within the Spaceflight Department of the Aerospace Engineering Faculty at Delft University of Technology. By examining past and upcoming missions, it becomes evident which type of manurers are expected in asteroid navigation and the levels of relative positioning accuracy that spacecraft must achieve to perform scientific measurements at the surface. The literature review presented in Section 2.1 helps to identify the type of operations and measurements that must be supported by ground-based testing before a mission is launched.

The second and central component of the research question focuses on robotic laboratory environments and how they can be designed and employed to verify and validate close proximity space operations. A classification of relevant robotic laboratories and detailed descriptions of selected examples applicable to this study are provided in Section 2.2.

2.1. Close-Proximity Operations to Asteroids

Asteroid missions have been of interest for several decades, particularly those involving NEOs that pose a potential collision risk to Earth. Additionally, the scientific community has recognised that studying these small bodies can offer valuable insights into the formation and evolution of the Solar System. These missions also present unique operational challenges, such as communication delays and limited situational awareness, which have accelerated the development of autonomous navigation and tracking capabilities. Siddiqi (2016) provides a comprehensive overview of deep space exploration missions conducted between 1958 and 2016, including those targeting asteroids. Interest in asteroid exploration extends well beyond this timeframe, with multiple active missions.

The review of past asteroid missions and the operational planning of ongoing and future missions such as Hera reveals a consistent structure in how close proximity operations are organised. These operations can be divided into three main stages, initially summarised by Van der Heijden (2022), and reinforced by the current mission profile of ESA's Hera mission (Küppers & Gil Fernandez, 2021). This structure is used to refine the knowledge of the target asteroid progressively.

- **Initial global characterisation:** This phase focuses on determining the basic physical parameters of the asteroid, including its overall shape, mass, rotation period and gravitational field. These properties are essential for accurate dynamical modelling of the asteroid's orbital motion and the spacecraft around it, as well as for planning subsequent manoeuvres.

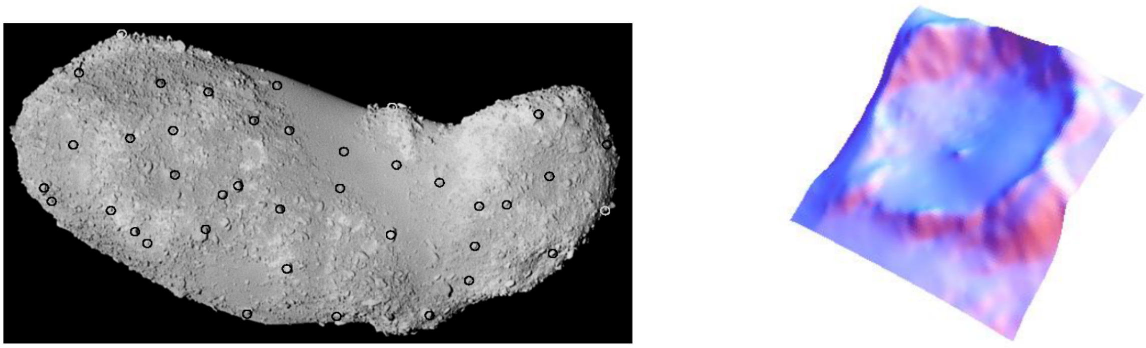
- **Detailed characterisation and modelling:** After the initial orbit determination, the spacecraft performs closer flybys to acquire data for building shape models and surface maps and identifying key surface features, usually referred to as landmarks. This phase often employs techniques such as Stereophotoclinometry (SPC) or Light Detection and Ranging (lidar) scanning.
- **Execution of scientific objectives:** With the asteroid fully mapped and its environment characterised, the mission enters its primary science phase. Depending on the mission architecture, this may include multipoint surface and subsurface investigations, sample collection, surface composition analysis, or impact experiments.

This mission architecture is clearly reflected in the most recent asteroid mission, Hera, on its way to observe the binary asteroid system Didymos–Dimorphos. The Hera spacecraft will begin with an **early characterisation phase** at altitudes between 20 and 30 km, during which the shape and mass of Didymos are estimated to initialise navigation filters and gravity models. It will then proceed to a **detailed characterisation phase**, with closer flybys between 10 and 20 km used to measure surface properties, subsurface structure, and environmental conditions. In the final stage, Hera will complete its **scientific objectives**, including a multipoint investigation of the crater formed by Double Asteroid Redirection Test (DART) mission, perform internal structure analysis using low-frequency radar, and gravity field measurements using onboard gravimeters and CubeSats (Küppers & Gil Fernandez, 2021).

Landmarks refer to distinctive features on the surface of a target body. These may include concave or convex regions, surface creases, rocks, boulders, or any other identifiable surface irregularities as defined by the mission research or navigation teams. Landmarks can be either natural or artificial, and their definition depends on the specific tracking or guidance method employed. Examples of natural landmarks are shown in Figure 2.1a, while synthetically modelled features used for navigation purposes are depicted in Figure 2.1b. Once defined, landmarks serve as reference points that can be tracked, either manually by ground-based operations teams or autonomously by onboard GNC systems. In some missions, artificial landmarks have been physically deployed on the surface to assist in optical tracking. A notable example is the use of reflective target markers during the Hayabusa and Hayabusa2 missions, shown in Figure 2.2.

The selected missions include NEAR Shoemaker, Hayabusa, Rosetta, Dawn, Origins, Spectral Interpretation, Resource Identification, and Security Regolith Explorer (OSIRIS-REx) DART, and Hera. Each mission targeted a different small body and employed a variety of navigation and imaging techniques, including landmark tracking, Natural Feature Tracking (NFT) (Olds et al., 2022), Target Marker Tracking (TMT) (Ogawa et al., 2020). All missions were equipped with optical systems capable of capturing high-resolution surface imaging, which was subsequently used for tasks such as relative navigation, shape reconstruction, and landing site identification. These missions were executed by three major space agencies: National Aeronautics and Space Administration (NASA), European Space Agency (ESA), and Japan Aerospace Exploration Agency (JAXA).

NEAR Shoemaker (NASA, 1996–2001) is identified as the most relevant heritage mission for this thesis due to the availability of a high-resolution shape model of its target asteroid, 433-Eros (Gaskell et al., 2006), and the subsequent creation of a physical 3D-printed model developed by Terwindt (2025) which can be directly used for experiments. The mission was launched on 17 February 1996, entered orbit around 433-Eros on 14 February 2000, and successfully performed a landing on 12 February 2001. Throughout the mission, NEAR Shoemaker conducted a series of close-proximity operations, including circular orbits at altitudes of 200 km, 100 km, 50 km, and 35 km. In the final phase, the spacecraft executed multiple



(a) Surface image used for landmark navigation on asteroid Itokawa Ground Control Points (GCP) (Morita et al., 2006).

(b) 3D surface map of a crater on 67/P. The landmark is the rock standing in the centre of the map (Pardo de Santayana & Lauer, 2015).

Figure 2.1: Examples of landmarks used for navigation on small Solar System bodies.

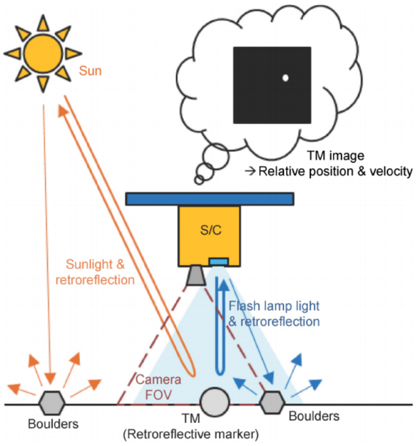

Sun
TM image
→ Relative position & velocity
S/C
Sunlight & retroreflection
Flash lamp light & retroreflection
Camera FOV
Boulders
TM (Retroreflective marker)
Boulders

Figure 2.2: Hayabusa Target Marker Tracking (TMT) (Ogawa et al., 2020)

low-altitude passes at distances below 5 km (Antreasian et al., 2001). Navigation relied primarily on optical landmark tracking, supported by Doppler measurements from the Deep Space Network (DSN). The optical navigation process integrated imaging data from the Multispectral Imager (MSI) with a pre-processed database of surface landmarks, including crater features, generated by the Jet Propulsion Laboratory (JPL) (Miller et al., 1995).

Hayabusa (JAXA, 2003–2010) targeted asteroid 25143 Itokawa. It demonstrated optical navigation through natural terrain features and artificial TMT markers. Although the mission was designed to be autonomous, a failure in the lidar system required human-in-the-loop intervention for part of the descent (Hashimoto et al., 2010). The mission used retro-reflective target markers to support landing, tracked using a wide-angle camera with artificial illumination from a flash lamp (Ogawa et al., 2020).

Rosetta (ESA, 2004–2016) studied comet 67P/Churyumov–Gerasimenko using optical landmark tracking. Navigation was supervised on ground and had limited autonomy. The mission deployed the Philae lander, whose site selection relied on 3D terrain reconstruction and polyhedral modelling (Muñoz et al., 2015). High-resolution terrain mapping was achieved using the Optical, Spectroscopic, and Infrared Remote Imaging System Camera (OSIRIS) camera

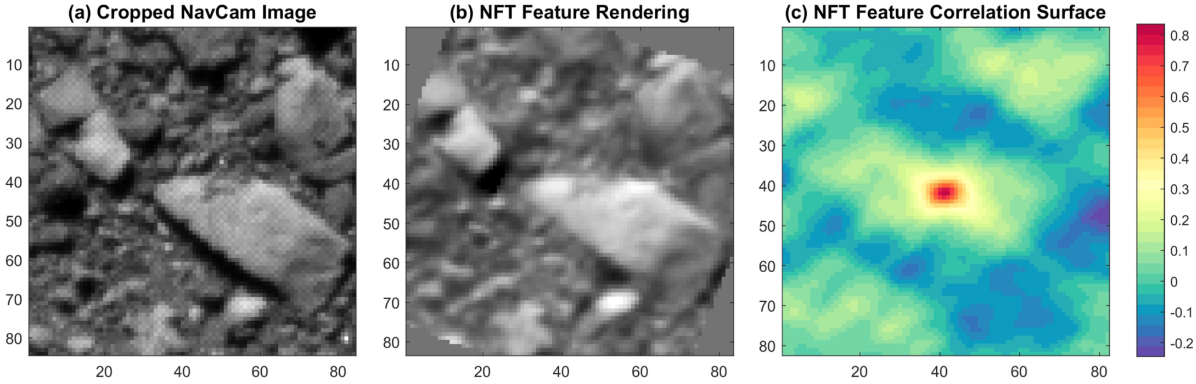


Figure 2.3: Landmark tracking using Natural Feature Tracking (NFT) Bennu (Olds et al., 2022)

suite and Navigation Camera (NAVCAM), enabling landmark-based trajectory control. A total of 1,157 surface features were tracked in over 6,280 images (Pardo de Santayana & Lauer, 2015).

Dawn (NASA, 2007–2018) orbited Vesta and Ceres using a method called L-maps landmark navigation (Mastrodemos et al., 2011). This method involves generating a Digital Terrain Map (DTM) of the target body. The landmarks visible in the camera Field Of View (FOV) are then compared with those from the DTM to determine the position of the spacecraft relative to the asteroid. However, the accuracy of this method is constrained by the precision of the pre-generated DTM and the available shape model of the asteroid, which is based on the resolution of the images captured.

OSIRIS-REx (NASA, 2016–2023) was the first mission to perform autonomous optical navigation based solely on natural terrain. It introduced Natural Feature Tracking (NFT), in which features were matched to a pre-uploaded high-resolution DTM, enabling state estimation via an onboard Extended Kalman Filter (EKF) (Olds et al., 2022). The method estimated the spacecraft's position by comparing the input from the navigation camera to synthetically modelled surface renderings. A correlation metric was then computed between the observed and rendered images to evaluate the confidence level of each match, thus updating the filter's state estimation. An example of this process is shown in Figure 2.3.

DART (NASA, 2021–2022) demonstrated autonomous kinetic impact guidance for planetary defence. The mission impacted Dimorphos, a moonlet of Didymos, using Small-body Maneuvering Autonomous Real-Time Navigation (SMART Nav), a guidance algorithm that processed images from the Didymos Reconnaissance and Asteroid Camera for Optical Navigation (DRACO) a narrow-angle instrument with 0.29° FOV. The SMART Nav system was integrated into the spacecraft GNC framework. It autonomously identified and distinguished between the two bodies and guided the spacecraft toward Dimorphos. These algorithms, built on decades of missile guidance expertise developed at Johns Hopkins University Applied Physics Laboratory (APL), enabled DART to achieve high-resolution imaging of Dimorphos, down to 3 cm/pixel, in the final hour before impact.

Hera (ESA, 2024–2026) will rendezvous with the Didymos–Dimorphos system in 2026. It will operate with full autonomy using Simultaneous Localization and Mapping (SLAM), fusing data from the AFC and onboard lidar (Volpe et al., 2020). This approach enables the spacecraft to construct a real-time model of the asteroid and simultaneously navigate relative to it. The

AFC, a monocular camera, plays a central role in this process by providing continuous visual input for feature detection and pose estimation. For the SLAM algorithm to be effective, the asteroid must remain entirely within the camera's field of view during the autonomous phase. At distances greater than 8.5 km, the full shape of Didymos fits within the image frame, allowing vision-based navigation to proceed without requiring a preloaded detailed shape model (Küppers & Gil Fernandez, 2021).

Discussion and Relevance to Laboratory Testing

Across all reviewed missions, optical navigation has evolved from ground-assisted landmark tracking (e.g., Rosetta) to fully autonomous onboard image processing (e.g., OSIRIS-REx, Hera). While these newer methods demonstrate promising capabilities in autonomous spacecraft operations, they often depend on specialised hardware developed by mission-specific contractors or space agencies. Such systems, including high-resolution cameras, advanced onboard processors, or deep-space communication interfaces, are not readily available for academic research environments.

As a result, the development of a robotic testbed for relative asteroid navigation must initially focus on defining a framework that simulates the spacecraft and asteroid motion realistically and can produce representative visual input. This input can then be post-processed to evaluate landmark detection, tracking, and navigation performance in a controlled laboratory setting. The emphasis is on building a modular and extensible system that supports scalable levels of autonomy as hardware capabilities improve.

Analysis of the selected mission heritage reveals a consistent operational structure, where all missions begin with an initial characterisation phase conducted from relatively large orbital radii. This phase typically aims to determine the asteroid's global shape, identify distinguishable landmarks, and estimate key parameters such as rotation rate, gravity field, and reference frames. It is considered a foundational step in developing and validating any visual navigation method for asteroid proximity operations, and is therefore prioritised in this study.

Previous work by Terwindt (2025) validated basic landmark tracking using a static camera and a manually rotatable 3D-printed model of 433-Eros, based on data from the NEAR Shoemaker mission. Although this setup demonstrated the viability of optical feature extraction, it was inherently limited to static or highly simplified motion profiles and excluded any dynamic interaction between the spacecraft and the asteroid model.

This thesis extends that work by addressing its main limitations. A new laboratory framework is developed to replicate dynamic trajectories using robotic motion of both the asteroid and the camera system. The objective is to enable simulation of realistic, low-complexity orbital characterisation phases, where relative motion can be performed, repeated, and evaluated. This modular approach allows the GNC lab to serve as a testing platform for progressively more advanced navigation scenarios, including elliptical orbits, inclined planes, and autonomous visual tracking.

2.2. Robotics Labs for space technology testing

This section investigates different hardware-in-the-loop simulators for manoeuvres of spacecraft, with a particular focus on proximity operations. Proximity manoeuvres are crucial for navigating spacecraft around another orbiting object, and include activities such as formation flying, docking, capture, and the robotic manipulation of target objects. The need to recreate the kinematics and/or dynamics of a spacecraft in motion on ground has led to the development of numerous simulators and testbeds in academic institutions, government facilities and industry laboratories (Wilde et al., 2020). However, information on their set up and working is scarce due to safety concerns. These simulators and testbeds vary in dimensions and most of them can test missions at reduced scales in comparison to the full-scale system laboratories at different NASA centres[1].

Rendezvous and Proximity Operations (RPO) are instrumental across various spaceflight operations and missions. These operations are not only pivotal to missions aimed at asteroid investigation, as detailed in Section 2.1, but are present in historical applications such as the docking manoeuvres executed during the Apollo Moon missions (Hatch et al., 1967).

Robotics laboratories are essential for developing, testing, and refining the technologies that enable these complex manoeuvres and operations. However, setting up the system in such a way that the simulator system reproduces the relevant characteristics of the space environment with an adequate degree of fidelity poses its own unique set of challenges as described by Wilde et al. (2020)

Relative Motion The relative motion between a chaser spacecraft and a target can at times involve multiple perturbations affecting the relative trajectories of the centre-of-mass of the two bodies. Usually the perturbations are compensated by the GNC system of the spacecraft. Testing of relative motion for different GNC systems requires a **kinematic simulator**

Quasi-Frictionless Environment Due to the lack of friction as a damping force in space, the effects of multi-body dynamics and contact dynamics are more pronounced in space compared to the terrestrial environment. During the mission, the spacecraft is impacted by many different forces and torques in the environment. The actuation system also creates internal forces and moments for attitude control. Evaluating the impact of these forces requires a **dynamics simulator** testbed equipped with robotic manipulators that provide contact dynamics feedback. For tracking landmarks around an asteroid, it will not be necessary to simulate any contact dynamics as the spacecraft will not be in contact with the surface. However, it is important to consider the differences in terms of the simulated motion of the spacecraft in a numerical simulation, where external forces and torques are present in the quasi-frictionless environment and the simulated motion of the spacecraft in the hardware experiment where air friction is present.

Lighting Conditions The lighting conditions during RPO is characterised by strong contrasts between the light brightly illuminated areas and the shadowed regions. In order to test the lighting conditions, it is necessary to simulate both the spectrum and the intensity of direct sunlight.

Following the classification created by Wilde et al. (2020, 2019), spacecraft manoeuvre simulators can be divided into two categories. The division is based on the nature of the simulated motion, kinematics, and dynamics simulators. However, it is worth mentioning that other different types, such as hybrid or kino-dynamics simulators, exist as well. The main difference between kinematics and dynamics simulators is with respect to the means used to simulate the forces and torques generated by the spacecraft during the hardware in the

[1] NASA Test Facilities Overview https://www.nasa.gov/setmo/facilities/ (accessed: 8 May 2025)

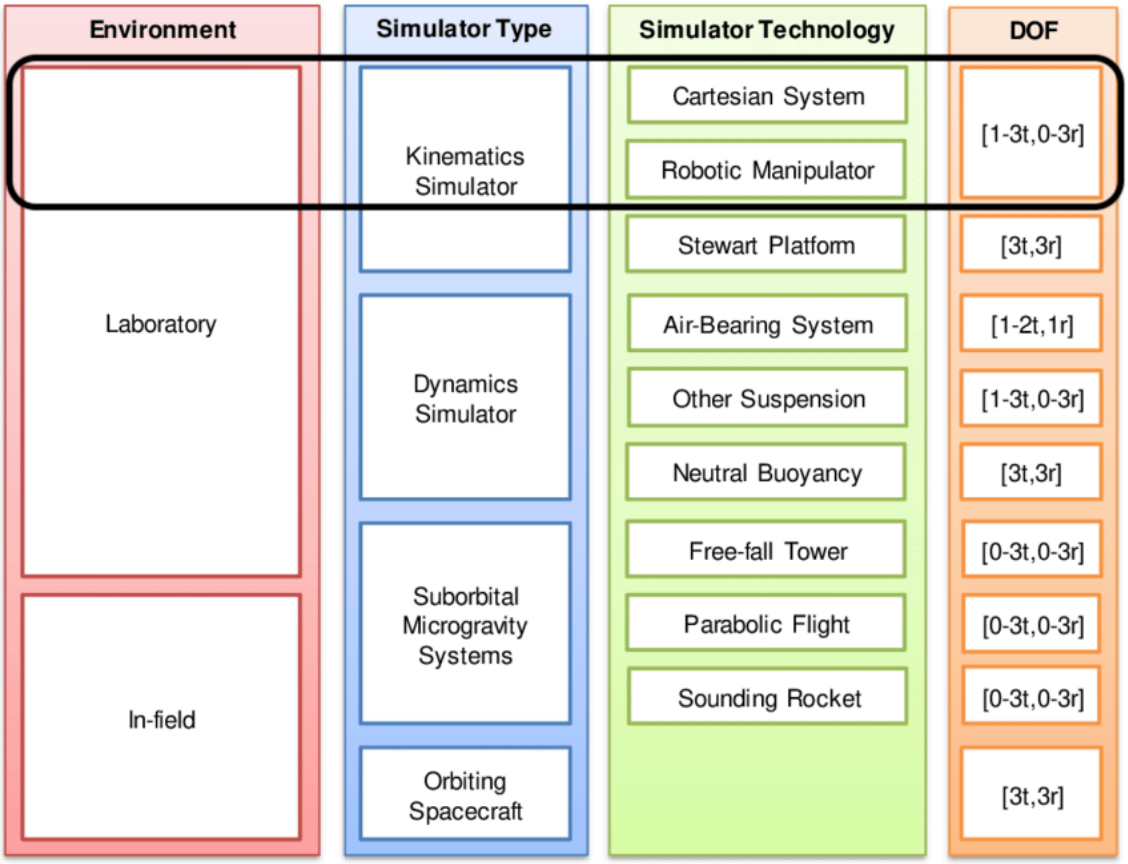


Figure 2.4: Overview of spacecraft-simulator systems (Wilde et al., 2019) and selection of design space for spacecraft simulator for the GNC Laboratory at TU Delft

loop testing. In the case of a kinematic simulator, the motion of the spacecraft (translational and/or rotational) is imposed on the simulated spacecraft using forces and torques generated by external actuators that are not necessarily the same actuation method as the ones used on the real spacecraft. For a dynamics simulator, the actuators used are representative of the actuators used in real flight, such as thrusters of reaction wheels (Wilde et al., 2020).

Kinematics Simulator These simulators are typically used to reproduce the relative orbital dynamics for testing of sensors and GNC systems. They must be equipped to move the test body precisely along commanded trajectories or into commanded positions and orientations (Wilde et al., 2020). The actuators and on-board systems are usually commanded by external computers. Therefore, the available computing power is usually higher than in the case of a dynamics simulator. The majority of kinematic simulators are Cartesian motion simulators with an independent positioning mechanism for each dof, and they simulate the orbital lighting conditions

Dynamics Simulator This type of simulator requires a decoupling of the dynamics of the simulation vehicle from its environment, and the motion of the simulated spacecraft must be controlled by internal onboard computers and actuators. Thus, the communication between the tested spacecraft and the environment must be wireless. This method has the advantage of better representing the real flight test scenarios; however, it drastically limits the available power and computing resources. To account for the usage of the real actuators, this type of simulator must be equipped with additional systems and suspension methods to isolate the simulated spacecraft from different effects of gravity and from some additional forces and torques resulting from the ground. Thus, this type of simulator has to account for both the kinematics and at least partially for the dynamics of the physical problem described.

Hybrid Simulator This simulator brings an extra layer of complexity and realism by combining the motion mechanics of kinematics simulators with dynamic decoupling elements from the dynamics simulator.

Kino-Dynamics Simulator The main advantage of this type of simulator is provided by the feedback of force and torque measurements to the simulation model to recompute the trajectories and improve the positioning. This method generally reaches good fidelity and in the case of close proximity operations, it reduces the risk of damaging the simulation mechanism compared to Hybrid Simulators.

Kinematics simulators are typically used to test sensor performance, navigation algorithms, or spacecraft teleoperation. Meanwhile, dynamics simulators are mainly used for actuators' performance testing and contact dynamics during docking and capture scenarios. Based on the current hardware and development status of the GNC Laboratory, the robotic testbed for asteroid navigation for this work will contribute towards developing a kinematic simulator.

As previously mentioned, detailed information on hardware simulators used for the verification and validation of space missions is limited. However, more substantial data are available for four key laboratories: European Proximity Operations Simulator (EPOS), due to its involvement in several ESA missions; Testbed for Robotic Optical Navigation at DLR (TRON-DLR) and GNC Rendezvous, Approach, and Landing Simulator (GRALS), through prior collaborations with students from the department; and Robotic Testbed for Rendezvous and Optical Navigation at Stanford (TRON-Stanford), via publicly available academic resources. These facilities play significant roles in the advancement of spacecraft proximity operations and hardware-in-the-loop testing. The EPOS facility at the German Aerospace Center (DLR) specialises in advanced rendezvous and docking simulations, and has contributed to the development of control algorithms for complex orbital operations in support of ESA missions (Boge et al., 2012; Boge & Schreutelkamp, 2002). The TRON-DLR facility focuses on hazard avoidance and optical navigation testing for autonomous spacecraft (Krüger & Theil, 2010; Woicke, 2019). GRALS, an ESA-developed testbed, is dedicated to spacecraft proximity operations and has been extensively employed to simulate and validate mission scenarios, including binary asteroid interactions[2]. Finally, the TRON-Stanford facility, part of the Space Rendezvous Laboratory at Stanford University, focuses on spacecraft navigation and rendezvous testing (Park, Bosse, & D'Amico, 2021).

European Proximity Operations Simulator 2.0

The EPOS facility is a simulation testbed developed jointly by Deutsche Forschungsanstalt für Luft- und Raumfahrt (DLR) and ESA focused on manoeuvres for the last few critical meters of the rendezvous phase. It has been extensively used for testing the Automated Transfer Vehicle (ATV) and H-II Transfer Vehicle (HTV) sensors used for approaching the International Space Station (ISS). The test facility has been updated to 6 dof for relative dynamic motion for approaching phase of 25 m to 0 m, contact dynamics behaviour during docking and space representative lighting and background conditions (Boge et al., 2012). This facility has been used for closed-loop rendezvous scenarios based on camera sensors, thus allowing for real-time hardware in the loop simulations. The setup will be valuable for creating an equivalent open-loop rendezvous scenario using the camera sensors for this thesis. An overview of the EPOS layout is shown in Figure 2.5.

[2] GRALS Testbed https://technology.esa.int/lab/gnc-aocs-pointing-laboratory, https://www.esa.int/ESA_Multimedia/Images/2018/05/GRALS_Testbed, and https://www.esa.int/ESA_Multimedia/Images/2018/07/GRALS_binary_asteroid_testing (accessed: 8 May 2025)

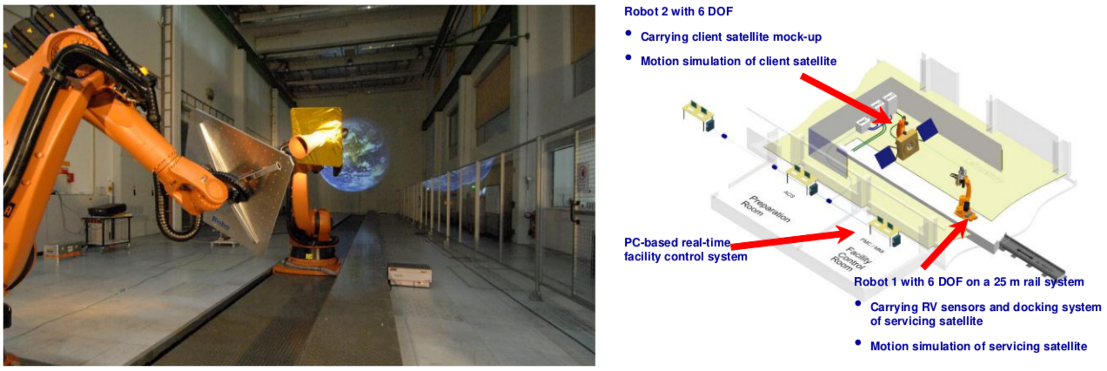

Robot 2 with 6 DOF
Carrying client satellite mock-up
Motion simulation of client satellite
Preparation Room
Facility Control Room
PC-based real-time facility control system
Robot 1 with 6 DOF on a 25 m rail system
Carrying RV sensors and docking system of servicing satellite
Motion simulation of servicing satellite

Figure 2.5: EPOS 2.0 facility with robotic-testbed (left) hardware overview (right) (Boge et al., 2012)

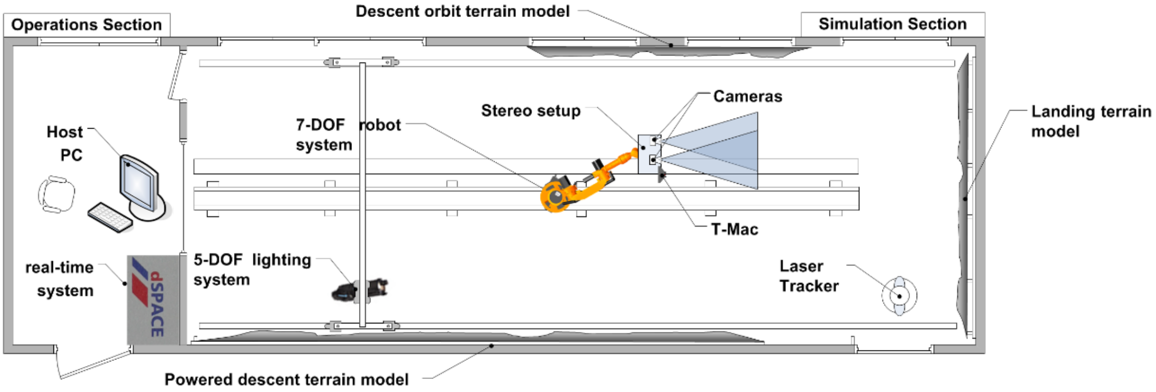


Figure 2.6: TRON-DLR hardware overview (Woicke, 2019)

Testbed for Robotic Optical Navigation at DLR

The TRON-DLR facility, developed by the German Aerospace Center (DLR), serves as a test environment for qualifying breadboard systems up to Technology Readiness Level (TRL) 4 and flight models up to TRL 5-6. It is designed to support optical navigation experiments involving active and passive sensors, such as lidar and cameras, under representative space conditions. The simulation framework integrates three main components: scaled dynamics using a 7-dof robotic arm, optical environment control through blackout conditions and dynamic lighting, and terrain geometry modelling for planetary landing scenarios (Krüger & Theil, 2010).

A rail-mounted robot enables dynamic positioning of sensor payloads, while a real-time system (dSPACE) ensures synchronised triggering and data collection across all sensors and ground-truth measurement devices. High-precision tracking is achieved using a laser tracker system measuring the robot's pose at 100 Hz. The laboratory also features modular terrain models mimicking Lunar and Martian surfaces, including descent and landing configurations. Surface topography is generated through 3D scanning and post-processing, with elevation variations up to 0.26 m across a 4.2 m × 2.2 m area. The robot moves longitudinally from 11,m to 1,m above the terrain with lateral displacement up to 1 m, supporting versatile trajectory profiles and observation geometries (Woicke, 2019).

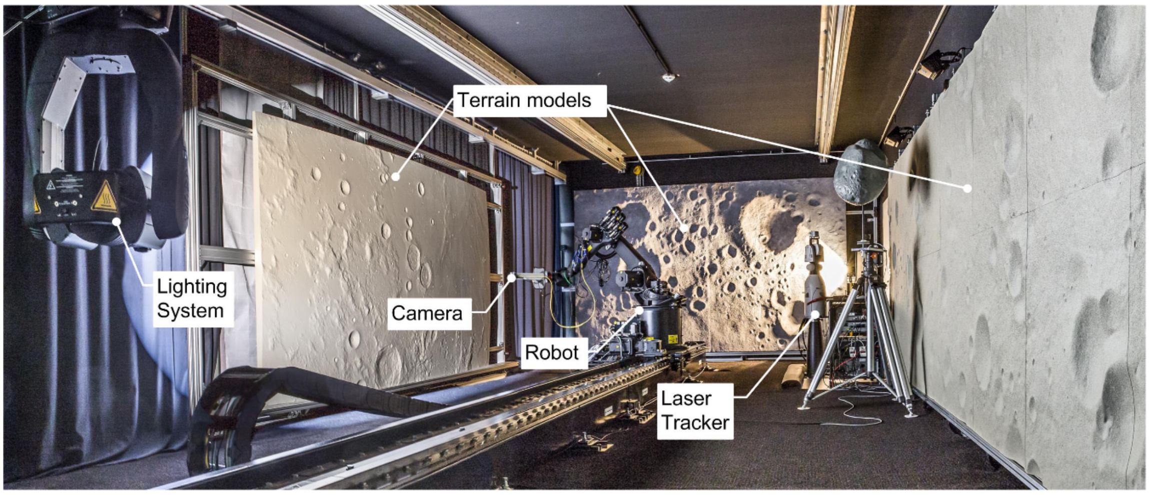


Figure 2.7: TRON-DLR Layout inside the laboratory (Woicke, 2019)

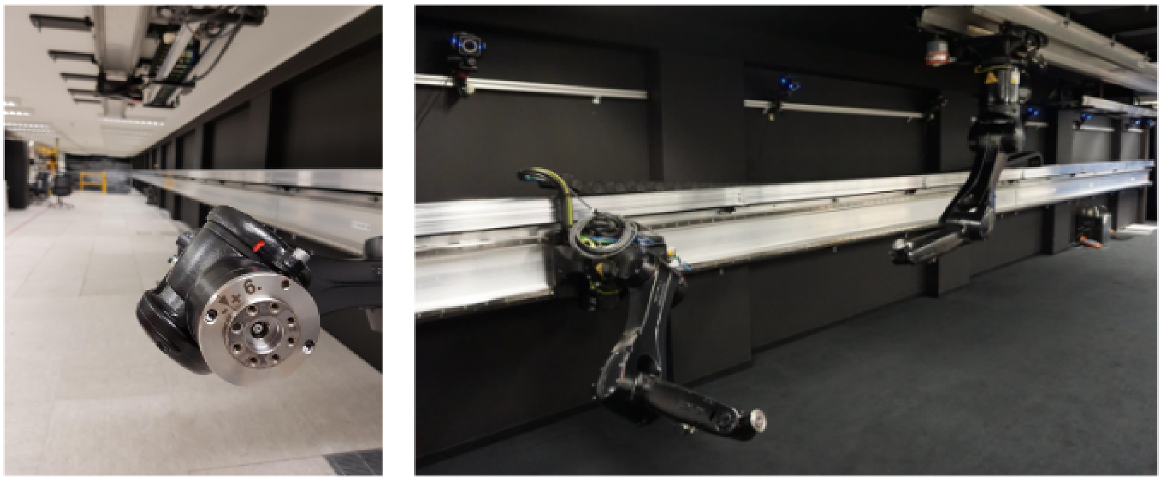


Figure 2.8: GRALS Wall robotic arm (left) Ceiling robotic arm (right) (Cattani, 2022)

GNC Rendezvous, Approach, and Landing Simulator

The GRALS facility, developed by ESA, is a hardware-in-the-loop testbed designed for the validation of visual and relative navigation algorithms under dynamic conditions. It primarily supports scenarios involving spacecraft rendezvous and landing. The setup includes a 33 m wall rail and a 5.6 m ceiling rail, each equipped with 1.1 m robotic arms (Figure 2.8). The environment is configurable through darkening elements and targeted illumination sources to replicate space like optical conditions.

A high fidelity ground truth system is provided by a 44-camera Vicon Tracking System, which enables accurate pose estimation of targets. Calibration and reference frame alignment are performed using a Vicon Wand. The robotic manipulators are mechanically integrated with the Vicon system to support synchronised motion tracking. For representative target emulation, the system includes a 1:35 scale model of Envisat, along with a laser rangefinder mounted via custom mechanical interfaces that ensure precise alignment with the simulated sensor platform (Cattani, 2022).

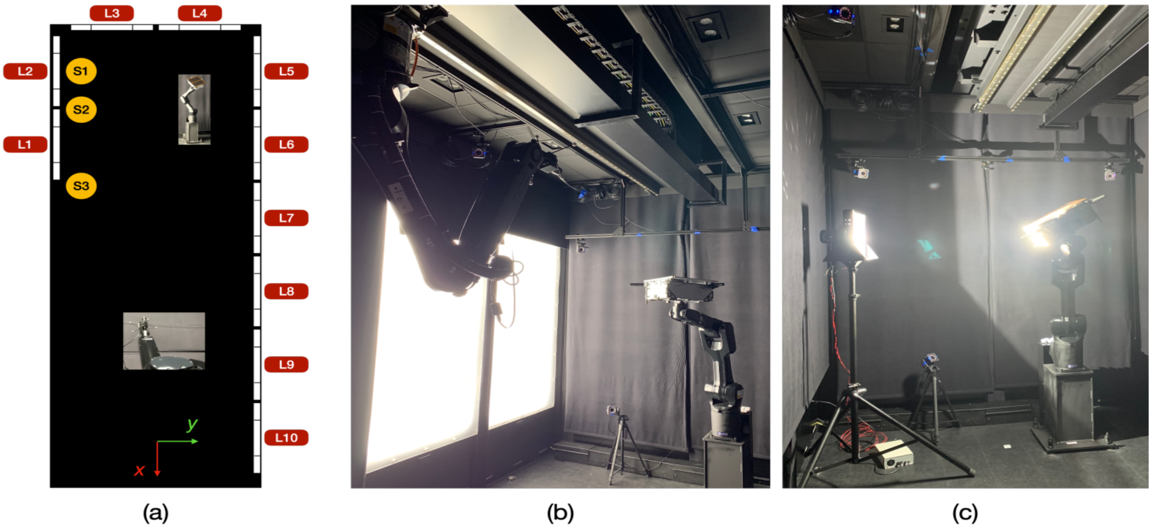


Figure 2.9: (a) TRON-Stanford facility layout with ten light boxes (L1-L10) and three sun lamp positions (S1-S3). (b) Spacecraft model illuminated by L1 and L2. (c) Model illuminated by the sun lamp at S1. (Park, Bosse, & D'Amico, 2021) (Park, Martens, et al., 2021)

Robotic Testbed for Rendezvous and Optical Navigation at Stanford

The TRON-Stanford facility at Stanford University[3] is the first robotic testbed designed for validating machine learning algorithms in the context of space optical navigation. It features two 6-dof KUKA robotic arms and a 12-camera Vicon motion tracking system for simulating and observing relative pose dynamics between a spacecraft-mounted camera and target mock-ups. One robotic arm, mounted to the ceiling, emulates the spacecraft's camera trajectory, while the other, fixed to the ground, holds the target structure. These elements operate within an 8 m × 3 m × 3 m simulation environment adjacent to a dedicated control room as shown in Figure 2.9 (Park, Bosse, & D'Amico, 2021).

To replicate the optical conditions of Low Earth Orbit (LEO), the facility includes ten calibrated light boxes with adjustable LEDs that provide diffuse Earth albedo-like illumination. Additional sun lamps enable direct lighting scenarios to complement the visual environment.

Real-time pose estimation is achieved by tracking infrared markers with the Vicon system, while telemetry from the KUKA arms provides joint-level feedback for calibration and verification. The testbed uses multiple reference frames to define camera, target, robot end-effector, and global coordinates, ensuring accurate alignment between physical motion and simulated orbital geometry.

Implications of Mission Heritage for Testbed Design

Establishing requirements for developing a robotic testbed for relative asteroid navigation presents a fundamental challenge due to the intrinsic link between mission architecture and system design. The resulting framework must balance mission feature modularity with hardware accuracy limitations. Accordingly, this study investigates both aspects.

This chapter has reviewed mission heritage and laboratory infrastructure to define representative manoeuvres and derive relevant test conditions. Across all missions considered, proximity operations begin with an initial global characterisation phase, typically executed from orbital distances of 100-300 km. These phases rely on circular or quasi-circular trajectories and use

[3] TRON Stanford University `https://slab.stanford.edu/` (accessed: 8 May 2025)

optical landmark tracking to derive asteroid shape and motion parameters. These findings directly inform the definition of experimental trajectories, which are modelled and evaluated in Chapter 5.

The primary objective of this work is to contribute towards developing a kinematic simulator focused on commanding the robotic system to reach absolute positions within the GNC laboratory. To this end, a reference frame framework and a software infrastructure to synchronise control inputs to the rover base and the robotic arm are required. While circular and arc-like trajectories were selected as a baseline operational profile, experimental results revealed undocumented performance limits in the combined system. The lack of existing documentation required this performance characterisation to be conducted as part of the research to define the testbed's performance.

In addition to trajectory planning, the review also identified the fidelity of surface models as a driving requirement for optical navigation. In response, Chapter 6 evaluates the capability of the GNC laboratory to generate high-resolution surface reconstructions of the asteroid model. These experiments assess the viability of the laboratory setup in supporting landmark-based relative asteroid navigation and further validate the relevance of the proposed testbed for the initial global characterisation phases part of an asteroid mission.

The synthesis of mission heritage and laboratory capabilities presented in this chapter establishes the top-level objectives driving the design of the robotic framework. These are formally refined in Section 7.2, which provides a structured reference for ongoing and future development of the GNC laboratory as a platform for experimental asteroid navigation research.

3 The Guidance, Navigation, and Control Robotics Lab

3.1. Overview GNC facility

The faculty of Aerospace Engineering of Delft University of Technology is also taking part in developing laboratories aimed to simulate and test different space missions on ground as described Section 2.2. With a long history of applied research for orbital dynamics for satellite applications tracking and monitoring celestial bodies the Satellite Communications and Orbital Mechanics (SCOM) laboratory[1] provided the test environment for multiple astrodynamics applications. The laboratory has contributed to a better understanding of space and dynamics by tracking satellites, aircraft, meteorites, and other celestial bodies. Since 2024, the department has decided to expand the capabilities of simulating space dynamics by setting up the Guidance, Navigation, and Control Robotics facility. The laboratory has to be developed to simulate spacecraft motion using rovers and robotics arms and investigating how this can be achieved represents the main focus of this thesis work.

The GNC Lab has been officially inaugurated in March 2025[2]. The facility is still under development. Several hardware components have been recently purchased; however, the facility does not yet have a framework for control and operation. This thesis project aims to provide a methodology to create trajectories within the laboratory that can be executed by one of the available robots. The other robot will be used as a mounting point for the target asteroid and if desired the mount can be rotated to replicate the spin motion of the target body. Creating this framework will expand on the possibilities of testing asteroid relative navigation. In addition, due to the novelty of the lab, one of the objectives of this thesis work was to understand what type of movement and with what accuracy can be produced in the laboratory. For this purpose, it was important to create an overview of the facility and the equipment and understand the limitations and potential tests that could be conducted.

The space is divided into two main areas: the experimental zone, also referred to as the dark room, and the control room for the robotic system and the OptiTrack. Blackout curtains separate these two areas to minimize external lighting disturbances and ensure controlled lighting conditions during the experiments. Moreover, the lab's illumination conditions replicate the space's dark environment through the use of black paint on the floor, walls, and ceiling. The experimental area dimensions are approximately 3.9 x 8.7 m. Figure 3.1 provides a detailed

[1] SCOM laboratory https://www.tudelft.nl/lr/organisatie/afdelingen/space-engineering/facilities/satellite-communication-and-orbital-mechanics-laboratory (accessed: 17 April 2025)

[2] Inauguration GNC laboratory TU Delft https://www.tudelft.nl/en/2025/lr/tu-delft-launches-four-new-and-upgraded-space-engineering-labs (accessed: 17 April 2025)

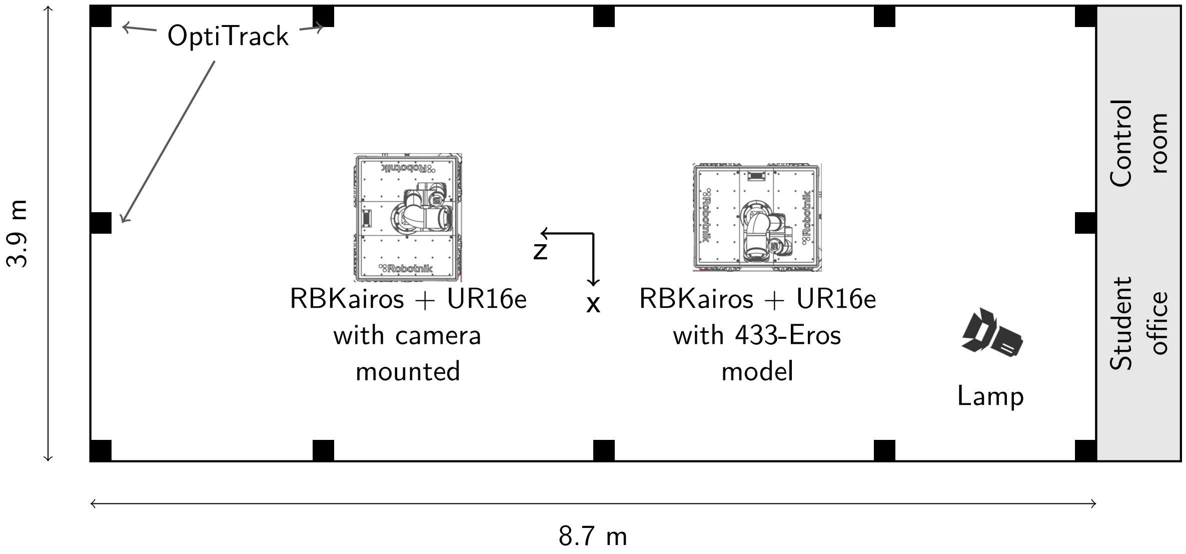


Figure 3.1: Floor plan of the Space Robotic Lab

floor plan of the lab, highlighting the layout of the experiment area and the location of key equipment, such as the OptiTrack cameras, the mobile Rb-Kairos+ rover base system, and the stationary Rb-Kairos+ used to mount the 3D-printed asteroid model.

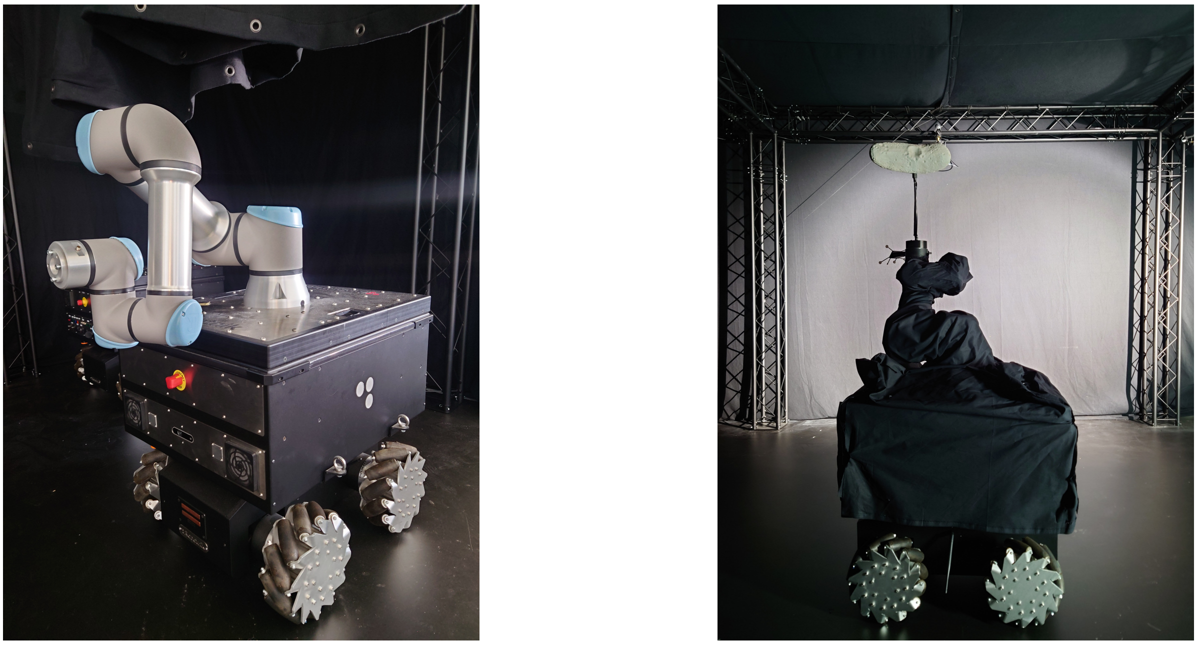
3.2. RB-Kairos+ Rover Base

RB-Kairos+ is an omnidirectional mobile manipulator that can accommodate a six dof UR16e robotic arm (see Figure 3.2a)[3]. The rover base, in combination with the 6 dof UR16e robotic arm, can be used to replicate the movement of the spacecraft. The model asteroid can be observed by mounting a camera on the end-effector of the robotic arm (Figure 3.8). The robotic system is designed in particular for indoor applications.

The RB-Kairos+ platform has been preconfigured by the Information Communication and Technology (ICT) department at TU Delft to join the university's secure network automatically. Access to the robot's onboard codebase is established via a secure shell (SSH) connection from any TU Delft workstation connected to that same network. This arrangement permits bidirectional communication between the control computer and the rover base. However, the system currently does not support true real-time command updates. Specifically, the control logic accepts movement commands only at the programme's launch. Once running, it cannot adapt to new environmental inputs or correct trajectories mid-execution. Although the robot can publish status messages and Robot Operating System (ROS) topics, the control logic of the robot, as of now, is only accepting control commands at the start of the program. Thus, once the program is started, it is not possible to adapt the movement to new elements present in the environment or to create corrections.

When integrating the rover base into the laboratory environment, two principal constraints must be addressed: the control command frequency of the mobile base and its occupied footprint. Unfortunately, the manufacturer has not formally documented the base's control command update rate. Consequently, the first objective was to determine these limitations empirically. For this application, the rover base is expected to move significantly slower than the robotic arm, which will perform the fine motion necessary to reach precise target points. Any movement of the base inherently introduces perturbations in the arm's trajectory, dimin-

[3] RB-KAIROS+ https://robotnik.eu/products/mobile-manipulators/rb-kairos/ (accessed: 17 April 2025)



(a) Robot 2: RB-Kairos rover base equipped with a UR16e robotic arm oriented with the tool flange perpendicular to the ground, enabling attachment of a camera.

(b) Robot 1: RB-Kairos rover base with a UR16e robotic arm oriented with the tool flange is parallel to the ground, enabling attachment of a 3D-printed asteroid model.

Figure 3.2: Mobile robotic platforms in the GNC Robotics Laboratory

ishing the smoothness of the performed trajectory. Therefore, control command frequencies lower than 0.5 Hz (i.e., one command every two seconds) will suffice to recreate desired trajectories within the laboratory workspace. In practice, this implies that the base should require at least two seconds to translate or rotate from its current pose before a new command is issued. For experiments conducted in this thesis, relative displacement commands have been defined in the range of 5 - 7 cm per command. This approach ensures that movement is discretised in small segments that can be tracked throughout the trajectory, while staying well below the maximum velocity constraints of the rover base.

Although the rover base can achieve a maximum velocity of 1.5 m/s (Auguste & Lumière, 2023), such speeds are excessive given the limited dimensions of the laboratory. At top speed, the base could traverse the laboratory's width in approximately two seconds and its length in roughly five seconds, rates deemed unsafe due to the presence of other equipment. The rover base is equipped with a mechanical safety button visible in Figure 3.2a, which, once pressed, mechanically cuts off the power supply to the servos of the rover base. This should be used with caution as a last resort to interrupt the movement of the rover. Hence, these extreme pose changes will be avoided. Instead, operations will be restricted to lower velocities, allowing controlled, incremental movements.

The physical dimensions of the rover base present in the laboratory are 760 mm (length) × 633 mm (width) × 690 mm (height)[4]. When the UR16e arm is mounted, the overall height increases to 1,410 mm. This volumetric envelope of the rover base, together with the dimensions of the laboratory, impose design constraints on the movement of the base:

[4] RB-Kairos datasheet hhttps://robotnik.eu/products/mobile-robots/rb-kairos-2/#rb-kairos-plus-ur-sixteen (accessed: 17 April 2025)

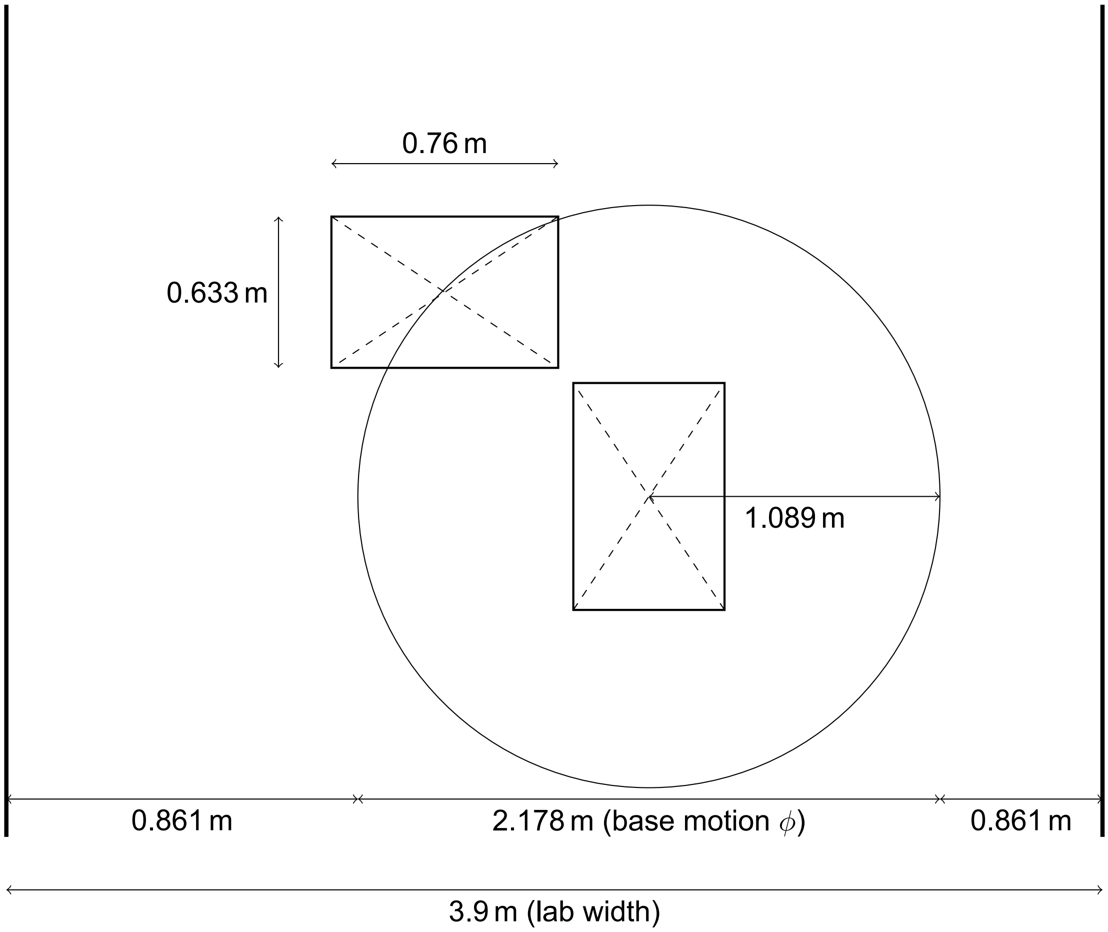


Figure 3.3: Safety constraints for the movement of the robotic base within the space of the laboratory

REQ-SYS-10 The chaser RB-Karios rover base shall crate revolutions around the target RB-Karios rover base with orbital distances between 1.089 m and 1.481 m.

Rationale: To avoid collisions between the rotating base and stationary centrally placed target rover base on which the asteroid model is placed, a minimum distance must be established between the rover bases as one moves around the other. This value is obtained by taking the worst-case scenario, when the rover base is aligned as shown in Figure 3.3. Due to the limited space of the laboratory, the stationary target base needs to be placed centrally within the laboratory space, with the length of the rover base perfectly aligned with the length of the laboratory. The diagonal of each rover base rounds up to 495 mm. To avoid the situation where the minimum orbital radius can be achieved but the edges of the rover base would touch, a safety margin of 10% is applied to ensure sufficient clearance. To ensure that the motion of the rover base still fits within the space available, the maximum radius that can be created within the space of the laboratory for a complete orbit should not exceed 1.481 m

3.3. UR16e Robotic Arm

UR16e is a 6dof robotic arm from Universal Robots able to carry a payload of up to 5 kg and has a reach of 900 mm. Regarding communication for control application, the maximum achievable control frequency is 500 Hz (Brandt, 2024). The datasheet provided by the manufacturer also provides information on the movement capabilities for each joint and details on the repeatability accuracy of the poses produced by the robot arm as summarised in Table 3.1. An overview of each join is shown in Figure 3.4

The control architecture employed by the robotic system for the UR16e bypasses the native URScript programming environment. Instead, low-level joint control is directly implemented to operate the robotic arm from the rover base computer. This implementation was facilitated by faculty staff member Alex Caon, who reconfigured the startup routine, enabling the UR16e to establish an automatic connection with the rover base computer upon power-up. The automatic connection eliminates the need for manual interaction with the Teach Pendant. The decision was motivated by safety concerns: access to the Teach Pendant interface requires physical connection via HDMI and USB, which places the operator within the manipulator's reachable space, which could result in unintended contact or injury if the system were to activate unexpectedly. Operation and programming of the robotic arm would then have to be performed inside the dark room.

Despite this setup, it has been observed that the robotic arm frequently enters a protective stop upon startup. In standard cases, this can be resolved remotely via SSH using the ROS service call:

```
rosservice call /ur_hardware_interface/dashboard/unlock_protective_stop
```

However, if the protective stop is triggered due to an impending self-collision, typically resulting from erroneous commands this service call is insufficient. In such cases, manual intervention via the Teach Pendant interface is required. The recovery procedure is outlined as follows:

1. Power down the robotic arm computer.
2. Connect a monitor via HDMI and USB mouse to the designated ports on the rover base.
3. Restart the robotic arm computer to initialise the drivers necessary for external display and input devices.
4. Upon entering the Teach Pendant interface, switch from *Remote Control* to *Local Control* in the upper right-hand corner.
5. Reinitialise the robot via the local control interface.
6. Manually move the robot arm into a safe configuration.
7. Confirm that the control session is active and functioning.
8. Switch back to *Remote Control*.
9. Disconnect the HDMI and USB devices and power down the robotic arm computer.
10. Restart the RB-Kairos system to resume standard operation.

These events became apparent during attempts to execute waypoint trajectories within the laboratory environment. It was discovered that the robot did not replicate the expected motion generated in the simulated environment. Upon inspection, it was confirmed that the robotic arm was mounted with a reversed reference frame. Specifically, the positive x and y axes of the UR16e are oriented oppositely to those of the robot base. This inconsistency has been compensated for in the software by adjusting the transformation matrices used to define the reference frame of the arm mount with respect to the base rover. Nonetheless, it is ***strongly recommended that the robot be remounted such that both coordinate frames align***, thereby simplifying future development and reducing the likelihood of command misinterpretation.

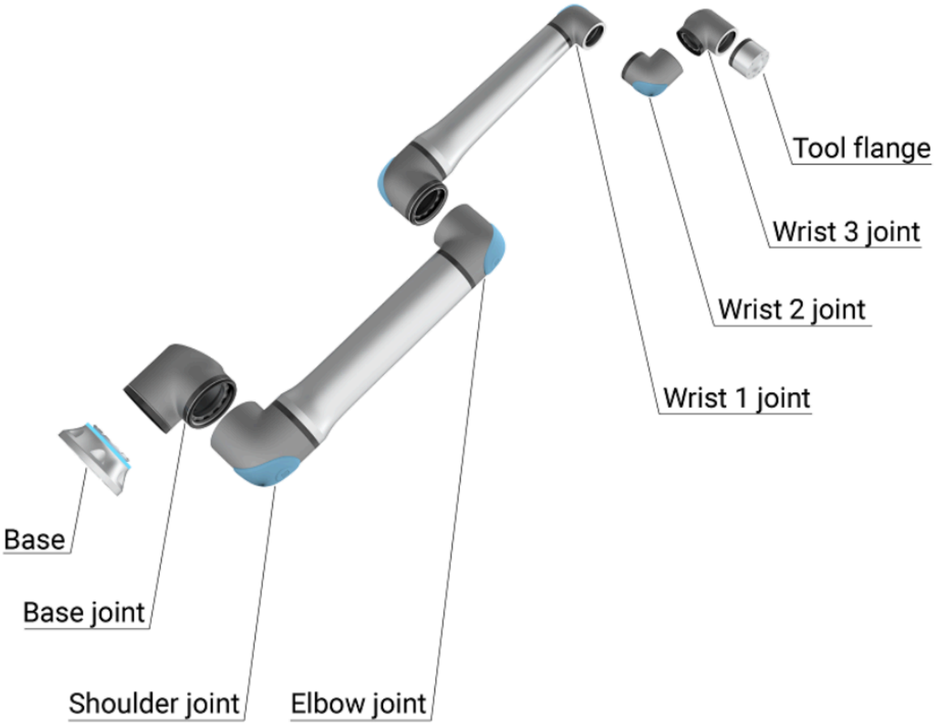


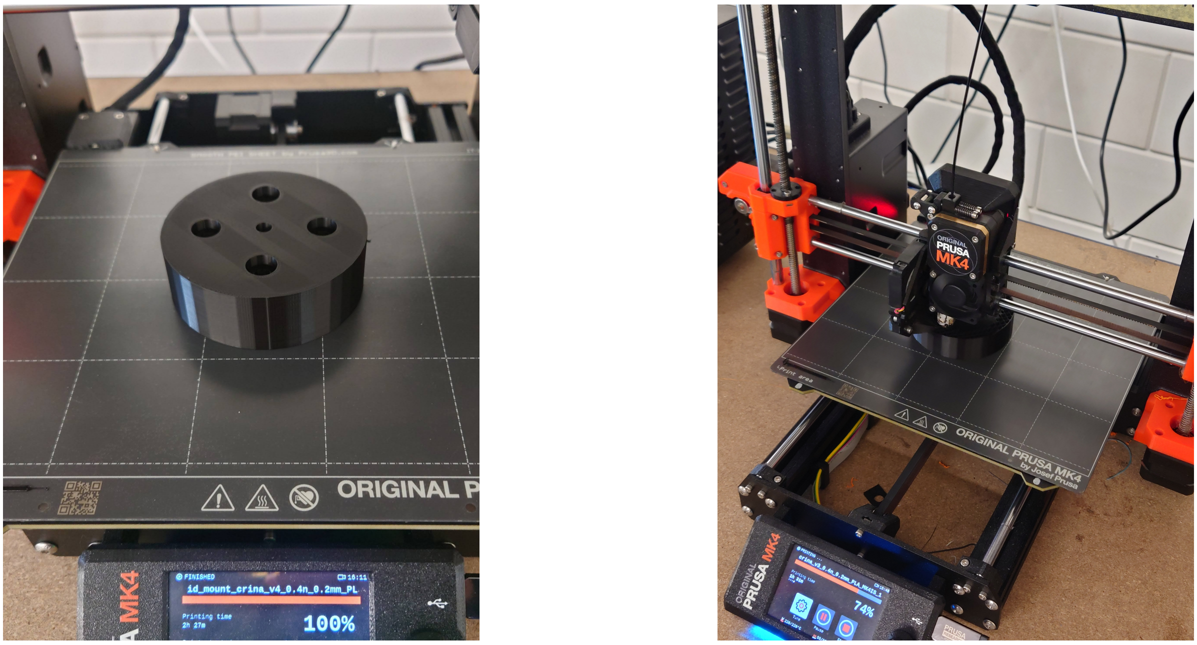
Figure 3.4: Join Definition for the UR16e robotic arm (Brandt, 2024)

Table 3.1: Movement specifications for the different joints of UR16e robotic arm (Brandt, 2024)

Property	**Value**
Typical tool speed	1 m/s
Pose Repeatability per ISO 9283	± 0.05 mm

Axis movement	**Working range**	**Maximum speed**
Base	± 360°	± 120°/s
Shoulder	± 360°	± 120°/s
Elbow	± 360°	± 180°/s
Wrist 1	± 360°	± 180°/s
Wrist 2	± 360°	± 180°/s
Wrist 3	± 360°	± 180°/s

A custom end-effector mount (see Figure 3.5a) has been developed to interface with the UR16e. The structure was fabricated using a Prusa MK4 3D printer (Figure 3.5b). This end-effector mounts directly to the standard UR flange, allowing modular integration with the payload, in the case of this work, the asteroid model. This design has the advantage that it can support the attachment of different models, provided they can be attached to an M4 rod, which is connected in the middle of the support. Further positioning of the model in the laboratory imposes constraints on the joint configuration of the robotic arm.


ORIGINAL P
id_mount_orina_v4_0.4n_0.2mm_PL
100%
MK4
ORIGINAL PRUSA MK4
ORIGINAL PRUSA MK4 by Josef Prusa
74%

(a) 3D printed interface connection between UR16e end effector and asteroid model

(b) Prusa MK4 3D Printer, used for creating the model mount

Figure 3.5: 3D printed model mount for the UR16e end effector

REQ-SYS-11 he stationary target rover (Robot 1) shall be preconfigured with joint angles of [3.080, –2.6279, 2.6279, –1.570796, 1.570796, 3.1415] rad for Base, Shoulder, Elbow, Wrist 1, Wrist 2, and Wrist 3 joints, respectively, when holding the Eros-433 model.

Rationale: The UR16e robotic arm must be placed in a safe joint configuration when configuring the target base to support the asteroid model. The joint values used are 3.080, -2.6279, 2.6279, -1.570796, 1.570796, and 3.1415 rad for the Base, Shoulder, Elbow, Wrist 1, Wrist 2 and Wrist 3, respectively. The corresponding values in degrees are 176, -150, 150, -90, 90, and 180° respectively. The precise pose configuration is illustrated in Figure 3.2b. This orientation has been selected as a tradeoff between lowering the target asteroid model and ensuring that the flanges of the robotic arm do not touch each other in the final joint configuration. Should that occur, despite the joint command being achievable, the internal safe mode of the robotic arm would detect a self-collision and disarm the joint control of the robotic arm to prevent damage to the robot. The model's height in the laboratory dictates the height at which the orbit around it has to be executed. The target's height may be reduced by up to 15 cm by lowering the size of the support rod of the asteroid model. Dimensions smaller than 15 cm for the support road pose the risk of the OptiTrack markers interfering with the view of the asteroid model, and it is not recommended.

REQ-SYS-12 The robotic system (Robot 2) shall be able to track trajectories with a minimum height of 162 cm in the space of the laboratory.

Rationale: The above-mentioned safe joint configuration of the joint angles imposes the minimum height at which the 433-Eros model can be placed. To recreate equatorial orbits around the 3D model, the robotic system shall be able to reach the minimum height to have the asteroid model centred in the field of view of the camera.

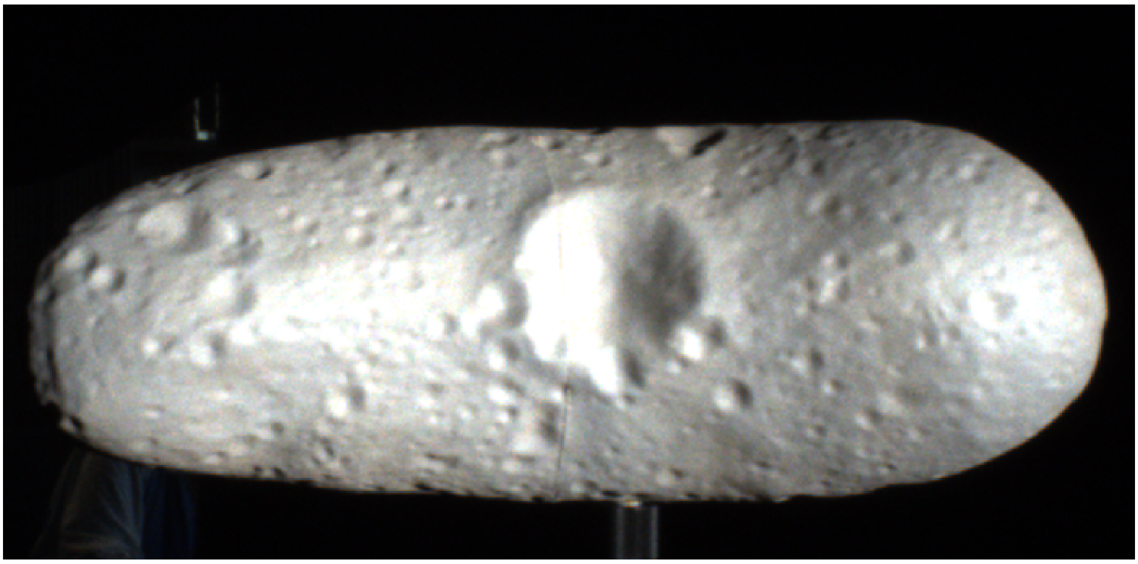


Figure 3.6: Image captured by Robot 2 of the 3D printed asteroid model of 433-Eros during experimental setup

3.4. 3D printed asteroid model 433-Eros

The 3D printed model of the asteroid shown in Figure 3.6 was created as part of a previous project within the department (Terwindt, 2025). This was achieved using the Formlabs 3L printer[5]. Building upon this work, where manual measurements were taken around a circular orbit around the asteroid model, this thesis aims to contribute towards creating automated trajectories using the robotic system, such that more visual measurements of the asteroid can be taken, in different orbits and orbit types.

The Formlabs 3L printer used for this task employs White V4 resin with a minimum layer thickness of 0.050 mm. Due to the build volume of the printer of 33.5 x 20 x 30 cm, the model of 433-Eros was cut into two parts, allowing it to be printed upright for better print quality. Internal supports were added to maintain structural integrity during printing and alignment pins were incorporated for assembly. This approach creates an artificial landmark on the surface of the asteroid model.

The surface features of the 3D-printed asteroid model will not be visible at a resolution higher than the print layer thickness, meaning that fine details may be lost or inaccurately represented in the physical model. This limitation must be accounted for when interpreting the experimental results, as the lack of high-resolution surface detail could affect the accuracy of landmark detection should the images be directly used by navigation filters. In addition, it is important to assess the implications of the printed asteroid models scale $10:10^6$. 1 cm to 1 km on the surface of 433-Eros, (Terwindt, 2025) for the types of trajectories that can be realistically replicated in the laboratory. Taking into account the scale of the asteroid model creates requirements on how far the camera must be placed from the asteroid model. Thus, for a 35 km orbit, the camera will have to be placed 35 cm away from the model. Similarly, for a 100 km orbit the camera will have to be 100 cm in the laboratory.

REQ-EROS-01 The generated laboratory trajectories have to be scaled by 10:1,000,000 (1 cm laboratory to 1 km orbit) to match the scale of the asteroid model.

Rationale: The laboratory dimensions also need to be accounted for. The most constraining dimension is the width of the laboratory, which is only 3.9 m. Applying again the scale of $10:10^6$

[5] Form 3L https://formlabs.com/3d-printers/form-3l/ (accessed: 25 May 2025)

to the laboratory dimensions, this means that 3 m in the laboratory space will be equivalent to 300 km in space. While the width of the laboratory is 3.9 m, additional safety margins must be accounted for. That is due to the presence of the truss structures on the sides of the room as well as the physical dimensions of the hardware, particularly the rover base (Section 3.2). The length of the rover base is 0.76 m, and it can constrain the points that can be reached to create the trajectory.

Thus, the operational requirement for creating a full orbit in laboratory space is a maximum of 3 m, corresponding to a 300 km orbit. It is important to note that these requirements pertain to the scale of the asteroid model used in this work, and should a different scale be selected for the asteroid model, the type of trajectories represented in the laboratory can also change. Orbital trajectories at scales larger than 300 km can be possible, provided that the scale of the asteroid also needs to be accounted for. A larger orbital radius would significantly reduce the size of the 433-Eros asteroid model from cm to mm. The printing and fabrication of the model become impossible. For larger celestial bodies, the range can be extended beyond this mark. The scale can be increased to create even closer orbits, flybys or surface observations. For this different type of models, or perhaps only partial surface portions of the orbital bodies can be printed at a higher resolution, similarly to what has been performed in other ground based laboratories (Figure 2.7).

3.5. OptiTrack

An OptiTrack camera system will be used in this research to track the movement of the spacecraft robot and the asteroid model with high precision. This system captures 3D motion in real-time by tracking reflective markers placed on objects, enabling accurate monitoring of their position and orientation within the lab. For the purpose of this study, the OptiTrack system is ideal as it provides sub-millimetre accuracy, which is crucial for evaluating the performance of the hardware without interfering with the motion of the different elements.

The OptiTrack system, available is Prime13/Prime13W [6], which offers high frame-rate capture, precision tracking, and continuous calibration features. The cameras can track both passive and active markers, with a positional accuracy of less than +/- 0.30 mm and rotational errors of less than 0.5°.

The setup available in the laboratory consists of 12 cameras, which have been placed around the roof of the truss structure of the laboratory as shown in Figure 3.1. The manufacturer designs the camera system to provide 3D accuracy over a 9 m x 9 m. Thus, the laboratory's space fits within the limits set by the manufacturer at only 3.9 m x 8.7 m. The cameras have been positioned in the laboratory to cover the full space of the laboratory, as in order to provide tracking measurements for markers, they need to be in the field of view of at least three cameras at once. The FOV of each camera is adjustable, and it is given by the number of frames per second (fps) at which the user chooses to operate the system. As shown in Table 3.2, there is an inversely proportional relationship between the fps and the FOV that the camera can achieve. A lower rate of recorded frames per second has the advantage of a higher field of view and, thus, better coverage of the available space within the laboratory. However, it is not suitable for recording high-speed movement, so a higher frame rate is preferred. However, two considerations were relevant when selecting which frame rate is required for this experiment. Selecting a higher frame rate would significantly decrease the FOV of the cameras. Given that the ceiling mount of the optic cameras is only 3 m above the ground, this can create blind spots in the lab where the markers cannot be correctly observed.

[6] OptiTrack Prime 13/13W https://docs.optitrack.com/ (accessed: 25 May 2025)

Table 3.2: OptiTrack: Prime 13/13W[6] technical specification for various frame rate settings

Frame rate	Resolution	FOV (3.5mm lens)
240 fps	1280×1024	82°×70°
360 fps	1040×832	71°×59°
500 fps	880×704	62°×52°
1000 fps	624×496	46°×38°

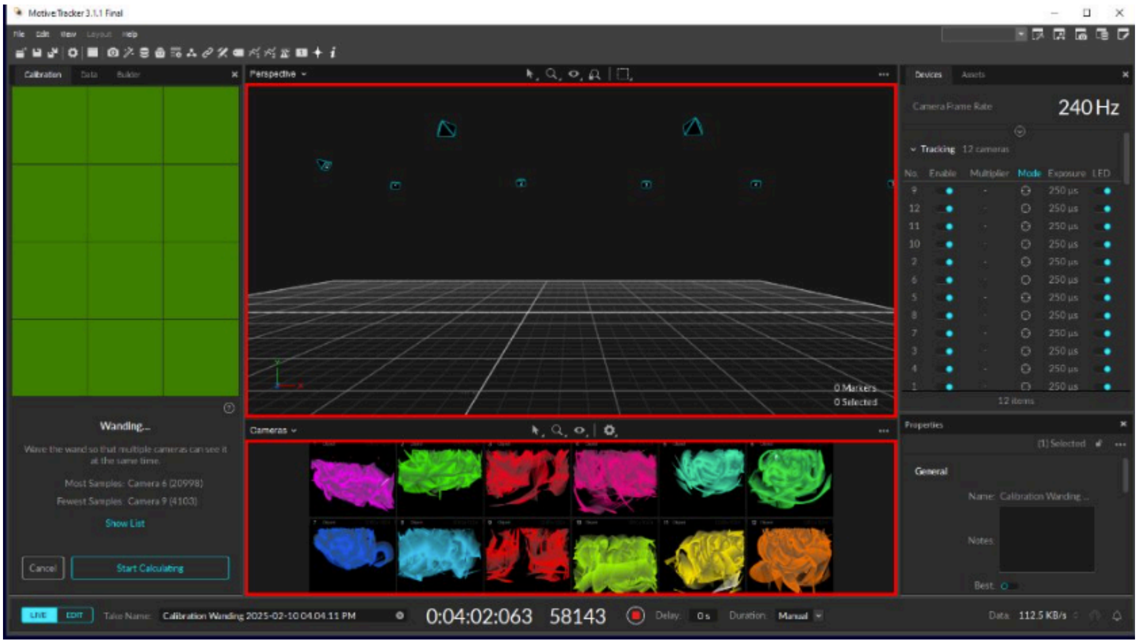
REQ-SYS-13 The motion-capture system shall use 240 fps for recording the motion profiles during experiments.

Rationale: Provided that the system will receive a command every 2 seconds, a 240 fps already provides 480 measurements for the duration the command needs to be executed. The robotic setup has not been designed to perform fast attitude modifications, and therefore, a higher frame rate will not contribute towards better operations. These capabilities are generally designed for more dynamic systems such as drones, where attitude change occurs far more rapidly, or even for monitoring human movement, where more subtle differences can impact the moving behaviour. The expected positional changes in this work are on the order of centimetres per second. At 240 fps, the resulting spatial resolution exceeds the system's own measurement accuracy, ensuring that each captured displacement is meaningful. Increasing the frame rate further would not enhance the quality of the data, as the measurements would then be dominated by the system's intrinsic positional accuracy of 0.3 mm, rather than the actual motion of the platform. Thus, for the scope of this work, a lower rate of 240 fps is selected such that the field of view of the OptiTrack system and the resulting resolution of the cameras are larger. This option allows for a larger coverage of the laboratory volume. For comparison, TRON-DLR is using 100 Hz sampling for pose measurements (Woicke, 2019)

Calibration and Data Recoding using OptiTrack

To ensure accurate tracking, the OptiTrack system needs to be set up and calibrated. Below is a step-by-step guide for the setup and calibration of the OptiTrack system:

1. **Open Motive:** The software needs to be initialised to start a new calibration
2. **Apply Masks:** When performing the calibration, it is important to remove all markers present in the room, as they can interfere with the calibration process. Once that has been ensured, the cameras may still detect certain lighted surfaces that may be interpreted as markers. These are shown to the user in the panel on the left-hand side, currently coloured green in Figure 3.7. Thus, it is important to apply masks on all the objects detected before starting the calibration process.
3. **Initial Calibration (Wand Wave):** Perform an initial calibration using the calibration wand. Move the wand in sweeping infinite motions throughout the tracking area, ensuring that all cameras detect the wand in various positions. This step establishes the spatial relationship between the cameras. For a good calibration, it is important to cover as much as possible from each camera's FOV, shown in the bottom panel, which coloured patterns (Figure 3.7). Once the FOV is sufficiently covered, the left-hand side panels turn green. It is then safe to press start calculating and continue with the calibration.
4. **Ground Plane Placement:** The ground plane tool shown in Figure 4.6 needs to be placed somewhere in the centre of the tracking volume to ensure it is in the FOV of as many cameras as possible. This point will represent the origin of the OptiTrack measurement points. This completes the calibration procedure.
5. **Marker Placement:** Attach passive or active markers to the asteroid model and spacecraft robot. Ensure that the markers are positioned in such a way that they remain visible to multiple cameras at all times.



Wanding...

Wave the wand so that multiple cameras can see it at the same time.

Most Samples: Camera 6 (20998)

Fewest Samples: Camera 9 (4103)

Camera Frame Rate 240 Hz

Tracking 12 cameras

No.	Enable	Multiplier	Mode	Exposure	LED
9	●	-	○	250 µs	●
12	●	-	○	250 µs	●
11	●	-	○	250 µs	●
10	●	-	○	250 µs	●
2	●	-	○	250 µs	●
6	●	-	○	250 µs	●
5	●	-	○	250 µs	●
8	●	-	○	250 µs	●
7	●	-	○	250 µs	●
3	●	-	○	250 µs	●
4	●	-	○	250 µs	●
1	●	-	○	250 µs	●

12 items

Name: Calibration Wanding ...

Notes:

Best: ○

Take Name: Calibration Wanding 2025-02-10 04.04.11 PM

0:04:02:063 58143

Delay: 0 s Duration: Manual

Data: 112.5 KB/s

Figure 3.7: Calibration panel of the OptiTrack system

6. **Create Rigid Bodies:** The markers will now appear in the middle panel shown in Figure 3.7. Each marker group corresponds to an object being tracked. Multiple markers can be selected with the mouse, and then by right-clicking, select Create Rigid Body. With this method, each object will only have one measurement point in terms of position and rotation in the X, Y, and Z axes.
7. **Record Data:** With the rigid bodies to be tracked by markers, it is now possible to record data. This can be done using the bottom ribbon of Motive shown in Figure 3.7. It is crucial to ensure that Live mode is selected. Edit mode can be used to review older measurements and their movement within the tracking volume. The name of the file can be modified as desired. The operator can press play (green arrow) to start recording and stop (red square) to stop the measurement. Information on the elapsed time since the start of the measurement and the number of recorded points will be provided in the ribbon in real time.
8. **Export Tracking Data:** Switching from the calibration tab to the data tab in the left-hand side panel shows the list of all measurements taken. By selecting any measurement of interest, the tracking data can then be exported in CSV format.

For detailed tutorials on calibration, synchronization, and optimal marker placement, the official OptiTrack documentation available online[6]. This documentation includes guidance on various applications, such as rigid body tracking, as well as examples of correct camera placement for optimal accuracy.

3.6. Asteroid Tracking Camera

The asteroid tracking camera is a critical component of the experimental setup. The camera used in the laboratory is the MER2-160-227U3C[7], which features a resolution of 1440 × 1080 pixels and supports frame rates up to 227 fps. These specifications make it suitable for capturing high-resolution images in high-speed dynamic scenarios.

As discussed in Section 3.4, both the scale of the asteroid model and the available labo-

[7] MER2-160-227U3C https://www.vision-camera.nl/USB3.0-Camera-1.6MP-Color-Sony-IMX273-MER2-160-227U3C (accessed: 17 May 2025)

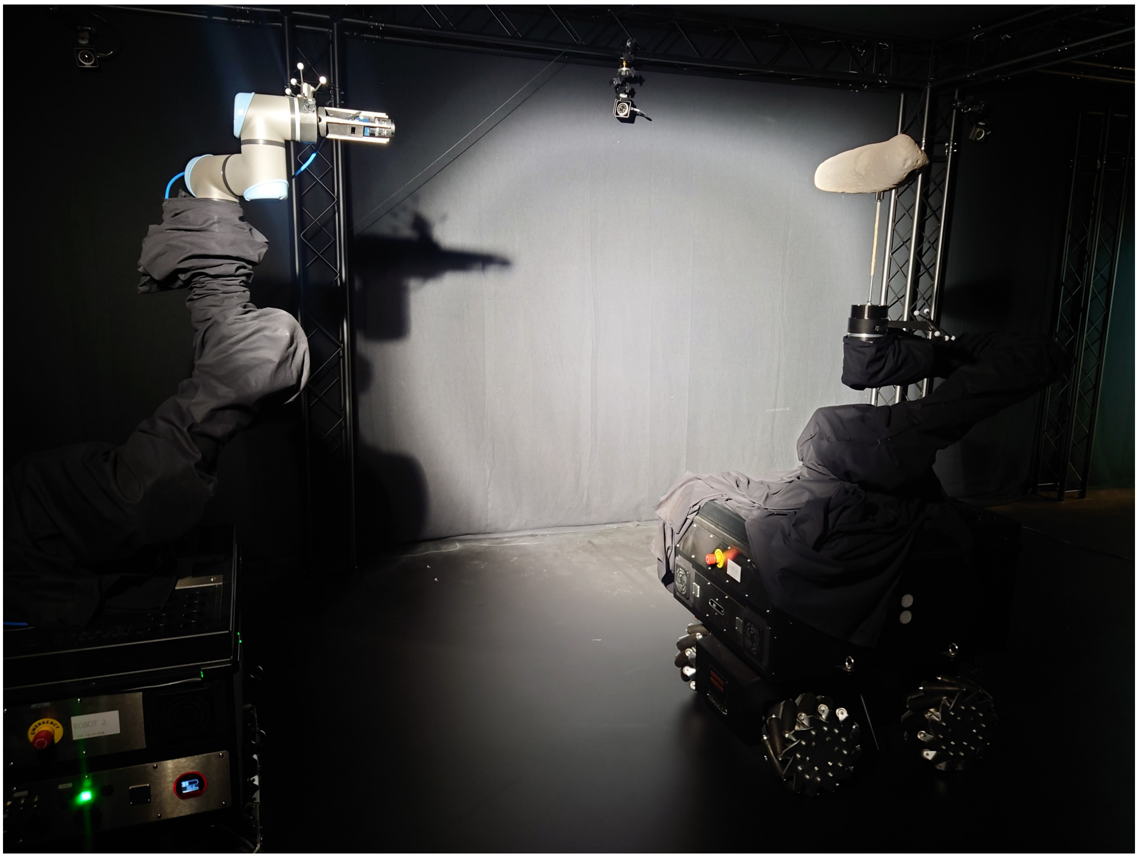


Figure 3.8: Robot 2 (left) with MER2 camera on its end-effector taking an image of Robot 1 (right) with the 3D model of 433-Eros. Illumination in the dark room is created using the KL FRESNEL 8 lamp.

ratory space constrain the camera-to-target distance required to obtain image measurements. The camera must be calibrated for each experimental scenario to ensure proper focus and sharp image quality. Inadequate calibration may result in blurred images, which can degrade downstream navigation performance.

In this work, where the focus lies primarily on generating trajectories using the robotic setup, optimal camera calibration was not performed. However, if the visual output were to be used for more advanced navigation filters, the camera settings would need to be adjusted for each experimental configuration. This is particularly important in landmark-based navigation, where image sharpness directly affects the accuracy of landmark detection and subsequent extraction of pixel coordinates. These coordinates are then transformed into real-world positions in the asteroid-fixed frame and used by the navigation module to estimate the spacecraft's relative state, its position and orientation with respect to the asteroid.

Further information on camera calibration procedures and experiments involving manual recreation of 30 km, 100 km, and 200 km orbits around asteroid 433-Eros can be found in the work of Terwindt (2025).

The camera is connected to the end effector of the robotic arm on Robot 2 such that it can take photos of the asteroid placed on Robot 1 as shown in Figure 3.8. The command lines required to operate the camera and the robotic system described in previous sections are summarised in Appendix A.

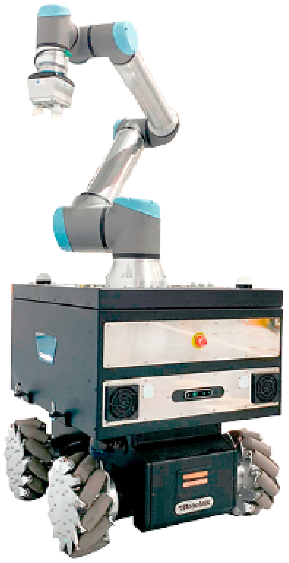


Figure 3.9: Simulation of an RB-Kairos equipped with a UR5 robotic arm and suction tool mounted on the end-effector - default configuration provided by the manufacturer for the simulation of the robot

3.7. Setting up a simulation for the laboratory

One important feature that most of the laboratories described in Section 2.2 are equipped with is simulation software for the hardware present and, in most cases, real-time communication between the control command and the hardware. As a newly developed laboratory, such features are not yet present in the GNC laboratory. Given that it is not yet possible to send direct control commands to the robotic system, it has been deemed essential to attempt to create a simulation of the commands before deploying the code on the physical hardware.

When it comes to robotic applications, there are multiple simulation software packages commonly used for visualising the behaviour of the rover, ranging from toolboxes such as the Robotics System Toolbox[8] of MATLAB to more realistic environments such as Unity, suitable for integration with virtual reality. The two most commonly used in robotics for 3D applications are RViz[9] and Gazebo[10]. Their popularity is because of their direct integration with ROS, which facilitates communication with topics, services, and messages. The two software have complementary roles, as RViz is instead tailored for sensor data and provides a visualisation of the system without considering the system's dynamics or interactions with the environment. Gazebo can overcome this shortcoming and simulate the physics of the system and the surrounding environment. Provided that the main objective of this work is to simulate the dynamic movement of the robotic system within the environment, being able to visualise potential interactions and attempt to avoid any collisions within the environment is of paramount importance. Thus, Gazebo is the most relevant candidate for this application.

The manufacturer provides an open-source GitHub with all the required packages and configurations files such that a simulated RB-Kairos can be visualised in an office space[11]. This simulation has been designed to run on Ubuntu 20.04 with ROS1 Noetic.

It is important to note that the default configuration of the system provided by the manu-

[8] Robotics System Toolbox https://www.mathworks.com/products/robotics.html (accessed: 11 Feb 2025)

[9] RViz http://wiki.ros.org/rviz (accessed: 11 Feb 2025)

[10] Gazebo https://gazebosim.org/home (accessed: 11 Feb 2025)

[11] Robotnik Automation Simulations https://github.com/RobotnikAutomation/rbkairos_sim (accessed: 11 Feb 2025)

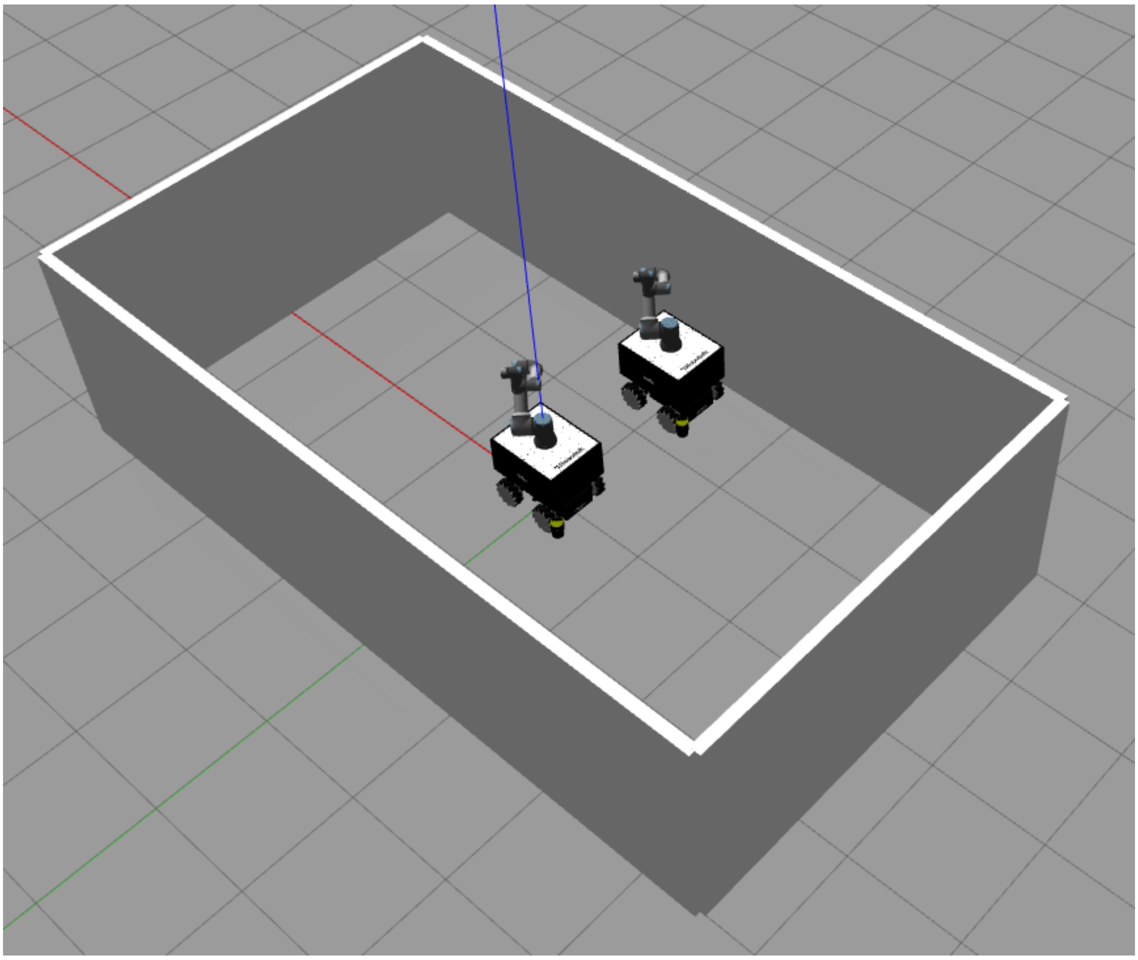


Figure 3.10: Gazebo simulation of the space available in the GNC laboratory with the two robots

facturer when launching the software is an RB-Kairos equipped with a UR5 robotic arm and EGH gripper, as shown in Figure 3.9. Other configurations are also available but need to be configured by the user. In the case of this work, it was required to modify the robotic arm from UR5 to UR16e, for which the Unified Robotics Description Format (URDF) files required to describe the geometry of the robotic arm can be readily found on the repository provided by the manufacturer. Thus, by changing the configuration files used to launch the system, the correct version of the robotic arm can be loaded onto the symultation. The griper present in the default configuration is not relevant for the purpose of this application, as the robotic system used has a camera mounted on its end-effector used for recording visual inputs. The URDF files of this camera are not available as the geometry of the camera is not modelled for the simulation environment. Similarly, for the second robot present within the space of the laboratory, the gripper is not mounted, but in place, a rod holding the 3D printed asteroid model shown in Figure 3.2b. Provided that the motion modelled for this application assumes at least 0.5 m distance between the camera and the asteroid model, it is not necessary to model the camera and the asteroid in the Gazebo simulation. Thus, the gripper will be removed from the simulation environment altogether.

The office space provided in the default configuration is much too complex to be representative of the laboratory space available presented in Section 3.1. Thus, the world configuration file had to be altered as well. For this purpose, a new wall layout has been created to replicate the space available for the robot to move. The layout follows the schematic shown in Figure 3.1.

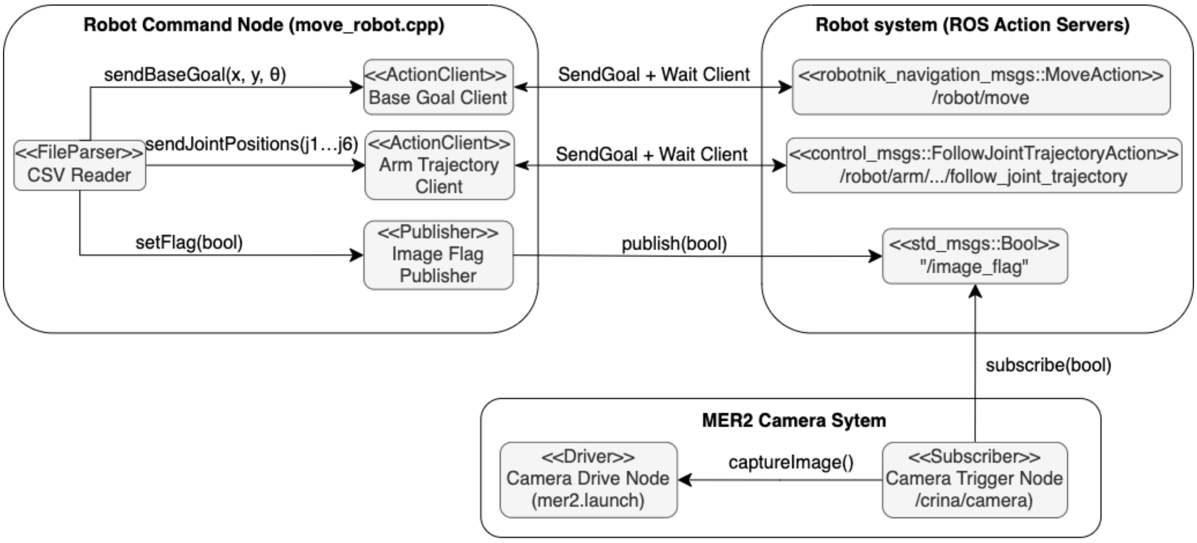

Robot Command Node (move_robot.cpp)
sendBaseGoal(x, y, θ)
<<ActionClient>> Base Goal Client
<<FileParser>> CSV Reader
sendJointPositions(j1…j6)
<<ActionClient>> Arm Trajectory Client
setFlag(bool)
<<Publisher>> Image Flag Publisher
SendGoal + Wait Client
SendGoal + Wait Client
publish(bool)
Robot system (ROS Action Servers)
<<robotnik_navigation_msgs::MoveAction>> /robot/move
<<control_msgs::FollowJointTrajectoryAction>> /robot/arm/.../follow_joint_trajectory
<<std_msgs::Bool>> "/image_flag"
subscribe(bool)
MER2 Camera Sytem
<<Driver>> Camera Drive Node (mer2.launch)
captureImage()
<<Subscriber>> Camera Trigger Node /crina/camera)

Figure 3.11: Architecture of the robot control and image acquisition pipeline.

Moreover, the new world environment will include a second instance of the RB-Kairos to be representative of the available free space. However, the second instance is not designed to accept control inputs and will be stationary during the simulation. This second instance represents the robotic system on which the 3D-printed asteroid model is positioned. If desired, in more advanced phases of the laboratory design, the simulation can be adapted such that the second robotics system also accepts commands. However, it is important to note that it would be more computationally demanding on the computer used for the simulation. Thus, a higher-performing GPU and a graphics card on the machine might become relevant for setting up and using the simulation. The resulting simulation with the correct hardware version and with the environment of the GNC lab is shown in Figure 3.10. Appendix B includes the commands for installing and building the simulation environment.

Another important aspect of this simulation is that the manufacturer does not provide readily available control nodes. Thus, those must be created using the topics available to communicate with the simulated rover. The control nodes in the simulation are created following a similar structure as the hardware, such that the robotic system can accept a control input file with commands for the rover base and the robotic arm. The connection between the different topics and action client used to read the control input file, send the goals to the base and the arm and take image measurements of the robotic arm is shown in Figure 3.11. The Robot Command Node reads input data from CSV files and communicates with the robot system via ROS action clients and publishers. It commands navigation and arm motion through the move and follow_joint_trajectory action servers, respectively, and controls image capture by publishing a Boolean flag. b nvb b The Camera System subscribes to this flag and triggers image acquisition accordingly.

The diagram shown in Figure 3.11 is meant to show the main components of the simulation and their input-output dependencies. For the Robot Command node, an additional activity diagram showcasing the chronological order of the commands being executed is presented in Figure 3.12.

The development of the digital twin simulation of the laboratory's robotic setup directly contributes to addressing the first research sub-question: how a robotic system can be designed and utilised to simulate navigation around a 3D-printed model of Eros which is further discussed in chapter Chapter 7.

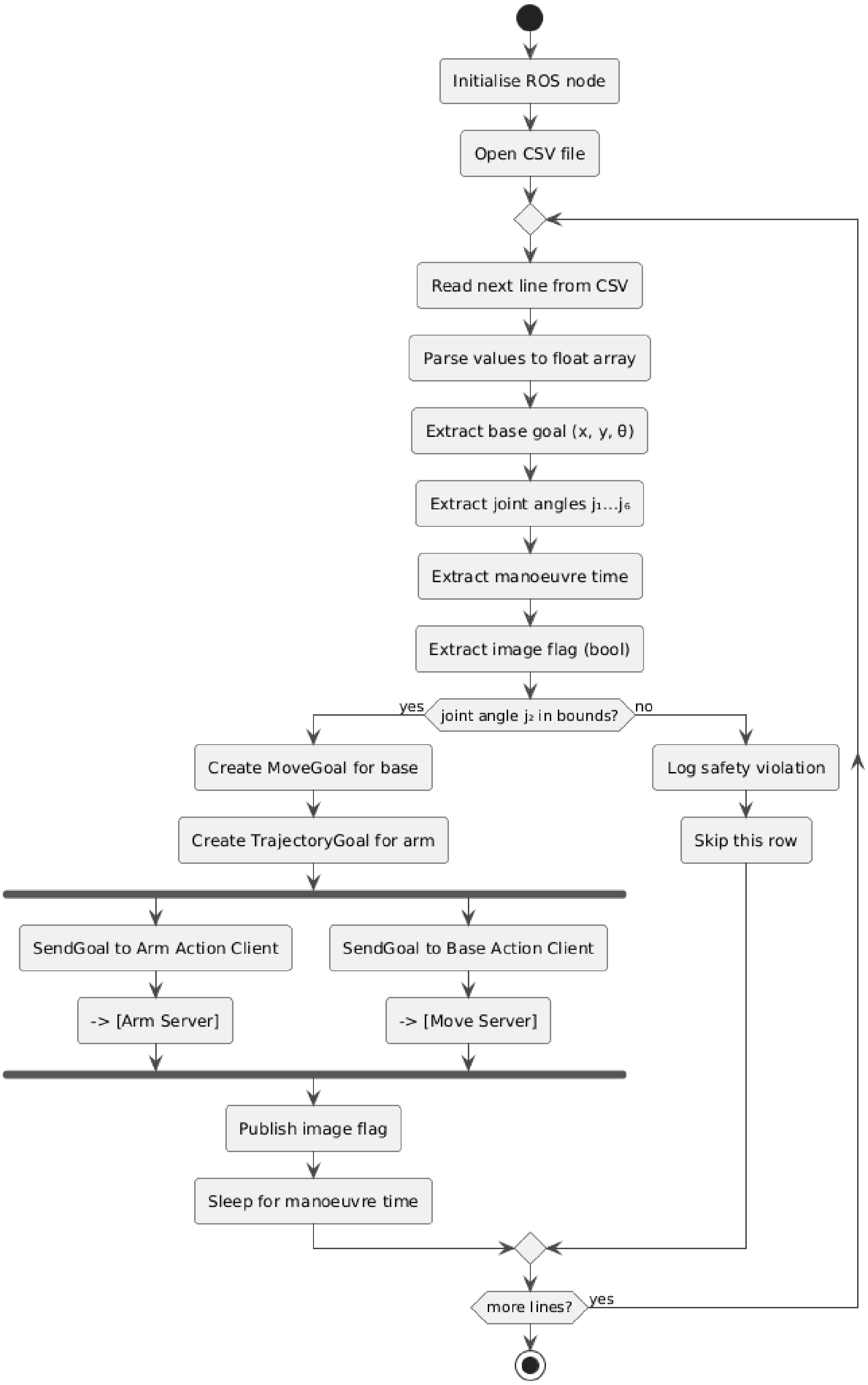


Figure 3.12: Flowchart of the robot command node execution logic.

4 Reference Frames

Having discussed the hardware components available for the robotic testbed, the next step is to define the reference frames required to describe and control the system's motion. The physical configuration of the testbed informs the selection of appropriate coordinate systems for modelling and reproducing the spacecraft's relative motion with respect to an asteroid, both in orbital simulation and within the laboratory environment. This chapter establishes the necessary framework to reproduce that motion in both scenarios. The chapter begins by evaluating the use of coordinate systems and which type is suitable for the motion required to recreate the relative motion as presented in Section 4.1. The spacecraft-asteroid reference frames and the required reference frames used to create the control inputs of the robotic system within the laboratory space are introduced in Section 4.2. Next, Section 4.3 provides the required background on frame transformations, and lastly, Section 4.4 gives an overview of the implementation of the reference frames further used for modelling the movement of the robotic system described in Chapter 5.

4.1. Coordinate Systems

The choice of the coordinate system is crucial in accurately expressing the position and orientation of the asteroid and spacecraft. This section evaluates various coordinate systems and their applicability in simulating the spacecraft's trajectory around the asteroid. While Cartesian coordinates are the most widely recognised due to their straightforward representation of three-dimensional space, alternative coordinate systems can sometimes provide better insight into the dynamics of celestial bodies or a more straightforward interpretation of a position on a surface. Transformations between these coordinate systems can also be considered to avoid potential singularities during numerical integration or to gain better insight for analysing results. However, in the selection process for this work, the main consideration would be to select coordinate systems that can provide a fair comparison between the spacecraft's motion around the asteroid and their laboratory replicas within the available space of the laboratory.

Translational State The translational state of a satellite orbiting an asteroid, such as its position and velocity, can be expressed in several coordinate systems, including Cartesian, Keplerian, Spherical, Modified Equinoctial, or Unified State Model elements. Each system has its strengths and weaknesses. However, Cartesian coordinates have the advantage of directly providing the position of a body in three-dimensional space using only three parameters as a direct result of the equations of motion being expressed in an inertial reference frame. Moreover, the laboratory where experiments will take place is equipped with a tracking system for which the measurements are produced in Cartesian space. This coordinate system allows for easier conversion of the trajectory provided by the simulation software into the operational space of the laboratory. However, while Cartesian coordinates suffice to describe a trajectory, camera-based navigation around an asteroid relies on identifying landmarks on the body's surface, which cannot be directly expressed in Cartesian coordinates. In this case, the spherical coordinates may be used, assuming that the centre of the body of interest is known and a point on the object's surface can be uniquely identified, thus allowing for the integration of the landmarks for navigation. Lastly, Keplerian Orbital elements are the fundamental method for describing the parameters of an orbit. This work focuses not on performing orbit simulation but rather on replicating an orbit trajectory in a laboratory setting; thus, for this scenario, Keplerian elements will suffice to uniquely define the orbit of the spacecraft around Eros-433. Other methods, such as the Modified Equinoctial Orbital or Unified State Model elements, use multiple parameters to reduce or avoid singularities. Compared to Kepler elements, they provide a faster performance for simulations, thus making them more useful for trajectory analysis and optimisation, which is beyond the scope of this work. Thus, they will not be discussed further as suitable candidates.

- **Cartesian Coordinates:** The three coordinates defining the position of a body are x, y, and z. Additionally, the velocity of the body can be described using the derivatives of the position along each axis, namely $\dot{x}$, $\dot{y}$, and $\dot{z}$. The coordinate systems for the laboratory is defined in Cartesian coordinates. Additionally, the translational and rotational offsets between the different reference frames can be more easily expressed using Cartesian coordinates and measuring the physical dimensions of the hardware and room.

- **Spherical Coordinates:** Spherical coordinates can be used to compute spherical harmonics for modelling the asteroid's gravitational field. They also express a position on the surface of the asteroid, which can be done using the tuple (ρ, θ, λ), where ρ is the distance from the centre of the reference frame, θ is the angle of the point of interest with respect to the Z-axis, and λ is the angle between the projection of the point onto the XY-plane and the prime meridian. The three different components are shown in Figure 4.1.

  For this work, spherical harmonic gravitational field modelling will not be included in modelling the environment of the asteroid, as this type of perturbation requires more advanced orbital modelling to be replicated by the robotic system. As the system does not yet have a benchmark for creating unperturbed trajectories, the inclusion of spherical harmonics in the orbital modelling should be performed at a later stage. However, this type of coordinate system can still be helpful to assess the results. By creating a 3D model of the asteroid as a result of images taken of the asteroid model in a circular Kepler orbit, the gravity field of the target body can be recomputed for this model to assess whether the laboratory-based reconstruction of the spherical harmonic coefficients is comparable to the benchmarked values from the NEAR Shoemaker mission.

  Typically, in the computation of spherical harmonics gravity field, the notation and naming convention are slightly different as ρ is referred to as *r*; θ is referred to as the colatitude ϕ and λ is referred to as the longitude

  It is possible to convert between the two coordinate systems. Spherical coordinates. can be transformed into Cartesian coordinates following:

$$\begin{aligned} x &= r\cos\lambda\sin\phi \\ y &= r\sin\lambda\sin\phi \\ z &= r\cos\phi \end{aligned} \tag{4.1}$$

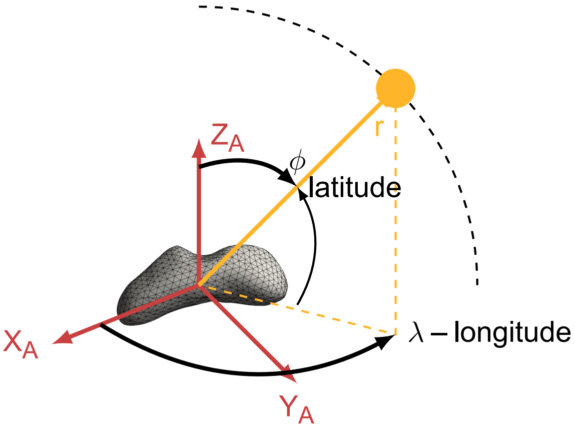


Figure 4.1: Spherical coordinates

Conversion from Cartesian coordinates into spherical can be mathematically described as follows:

$$
\begin{aligned}
r &= \sqrt{x^2 + y^2 + z^2} \\
\text{longitude} : \lambda &= \arctan\left(\frac{y}{x}\right) \\
\text{colatitude} : \phi &= \arccos\left(\frac{z}{r}\right)
\end{aligned}
\tag{4.2}
$$

- **Kepler Orbital Elements:** These are the most common way of describing the orbit of a spacecraft in three-dimensional space using six parameters: a: semi-major axis, e: eccentricity, θ: true anomaly, i: inclination, Ω: Right Ascension of the Ascending Node (RAAN), and ω: argument of periapsis τ. The elemnets are shown in Figure 4.2 (Wakker, 2015).

Rotational State A celestial body's rotational state or attitude, such as an asteroid or spacecraft, is described by its orientation relative to a reference frame. Orientation is typically defined in flight dynamics using either Euler angles or quaternions (Wie, 1998). Similarly, rotations be-

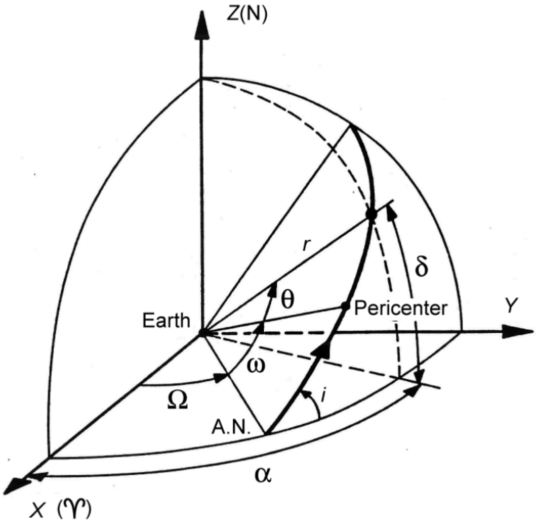


Figure 4.2: Definition of the orbital elements (Wakker, 2015)

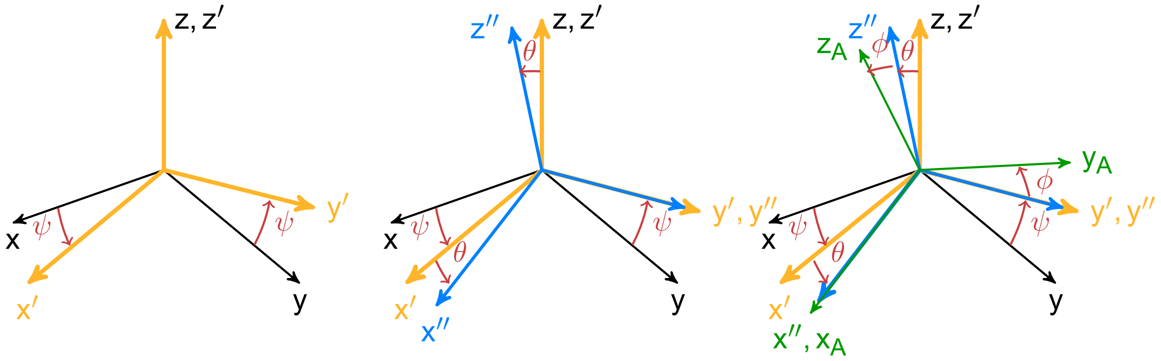


Figure 4.3: Rotation of the intermediate reference frames between I and A based on Euler angles

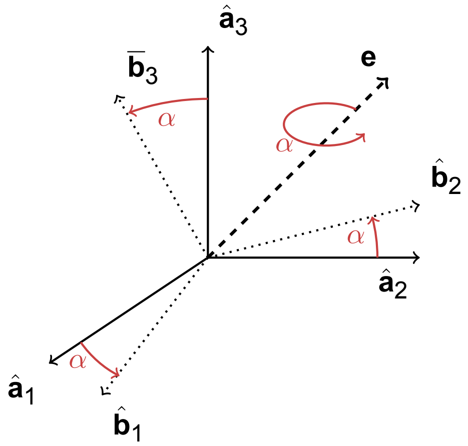


Figure 4.4: Euler eigenaxis rotation

tween two different reference frames can be defined to facilitate the conversion of results from one to another and ensure the different inputs and output compatibility. For example, the orientation of the previously defined asteroid frame can be expressed as that of the inertial fixed non-rotating frame. Another usage is to define the rotation necessary to make the outputs of the spacecraft in the simulation from the body frame compatible with the orientation of the simulated spacecraft in the laboratory, defined in terms of the reference frames of the robotic system.

Similarly, as in the case of translational motion, other types of representations exist, such as Rodrigues Parameters or modified Rodrigues Parameters that are well established in the industry for attitude definition and are often used for attitude control of spacecraft or optimisation of control commands. However, such features cannot yet be tested using the available hardware. Thus, implementing them in the software simulation would not show their benefits for the scope of this work and will not be considered further.

- **Euler Angles** describe the rotation of a rigid body around the X, Y, and Z axes. In aerospace applications, these rotations are commonly referred to as roll (ϕ), pitch (θ), and yaw (ψ). Mathematically, the angles are defined in the following ranges

$$\begin{aligned} -\pi &\leq \phi < \pi \\ -\pi/2 &\leq \theta < \pi/2 \\ -\pi &\leq \psi < \pi \end{aligned} \tag{4.3}$$

Any rotation between two reference frames can be performed using up to three successive rotations, as shown for the inertial frame to the asteroid reference frame (Figure 4.3). However, different types of rotation combinations aimed to describe a particular movement can lead to singularities, such as Gimbal locks, thus making them less reliable for control applications.

- **Quaternions** avoid rotational singularities by representing a rotation using a unit rotation axis $\hat{\mathbf{e}} = \begin{pmatrix} e_1 & e_2 & e_3 \end{pmatrix}^T$ and a rotation angle α, as depicted in Figure 4.4. Although quaternions are more abstract and less intuitive than Euler angles or rotation matrices, they are often preferred for their numerical robustness and singularity-free representation. A quaternion vector can then be introduced as a vector with four components $\mathbf{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T$, each further defined in Equation (4.4). It is important to note that the attitude quaternion definition must remain normalised, which introduces a constraint on their components to satisfy $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. The quaternion product is another important property of quaternions needed for determining the time derivative of the orientation. For two quaternions $\mathbf{p} = p_0 + p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k}$ and $\mathbf{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$, the product $\mathbf{p} \otimes \mathbf{q}$ is defined in Equation (4.6)

$$\begin{aligned} q_0 &= e_1 \sin(\alpha/2) \\ q_1 &= e_2 \sin(\alpha/2) \\ q_2 &= e_3 \sin(\alpha/2) \\ q_3 &= \cos(\alpha/2) \end{aligned} \tag{4.4}$$

$$\begin{aligned} \cos(\alpha) &= 1 - 2\sin^2\left(\frac{\alpha}{2}\right) \\ \sin(\alpha) &= 2\sin\left(\frac{\alpha}{2}\right)\cos\left(\frac{\alpha}{2}\right) \end{aligned} \tag{4.5}$$

$$\mathbf{p} \otimes \mathbf{q} = \begin{bmatrix} p_0q_0 - p_1q_1 - p_2q_2 - p_3q_3 \\ p_0q_1 + p_1q_0 + p_2q_3 - p_3q_2 \\ p_0q_2 - p_1q_3 + p_2q_0 + p_3q_1 \\ p_0q_3 + p_1q_2 - p_2q_1 + p_3q_0 \end{bmatrix}. \tag{4.6}$$

As with the selection of coordinate systems for translational motion, the choice of attitude representation is primarily influenced by the capabilities of the experimental setup. In this study, the spacecraft is modelled as a point mass as further discussed in Section 5.5 as the research objective is limited to verifying whether the robotic system can reach trajectory points.

The spacecraft's attitude is effectively controlled through the movement of the robotic arm's end-effector, which ensures the onboard camera remains pointed at the target object throughout the motion. Since no complex orbit geometries or attitude control challenges are considered, and the orientation is assumed to align with the camera's field of view, the added complexity of implementing quaternions is not justified. Euler angles suffice to describe the spacecraft's rotational state for this application.

4.2. Reference Frames

To create the desired motion profiles, the orbital reference frames required are discussed in Subsection 4.2.1 and the laboratory reference frames in Subsection 4.2.2.

4.2.1. Orbital Reference Frames

Reference frames in dynamics are defined as a set of three orthogonal axes with a common origin used to describe the position and orientation of bodies. In this report, the reference frames will adhere to the right-hand rule convention. Newton's Laws of Motion, which govern the motion of different bodies, are derived from an inertial reference frame. Such a frame is

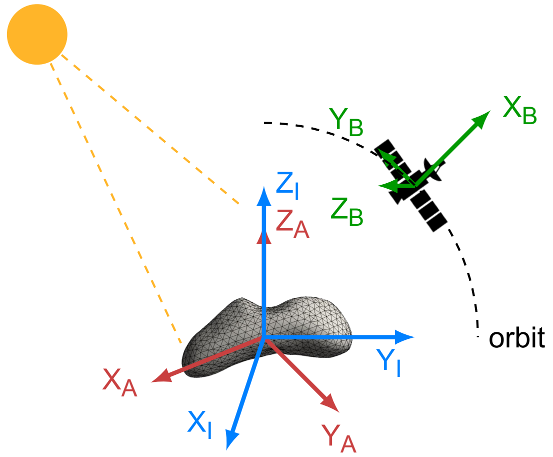


Figure 4.5: Reference frames for simulating the trajectory of the spacecraft around 433-Eros

considered stationary or moving at a constant linear velocity. When analysing the problem of a satellite orbiting an asteroid, the relevant reference frames identified are depicted in Figure 4.5 and will be discussed in detail below.

Selecting an inertial reference frame is crucial for deriving the equations of motion. The Asteroid-Satellite system orbiting the Sun can be compared to the Earth-Moon system, where a heliocentric reference frame is an inertial frame with the Sun as the origin. For long-term missions, using an inertial, non-rotating Sun-Centred (Heliocentric) Reference Frame is most suitable, as it allows for the analysis of the trajectories of all relevant bodies, similar to the case of 433-Eros full mission trajectory as provided by NASA. However, since the motion of interest in this experiment focuses on a small segment of 433-Eros orbit around the Sun, it is practical to select a different inertial frame, specifically one centred at the asteroid's centre of mass, for easier mathematical derivation and frame transformations. Similarly, the illumination lamp replicating the sunlight conditions present within the laboratory will not have a reference frame assigned. However, its position and orientation will be defined with respect to the global inertial frame of the OptiTrack system used to measure the relative potion and orientation of the objects present within the laboratory.

Reference Frames for Image-based Relative Asteroid Navigation The main reference frames are shown in Figure 4.5 with each individual frame discussed further in more detail.

I **Inertial Reference Frame:** This reference frame has its origin at the centre of mass of the asteroid, with axes X_I, Y_I, and Z_I. It is an inertial, thus non-rotating, non-accelerating reference frame, meaning it does not rotate with the asteroid and remains fixed relative to the distant celestial bodies. Here, the Z_I is aligned with the asteroid's angular momentum vector and is thus defined along the asteroid's spin axis. The remaining two axes are defined in the prime meridian plane of the asteroid with X_I originating at the centre of gravity and pointing towards the prime meridian, which has been defined as a crater on one extremity of the asteroid. Y_I is perpendicular to both X_I and Z_I to complete the right-handed system.

A **Asteroid Reference Frame:** Originating at the asteroid's centre of mass, this rotating reference frame follows the right-hand rule, where X_A aligns with the principal axis corresponding to the maximum moment of inertia, Y_A with the intermediate moment, and Z_A with the minimum moment. It determines the asteroid's attitude relative to the inertial frame, defining rotations about all three axes. Razgus (2017) adopts this approach, who describes the asteroid as a flat-spinner, with Z_A coinciding with the rotation axis. Selecting the two reference frames with coinciding origins and axes provides the benefit that the asteroid's motion in space can be described independently from the surface-related analysis of the asteroid. In this reference frame, it is possible to define the landmarks on the surface of the asteroid using three coordinates with respect to the centre of gravity of the asteroid.

B **Spacecraft Reference Frame:** A body-fixed reference frame is a right-handed coordinate system with axes X_B, Y_B, and Z_B. In this work, it is assumed that the navigation camera will point directly towards the asteroid. The camera positioning can be expressed with respect to the body of the spacecraft as it is mounted. Thus, it will be convenient to define X_B aligned with the pointing orientation of the camera. Previous works, for which communication with neighbouring spacecraft is of importance, have used the communication antennas as direction for defining Y_B (Munuera Vilalta, 2024). In this work, the pointing of the communication antennas is irrelevant. However, this assumption can be kept for simplicity and potential later integration of the two works. Lastly, Z_B is defined perpendicular to the two axes to complete the right-handed coordinate system. This convention will allow for an easier processing of the sensor readings to express the orientation of the navigation camera and assume the spacecraft is aligned. It is important, however, to consider that in real scenarios, misalignments and sensor errors need to be accounted for.

Simplified Frame Usage for Orbital Propagation While a body-fixed reference frame for the spacecraft B is formally introduced to remain consistent with the dynamics of rigid body motion and to enable future extensions to attitude modelling, its use is not required for the translational analysis conducted in this work. The spacecraft is modelled as a point mass under the influence of the gravitational field of asteroid 433-Eros. This simplification is valid given the objective of the study: to determine whether the required spatial trajectory can be reconstructed within a laboratory environment. At the same time, the spacecraft's rotational state or inertia properties cannot be replicated by the robotic hardware available.

The orbital trajectory is computed using Keplerian dynamics, where the spacecraft motion is integrated using the equations of motion in an inertial reference frame I centred at the asteroid's centre of mass. This inertial frame remains non-rotating with respect to distant celestial bodies and serves as the frame in which Newton's second law is applied. Under this model, the only state variables are the Cartesian position and velocity vectors, $\mathbf{r} = [x, y, z]^T$ and $\dot{\mathbf{r}} = [\dot{x}, \dot{y}, \dot{z}]^T$, respectively as further presented in Section 5.5.

As no attitude dynamics or control are simulated, the spacecraft orientation is not propagated, and the body-fixed frame B does not contribute to the trajectory generation. Nevertheless, it is conceptually retained in the frame architecture to allow for the later incorporation of camera field-of-view constraints, target pointing, and sensor alignment verification as the development of the robotic testbed for relative asteroid navigation will continues to develop. The spacecraft's orientation in the experimental phase relies on the orientation of the end-effector of a robotic arm such that the camera is pointed towards the asteroid model. The camera's position in the laboratory space will represent the spacecraft's position. In this case, the spacecraft's body frame can be reintroduced to define the alignment of the sensor with respect to the trajectory, even if it was not required during the orbital propagation.

In the laboratory setup, the position vectors resulting from the simulation are mapped from the asteroid-centred inertial frame I to the laboratory inertial frame L through linear spatial scaling. This conversion ensures that the orbital trajectory, originally computed over a scale of hundreds of kilometres, can be recreated within the limited physical dimensions of the experimental environment. The mapping uses a fixed ratio between the simulation domain and

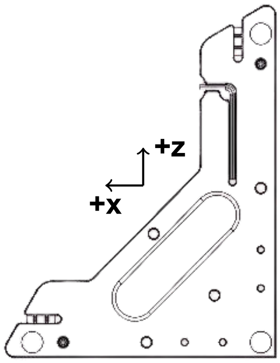


Figure 4.6: OptiTrack ground plane and origin calibration square with preset coordinate system

the laboratory workspace, allowing each simulated position to be assigned a corresponding reachable position in the robotic workspace.

Given that the spacecraft is modelled as a point mass, its orientation is not propagated as part of the simulation. Instead, the assumption is that the camera mounted on the robotic arm remains pointed at the target throughout the motion. As a result, the spacecraft's orientation is implicitly defined through the programmed motion of the robot's end-effector, which is aligned with the direction between the camera and the target. This allows the system to replicate the correct relative geometry between the spacecraft and the asteroid without the need to simulate full attitude dynamics. The spacecraft body frame B can, therefore, be considered fixed relative to the instantaneous line-of-sight, and any transformation involving this frame is determined by the relative pose between the chaser and the target in the OptiTrack system. This simplified assumption is sufficient for validating the positional reachability and visual alignment requirements of the current experimental setup.

4.2.2. Laboratory Reference Frames

The hardware used for the experimental part of the project uses different reference frames to determine the spatial positions of various components within the lab. That is because there is no direct method to command the robotic setup to achieve a specific location within the laboratory's space. The setup can only accept commands relative to its current position. The OptiTrack setup available in the laboratory described in Section 3.5 can be used to determine the positioning of markers. This approach can give an initial estimate of the positioning of the hardware in the laboratory. The proposed method for this work will create a translation between absolute coordinates within the space of the laboratory and the command inputs required for the setup to follow to reach that target position. A series of reference frames and their relations need to be introduced for this.

L **Inertial Reference frame of the Laboratory:** This is a right-hand coordinate system introduced for ease of interpretation of the space available within the laboratory. Its origin will be assumed to match the origin of the OptiTrack reference frame when the ground plane calibration is performed. However, the OptiTrack reference frames are set in a more counter-intuitive manner. Thus, the laboratory reference frame is initialised such that the vertical z-axis is pointing upwards along the laboratory height, the y-axis is pointing along the laboratory length with the positive direction being aligned with the positive z-axis of the OptiTrack, and lastly, the x-axis is completing the right-hand coordinate system pointing along the laboratory width with the positive direction opposite to the positive x direction of the OptiTrack frame.

O **OptiTrack Reference frame:** This type of reference frame is necessary such that the position and orientation of different objects within the lab can be expressed in a consistent manner. An inertial reference frame origin can be created using the calibration equipment provided by the manufacturer, showcased in Figure 4.6. The calibration square[1] can be aligned such that it references the desired axis orientation. In this case, the two axes can be aligned with the laboratory's width and length. Motive recognises the longer leg on the calibration square as the positive z-axis and the shorter leg as the positive x-axis. The positive y-axis will automatically be directed upward in a right-hand coordinate system. It is important to note that such a calibration and definition of the global inertial OptiTrack reference frame needs to be performed before the experiment.

Reference Frames for the OptiTrack markers: The OptiTrack system relies on object identification through the use of markers. Two unique sets of markers can be created to identify the spacecraft/chaser body and the asteroid/target. The markers need to be placed on adjacent interfaces to the camera sensor and the 3D printed asteroid model. The tracking software then defines the reference frames based on the pivot point or bone of the Rigid Body, which is used to define its position and orientation. The default position of the bone for a newly created rigid body is at its geometric centre, and its orientation axis will align with the global coordinate axis of frame o. An additional reference frame identifying the position of the rover base of the chaser robot is introduced. This is useful in identifying the mount of the robotic arm placed on the rover base.

OBC **OptiTrack Rover Base Chaser Reference Frame**: This frame describes the pose of the rover base acting as the spacecraft body (chaser platform). It is defined by a rigid body of markers mounted on the base top surface in the corner, as shown in Figure 4.5. Its orientation is assumed to be aligned with the global frame O, and its position is reported directly from the OptiTrack system in coordinates expressed in O for each experiment as shown in Table 4.1.

OC **OptiTrack Chaser Reference Frame** This frame represents the rigid body of the markers mounted on the robotic arm end effector. It identifies the relative position and orientation of the camera, which is used for landmark tracking. Like the previous frame, its orientation is aligned with O, and its position is measured relative to the origin of the OptiTrack frame for each experiment.

OT **OptiTrack Target Reference Frame** This frame identifies the position of the target body, the 3D-printed model of asteroid 433-Eros, as tracked via markers. It is similarly aligned with the global frame O, and the locations of the markers in the O frame are used to determine the target to be tracked using the end effector of the chaser robot.

Reference Frames for the spacecraft (chaser) and asteroid (target): The tracking system available in the laboratory can monitor the position and orientation of the rigid body defined by the two unique marker configurations not the actual position of the camera and the asteroid. Thus, two additional reference frames need to be defined to go from measurements in the OptiTrack reference frame of the two objects. Thus creating analogous reference frames for the body reference frame, which in the laboratory will correspond to the chaser reference frame, and an analogous reference frame for the asteroid reference frame, which in the laboratory corresponds to the target reference frame. Thus, the origins will be assumed to correspond with the simulation reference frames, and translation and rotations of the two frames need to be expressed with respect to the corresponding OptiTrack reference frames:

[1] OptiTrack Calibration https://docs.optitrack.com/motive/calibration (accessed: 5 Feb 2025)

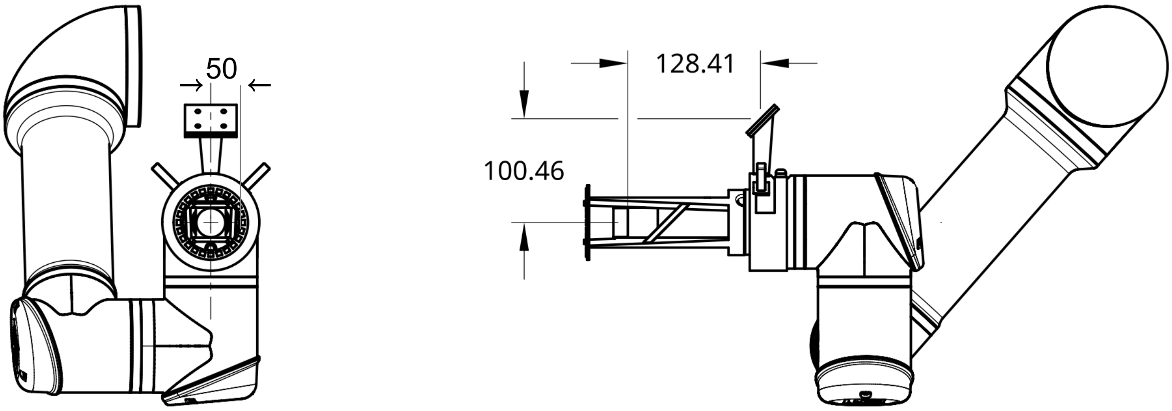


Figure 4.7: Distances (mm) in Cartesian coordinates from the centre of the OptiTrack Chaser Reference Frame to the centre of the camera representing the origin of the Chaser Frame adapted from Sundberg (2024)

C **Chaser Reference Frame:** This reference frame needs to be defined at the centre of the camera sensor to match the assumption made in the simulation software that the camera pointing will match the orientation of the spacecraft. Thus, the origin of this reference frame is defined by measuring the distance from the centre of the mounting points of the OptiTrack marker support to the centre of the camera as explained in the work of Sundberg (2024) shown in Figure 4.7.

T **Target Reference Frame**: For the target, in this case, the 3D printed asteroid model, an interface to support the asteroid model needs to be manufactured before the final dimensions can be provided. This support aims to create an interface between the end-effector of the robotic arm and the 3D-printed asteroid so that this can be rotated around its rotation axis. The shift in the origin from the Target reference frame with respect to the OptiTrack Target Reference frame will be computed in a similar manner.

Reference Frames for the Chaser Robot

BC **Base Reference Frame** The OptiTrack convention can be maintained to avoid additional conversions for rotating the reference frames. Thus, Y_{BC} defines the vertical direction of the rover base, X_{BC} is the direction of forward movement aligned with the length of the rover base and Z_{BC} completest the right-handed coordinate system as it points along the width of the rover base.

AMC **Mount Robotic Arm Reference Frame**. This reference frame creates a correlation between the position of the end-effector and that of the rover base. The origin of this reference frame is defined by computing the displacements from the centre of the rover base to the centre of the mounting point of the robotic arm placed on the top of the rover base. While this point could be already defined in the Base Reference Frame, the direct correlation between the position of the end-effector with respect to the position of the base cannot be directly requested from the system. However, the robotic arm can return the position of its end effector with respect to the reference frame of the robot's base, which coincides with its mounting point on the robotic arm. Y_{AMC} defines the vertical direction of the robotic arm. Z_{AMC} and X_{AMC} are chosen in the same direction as Z_{BC} and X_{BC}, and the plane between the two couples are assumed to be parallel.

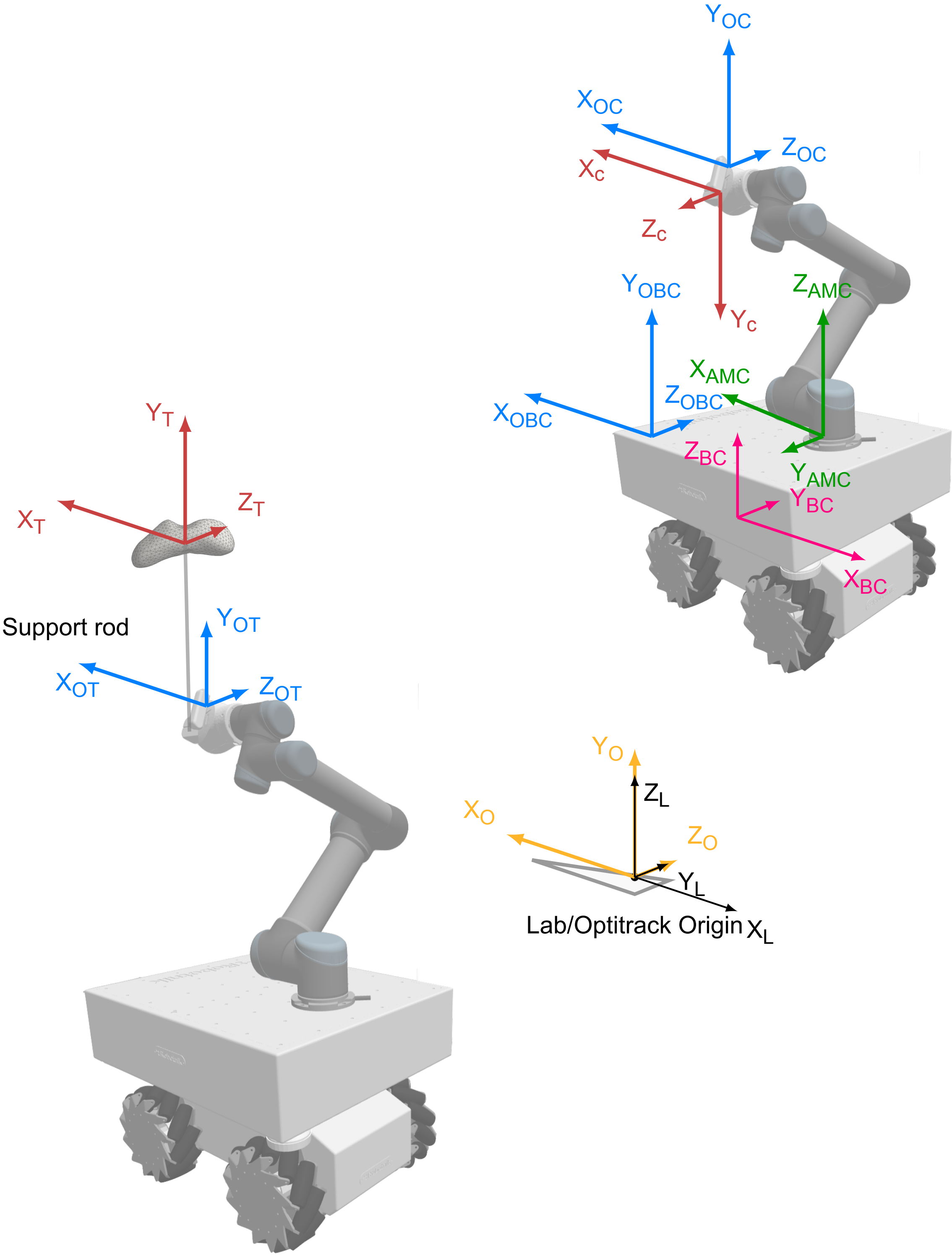


Figure 4.8: Reference frames for simulating a full spacecraft trajectory around Eros in the lab. From left to right, bottom to top (OT) OptiTrack Target, (T) Target, (O) OptiTrack (L) lab, (B) Base Chaser, (OBC) OptiTrack Base Chaser, (M) Arm Mount, (C) Chase, (OC) OptiTrack Chaser

The described offset and orientations used to define and place the reference frames with respect to one another are summarised and presented in Section 4.4, Table 4.1.

4.3. Frame Transformations

Frame transformations are essential for describing the dynamics and kinematics of different bodies relative to each other and their environment. Forces and moments, whether internal or external, may be defined in various reference frames related to the asteroid, the spacecraft, or another chosen point of reference. To analyse the motion according to Newtonian mechanics, it is necessary to express these forces and moments within a common reference frame.

When are frame transformations required

- **Dynamics:** This involves understanding how an applied moment affects the rotational acceleration of the spacecraft. For instance, controlling the spacecraft's attitude to maintain a desired orientation while tracking a specific landmark on the asteroid's surface. In this case, frame transformations are required such that the external torques due to the environment, the control torque from the control system of the spacecraft, and any other potential disturbances can all be expressed in the same reference frame to compute the total moment acting on the spacecraft.
- **Kinematics:** This focuses on how changes in rotation affect the spacecraft's attitude. Frame transformations are necessary to express the changes in orientation going from an inertial reference frame where the equations of motion are defined to the body reference frame to evaluate by how much the attitude of the body has changed where the deviation can be expressed as mentioned in the previous section using Euler angles or quaternions

Mathematical formulation A frame transformation typically involves both translational and rotational components. To express a vector $\mathbf{v}_A$ defined in reference frame A in another frame B as shown below:

$$\mathbf{v}_B = \mathbf{T} + \mathbf{C}_{B/A}\mathbf{v}_A \tag{4.7}$$

where $\mathbf{T}$ is the translation vector between the origins of frames A and B, and $\mathbf{C}_{B,A}$ is the Direction Cosine Matrix (DCM) defining the rotation from frame A to frame B (Mooij, 2024). An element-wise expansion of this equation is given in Equation (4.8). This method leverages the orthogonality of the transformation matrix, which makes it invertible, allowing for transformations from one reference frame to another as shown in Equation (4.12) and Equation (4.13).

$$\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix} + \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \tag{4.8}$$

Changes in orientation within a reference frame are defined by rotation matrices around the z-, y-, and x-axes:

$$\mathbf{C}_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{C}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \mathbf{C}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{4.9}$$

The transformation matrix from the inertial frame I to the body frame B is obtained using the 3-2-1 sequence $\psi \rightarrow \theta \rightarrow \phi$

$$\mathbf{C}_{B/I} = \mathbf{C}_z(\psi)\mathbf{C}_y(\theta)\mathbf{C}_x(\phi) \tag{4.10}$$

The inverse transformation matrix $\mathbf{C}_{I,B}$ can be determined can also be expressed as:

$$\mathbf{C}_{I/B} = \mathbf{C}_{B/I}^{-1} = \mathbf{C}_{B/I}^{T} = \mathbf{C}_x(-\phi)\mathbf{C}_y(-\theta)\mathbf{C}_z(-\psi) \tag{4.11}$$

Once the transformation matrix is known, vectors presented in one frame of reference A can be expressed in another frame of reference B using the transformation matrix $\mathbf{C}_{B/A}$. This can mathematically be expressed as follows:

$$\mathbf{r}^B = \mathbf{C}_{B/A}\mathbf{r}^A. \tag{4.12}$$

The inverse can then also easily be determined using:

$$\mathbf{r}^A = \mathbf{C}_{A/B}\mathbf{r}^B = \mathbf{C}_{B/A}^T\mathbf{r}^B. \tag{4.13}$$

Table 4.1: Overview of DCM and translation vectors between successive frames

Frame pair (A, B)	DCM $\mathbf{C}_{B/A}$	Translation $\mathbf{T}_{B/A}$
L→L1	$\mathbf{C}_x\left(-\frac{\pi}{2}\right) = \begin{bmatrix}1 & 0 & 0\\0 & 0 & -1\\0 & 1 & 0\end{bmatrix}$	$\begin{pmatrix}0\\0\\0\end{pmatrix}$
L1→O	$\mathbf{C}_y(\pi) = \begin{bmatrix}-1 & 0 & 0\\0 & 1 & 0\\0 & 0 & -1\end{bmatrix}$	$\begin{pmatrix}0\\0\\0\end{pmatrix}$
O→OBC	$\mathbf{I}_3$	$\begin{pmatrix}x_{O\_OBC}\\y_{O\_OBC}\\z_{O\_OBC}\end{pmatrix}_O$
O→OC	$\mathbf{I}_3$	$\begin{pmatrix}x_{O\_OC}\\y_{O\_OC}\\z_{O\_OC}\end{pmatrix}_O$
O→OT	$\mathbf{I}_3$	$\begin{pmatrix}x_{O\_OT}\\y_{O\_OT}\\z_{O\_OT}\end{pmatrix}_O$
OBC→BC	$\mathbf{C}_z(\pi)\,\mathbf{C}_x\left(-\frac{\pi}{2}\right) = \begin{bmatrix}-1 & 0 & 0\\0 & 0 & 1\\0 & 1 & 0\end{bmatrix}$	$\begin{pmatrix}-\frac{bL}{2} + mxT\\-\frac{bH}{2} - hT\\-\frac{bW}{2} + mzT\end{pmatrix}_{OBC}$, $bL = 0.760\,m$ $bW = 0.633\,m$ $bH = 0.690\,m$ $mxT = 0.0803\,m$ $mzT = 0.0803\,m$ $hT = 0.019\,m$
BC→AMC	$\mathbf{C}_z(\pi) = \begin{bmatrix}-1 & 0 & 0\\0 & -1 & 0\\0 & 0 & 1\end{bmatrix}$	$\begin{pmatrix}0\\0\\\frac{bH}{2}\end{pmatrix}_{BC}$, $bH = 0.690\,m$
OC→C	$\mathbf{C}_x(\pi) = \begin{bmatrix}1 & 0 & 0\\0 & -1 & 0\\0 & 0 & -1\end{bmatrix}$	$\begin{pmatrix}x_C\\y_C\\z_C\end{pmatrix}_{OC}$, $x_C = 0$ $y_C = -0.10046\,m$ $z_C = -0.12841\,m$
OT→T	$\mathbf{I}_3$	$\begin{pmatrix}x_T\\y_T\\z_T\end{pmatrix}_{OT}$, $x_T = -0.13\,m$ $y_T = 0.44\,m$ $z_T = 0.14\,m$

4.4. Summary Frame Implementations

Table 4.1 presents the relative transformations between reference frames defined in the experimental setup. Each row specifies the rotation matrix $\mathbf{C}_{B/A}$ and translation vector $\mathbf{T}_{B/A}$ that map coordinates from frame A to frame B. Rotations are either expressed as identity matrices $\mathbf{I}_3$, indicating no relative orientation change, or as intrinsic Euler rotations, following the convention introduced in the previous section.

For the frame pairs O $\rightarrow$ OBC, O $\rightarrow$ OC, and O $\rightarrow$ OT, the translation vectors are test specific and obtained from the initial OptiTrack calibration before each test. These values define the positions of the marker frames with respect to the lab frame at the start of each experiment. All other translation vectors are fixed and based on mechanical design parameters or marker placements.

The rover base dimensions are defined as: length bL = 0.760 m, width bW = 0.633 m, and height bH = 0.690 m. The calibration tool used as markers for the rover base gives the following offsets from the edge of the rover mxT = 0.0803 m (longitudinal), mzT = 0.0803 m (lateral), and hT = 0.019 m (vertical), defining the displacement between the OptiTrack markers and the geometric centre of the tool. For the camera frame C, the vector $\begin{pmatrix} x_C & y_C & z_C \end{pmatrix}^T$ defines the position of the camera lens with respect to its tracked marker. Similarly, the vector $\begin{pmatrix} x_T & y_T & z_T \end{pmatrix}^T$ defines the position of the asteroid centre relative to its OptiTrack marker.

5 Modelling the Movement of the Robotic Setup

The software for developing the framework for the control input commands is presented in Section 5.1. The proposed test aimed to explore the performance of the hardware in creating movements that can support asteroid relative navigation are discussed in Section 5.2 with, particularities of for the rover base in Section 5.3, for the robotic arm in Section 5.4 and for the full system in Section 5.5. Lastly, uncertainty sources are discussed in Section 5.6.

5.1. Software Used

The software developed to model the movement of the robotic setup within the laboratory space was implemented in python. This choice reflects the need for an open-source, license-free environment that is easily adaptable to various applications, unlike proprietary alternatives such as Matlab. The code developed for this project can be run on different types of computers, provided they have Python 3.11 installed. In addition, for the work of this thesis, a virtual environment was created to manage the packages and the dependencies between them such that the code can be compiled and run on different machines. A requirements text file is provided to be used when setting up the interpreter of the project, and it can be found in the GitHub repository.

Two main software packages were used. The first is SymPy Physics Mechanics[1], which enables symbolic modelling of reference frames and transformation chains between points and vectors defined within those frames. The second is the Robotics Toolbox for python, developed by Peter Corke[2]. This toolbox provides predefined robotic models and tools for solving inverse kinematics, allowing conversion between Cartesian coordinates for the end-effector and the corresponding joint angle commands. While several Universal Robots models are available as predefined models, the UR16e is not included by default. However, it can be manually defined using a URDF description file. This file has been retrieved from the GitHub repository of the manufacturer, and it is the same version that has been used for setting up the Gazebo simulation of the hardware shown in Figure 3.10.

[1] SymPy Mechanics https://docs.sympy.org/latest/explanation/modules/physics/mechanics/index.html (accessed: 10 Feb 2025)

[2] Python Robotics Toolbox https://petercorke.github.io/robotics-toolbox-python/ (accessed: 10 Feb 2025)

5.2. Proposed Tests

The modelling task was divided into a series of test cases, each designed to validate a specific movement pattern of the robotic system. These scenarios were structured to isolate the contributions of the rover base and the robotic arm to the system's overall accuracy. The first set of test cases investigates the motion of the robotic base. In these, a series of different tests were conducted, in which either translation or angular commands were provided to the robotic base. In the subsequent test cases, the base remains stationary while the focus shifts to the robotic arm. Various tests are conducted to assess whether the arm's end effector follows the expected trajectory, either linear or circular. The following scenarios evaluate the robotic arm's ability to track a path while continuously orienting toward a fixed target. In the final case, the base and the arm are commanded to follow a circular trajectory, approximating a Keplerian path within the laboratory environment. These configurations allow for the evaluation of both isolated and integrated system performance.

Each movement case was implemented using a dedicated Jupyter Notebook[3]. The process begins with defining all required reference frames, followed by constructing the robotic arm model using the Robotics Toolbox. Next, the trajectory that the end effector must follow is defined. Inverse kinematics are used to compute the corresponding joint commands. After determining the motion profile, an inputFile.csv is generated, containing the commanded rover base movement, joint angles, manoeuvre duration, and camera trigger flag (Table 5.4). The joint commands are then plotted to verify continuity and smoothness (Figure 5.14). Finally, the reference frames and their orientations are visualised to confirm the correctness of the setup. During the initial setup and calibration of the system, the reference frames would have to be aligned as shown in Figure 4.8. The resulting input commands can then be tested in the Gazebo simulation to check that the system executes the expected movement therefore testing it on the hardware.

All trajectory tracking tests begin with an initial calibration of the laboratory space using the OptiTrack motion capture system. This process establishes a consistent global coordinate system, the OptiTrack frame O, as defined in Section 4.2. Marker positions determine the locations of the chaser arm, chaser base, and target robot. Each is assigned a corresponding local reference frame that shares orientation with frame O. Additional reference frames are defined based on hardware constraints and operational alignment requirements, as detailed in Section 4.2. Throughout all tests, the OptiTrack system remains continuously active to track all markers' 3D positions and orientations in real-time. The system records the x, y, and z coordinates, along with the rotation data of each tracked object, with respect to frame O, and stores the measurements in .csv format. These files are later post-processed to analyse motion consistency and system performance. OptiTrack measurements are illustrated in Figure 5.1 for Case 1 and in Figure 5.2 for Case 2.

The cases considered are presented in the following sections. It is important to note that each case has multiple additional tests summarised in Table 5.1 - 5.3.

- **Case 1:** Mobile Base, Static Arm
- **Case 2:** Static Base, Mobile Arm
- **Case 3:** Coordinated Base and Arm Movement (Keplerian Trajectory)

When determining which tests are necessary to evaluate whether the proposed framework is suitable for modelling the movement of the robotic system to replicate Keplerian trajectories within the laboratory, three aspects are relevant: accuracy, repeatability and generalization. These properties have to be evaluated for the movement of the robotic components both individually and combined. The RB-Kairos rover base does not have information provided by its manufacturer in terms of the accuracy and reproducibility of input commands to the rover.

[3] Jupyter https://jupyter.org/ (accessed: 11 March 2025)

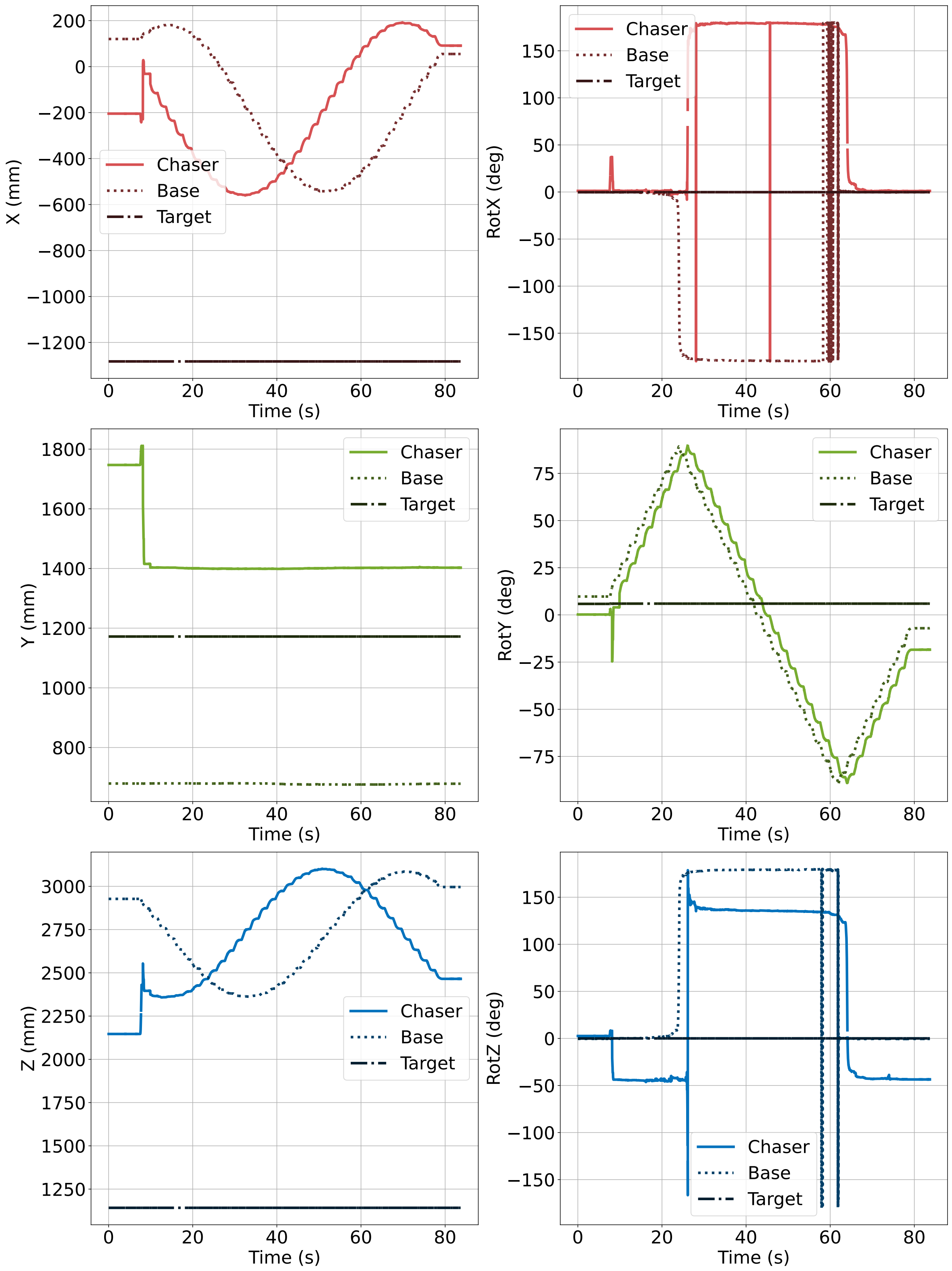


Figure 5.1: Raw OptiTrack marker position data recorded during test C1_A1_spin.

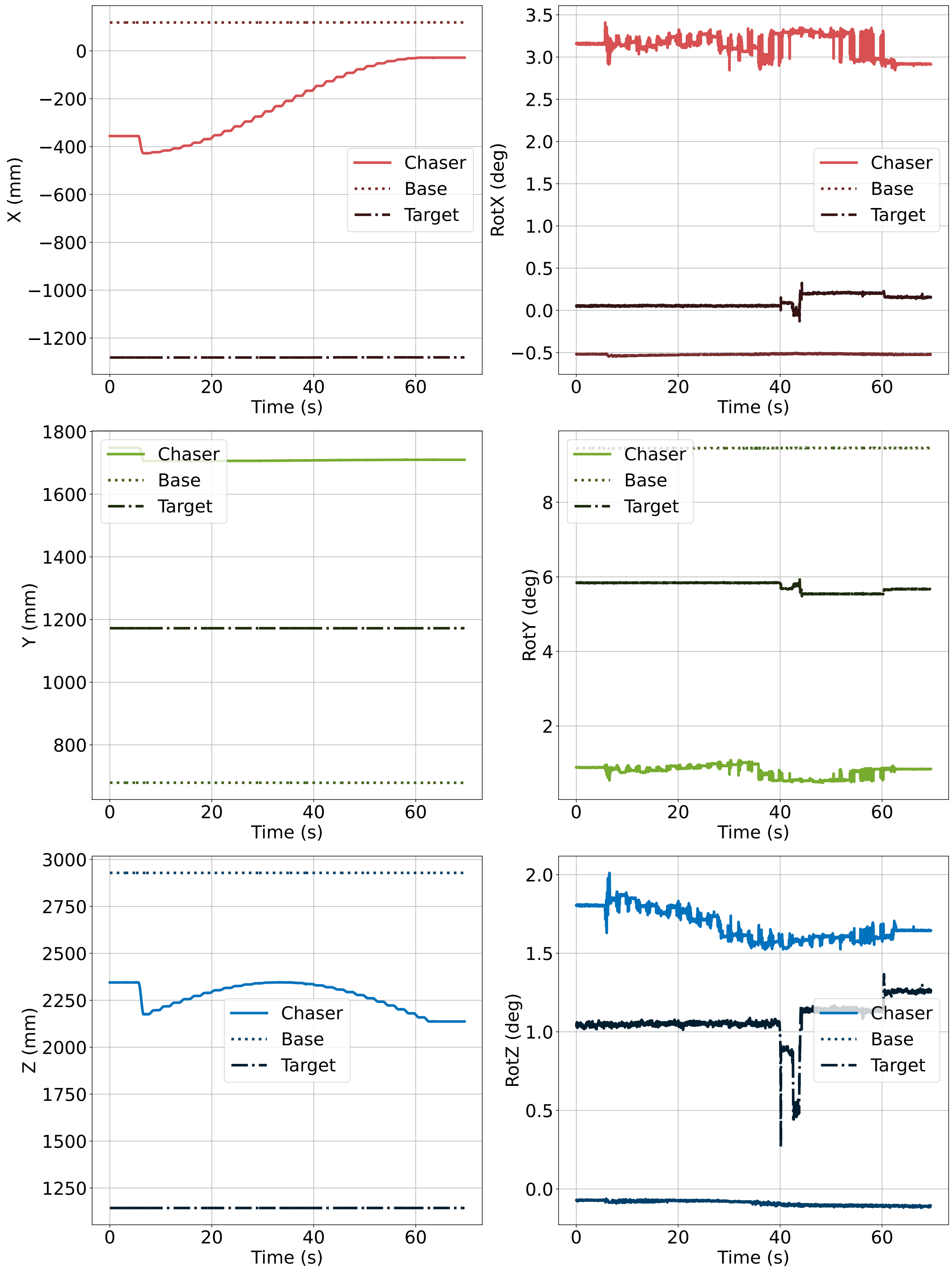


Figure 5.2: Raw OptiTrack marker position data recorded during test C2_B3_linear_x_repeat.

Table 5.1: Test Matrix for Case 1: Mobile Base, Static Arm

Test ID	Name	Description
C1_A1_spin	Base rotation	360° rotation at initial base location
C1_A2_spin_offset1	Base rotation - offset 1	Same as A1 from alternate location
C1_A3_spin_offset2	Base rotation - offset 2	Same as A1 from third location
C1_B1_linear_x_repeat	Line repeat - X axis	5x back-and-forth on X axis
C1_B2_linear_y_repeat	Line repeat - Y axis	5x back-and-forth on Y axis
C1_B3_linear_xy_repeat	Line repeat - XY diagonal	5x repeat on diagonal vector
C1_D1_circle_base_start1	Base-only circle – start 1	Combined linear/angular input
C1_D2_circle_base_start2	Base-only circle – start 2	Repeat from alternate base pose
C1_D3_circle_base_start3	Base-only circle – start 3	Repeat from third pose

Table 5.2: Test Matrix for Case 2: Static Base, Mobile Arm

Test ID	Name	Description
C2_A1_line_x	Linear X motion	End-effector moves 0.3 m in X
C2_A2_line_y	Linear Y motion	End-effector moves 0.12 m in Y
C2_A3_line_xy	Linear XY motion	Diagonal path along [0.2m, 0.1m]
C2_B3_half_circle_xy	Half-circle XY plane	Semi-circle in horizontal plane
C2_B4_half_circle_yz	Half-circle YZ plane	Semi-circle vertical plane
C2_B4_half_circle_xyz	Arbitrary half-circle	Semi-circle in arbitrary 3D plane
C2_C1_track_target_line	Target tracking linear	End-effector tracks a line while pointing at target
C2_C2_track_target_circle	Target tracking circular	Partial orbit around target while maintaining pointing

Table 5.3: Test Matrix for Case 3: Coordinated Base and Arm Movement

Test ID	Name	Description
C3_A1_kepler_orbit_100k_0deg	Equatorial orbit	Full circle simulating 100 km orbit, 0° inclination
C3_A2_kepler_orbit_100k_30deg	Inclined orbit	100 km equivalent with 30° inclination
C3_B1_surface_scan	Surface scan	Base moves around target, arm performs local sweeps

Thus, tests were devised to assess the accuracy and repeatability of linear control inputs and angular control inputs to the rover base, as these are commanded separately; this is done through tests C1-A1 to C1-B3. Once more information on accuracy and reproducibility is gathered, the generalization tests can be more fairly assessed to evaluate whether the errors in positioning the robotic system within the lab are due to the framework developed for this work or are a limitation of the setup available. Lastly, tests C1-D1 are meant to evaluate the suitability of the rover base in creating circular motions within the laboratory's space using a combination of linear and angular command inputs, which have been evaluated through previous tests.

In the case of the robotic arm, the accuracy and pose reproducibility of the arm have already been evaluated and assessed by its manufacturer in conformity with the ISO standard 9283 to have a pose repeatability within 0.05 mm (Brandt, 2024). Thus, the tests devised for Case 2, when the rover base is static, and the robotic arm is moving, are aimed to evaluate the developed framework's performance to command the robotic arm's end-effector to follow a given trajectory defined within the laboratory's space.

When determining the tests to be performed in Case 3, where the movement of the rover base is coordinated with that of the robotic arm, it is relevant to evaluate which applications are necessary for asteroid navigation. The NEAR mission, described in Section 2.1, has served as a source of inspiration, particularly as the asteroid model available in the laboratory is that

Table 5.4: Sample control input file for robot movement and arm configuration for test C1_A1_spin

Idx	baseX [m]	baseY [m]	baseTh [rad]	shoulder_pan [rad]	shoulder_lift [rad]	elbow [rad]	wrist1 [rad]	wrist2 [rad]	wrist3 [rad]	Δt [s]	Photo [-]
1	0.0	0.0	0.1745	0.221098	-1.873097	1.4438	0.429297	0.223716	-2.356196	2.0	0
2	0.0	0.0	0.1745	0.221098	-1.873097	1.4438	0.429297	0.223716	-2.356196	2.0	0
3	0.0	0.0	0.1745	0.221098	-1.873097	1.4438	0.429297	0.223716	-2.356196	2.0	0
4	0.0	0.0	0.1745	0.221098	-1.873097	1.4438	0.429297	0.223716	-2.356196	2.0	0
5	0.0	0.0	0.1745	0.221098	-1.873097	1.4438	0.429297	0.223716	-2.356190	2.0	0

of 433-Eros, which was the target of the mission. Two main mission phases have been identified: circular orbits around the asteroid for gravitational field modelling and surface scanning. The first phase involves performing circular orbits at various altitudes, 50 km, 35 km, or 19 km, to assess the gravitational field. Since the tests conducted in the laboratory assume unperturbed circular orbits, the specific altitude is not critical. When scaled to the dimensions of the laboratory, the motion profiles will be equivalent. An altitude of 100 km is selected for consistency with previous research (Munuera Vilalta, 2024), allowing continuity and the opportunity later to evaluate the impact of perturbations on the robotic system and whether they can be measured in the lab environment. The second application of interest is surface reconstruction. In the confined space of the laboratory, the robotic system does not follow a trajectory typical of a spacecraft in orbit. Instead, the scan is guided by the overlap criteria required by the 3D scanning software used in 3D printing applications. This approach is adopted due to the unavailability of specialised surface reconstruction software used in space missions. As a result, the currently available 3D printing-oriented software is used for an initial feasibility assessment of the proposed surface scanning framework. Future work could adapt the methodology to more accurately simulate continuous spacecraft movement within the lab using the developed code modules, should mission-specific reconstruction software become available.

To ensure consistency and clarity in managing the numerous validation tests conducted on the robotic framework, a structured naming convention was adopted. Each test name follows the format C[CaseID]_[GroupID]_[Description], where CaseID refers to the main test category: C1 for the mobile base with static arm, and C2 for the static base with mobile arm and lastly, C3 for the coordinated movement of the base and the arm. The GroupID is an alphabetical identifier grouping related tests such as spinning motion, linear motion, circular motion or target tracking. The Description provides a short explanation of the specific test being performed.

5.3. Case 1: Mobile Base, Static Arm

A series of movement scenarios were simulated to validate the modelling and command generation and evaluate the system's accuracy. The rover base can only be commanded using relative positioning inputs along the x and y coordinates and a rotation input defined by an angle θ. These inputs are defined in the reference frame of the rover base BC. The tests were designed to evaluate the performance of these two types of command inputs.

5.3.1. Input Command Generation

Three tests were performed in which the rover base received only angular command inputs. For this purpose, a full 360° rotation test was implemented, where the robotic base completes a full spin around its vertical axis. A key challenge in this test is that the rover base accepts commands relative to the reference frame BC, which corresponds to the geometric centre of the base. However, the position tracked within the laboratory space relies on markers placed on the surface of the robotic base, which are used to define the reference frame OBC. This placement represents a source of uncertainty for the system performance, and it is further discussed in Section 5.6

Group A: Angular Command Input

Starting from the current position of the rover base, a full 360° rotation is achieved by discretizing the trajectory into angular command increments of 10°. This results in a trajectory composed of 37 discrete rotation points. For each increment, a new rotational reference frame, denoted $\mathrm{BC}_{\mathrm{new}}$, is computed using the current angle of rotation and the z-axis of the previous BC frame. The distance between the geometric centre of the base (origin of BC) and the surface-mounted markers (defining OBC) remains constant, as both are fixed in relation to the structure of the robotic base. By computing the DCM between the old BC frame and the new $\mathrm{BC}_{\mathrm{new}}$ frame, it is possible to evaluate how the vector between OBC and BC rotates within the laboratory frame as the rover base spins. Knowing this vector in both the original and rotated frames allows the computation of the displacement of the origin of frame OBC from its initial position due to rotation. This step is required for computing the expected position of the rover base in the simulation to then later be compared with the OptiTrack measurements. This displacement is measured using the OptiTrack system. A sample of the command input file is shown in Table 5.4.

The robotic arm is held in a fixed configuration by maintaining constant joint values, as illustrated in Figure 3.4. The test is repeated three times, defining Group A with test A1, A2, and A3. Each test begins from a different initial position of the rover base within the laboratory. This repetition aims to evaluate whether the initial placement of the base influences performance.

Group B: Translational Command Input

The following Group B test cases are designed to evaluate the framework's performance in tracking linear motion. Group B comprises three distinct tests: C1_B1_linear_x_repeat, C1_B2_linear_y_repeat, and C1_B3_linear_xy_repeat, each assessing the reproducibility of a commanded trajectory. In each test, the robotic base is instructed to execute back-and-forth motions along a predefined linear path, repeated five times, to evaluate the consistent reproduction of both start and goal positions. These tests are conducted along the x-axis, y-axis, and diagonal xy-axis, employing displacement commands of 1 m, -2 m, and a combined displacement vector [1 m, -2 m], respectively. In contrast to Group A, where the initial positions of the rover base vary across the laboratory surface, Group B consistently initiates the rover base from an identical starting position. This approach ensures a fair comparison of the framework's performance when executing linear displacement commands in the x, y, and combined directions. Each linear trajectory is discretised into 20 waypoints. Given the motion is repeated in both directions five times, each test generates a total of $2 \times 20 \times 5 = 200$ data points. Group C has been reserved for the case when more testing time would have been available to further evaluate whether the starting position within the laboratory would impact the tracked trajectory of the tests proposed in Group B.

Group D: Generalization Tests

The final test group associated with Case 1 is Group D, designed to evaluate the rover base's capability to execute and maintain a circular trajectory within the laboratory environment. This group of tests aims to assess the rover's ability to follow the continuous curved motion required to create a full circular trajectory as part of a Kepler trajectory. A circular path with a fixed radius of 1.2 m is defined in the horizontal plane of the laboratory, corresponding to the robot's body-centred x-y coordinate frame. This radius has been selected as in previous analysis, it became apparent that a 1 m radius will be required for the end effector of the robotic arm when replicating a 100 km orbit. For this reason, a 20% margin is applied to the corresponding radius the rover base has to execute to allow some contingency. The radius can be later adjusted as fit in Case 3 when the movement of the rover base is coordinated with the motion of the robotic arm. In this case, the trajectory is discretised into 37 equidistant waypoints, forming a complete revolution. For this group of tests, it was also relevant to evaluate whether the starting position of the motion impacts the tracking and execution performance of the system; therefore, three tests, respectively, D1, D2 and D3, have been executed, all recreating the same trajectory from different starting locations. The findings on the impact of the starting location are summarised in Section 5.6 discussing uncertainty sources.

The angular distribution is computed by uniformly sampling the interval $[0, 2\pi]$, where each angular position θ_i defines a Cartesian waypoint as:

$$x_i = r \cdot \sin(\theta_i), \quad y_i = r \cdot \cos(\theta_i), \quad z_i = 0 \qquad (5.1)$$

5.3.2. Error Metrics for the Rover Base

When assessing the rover base's performance, it is impossible to observe the command signal directly but only the resulting motion. Therefore, it is the Expected Position, which refers to the computed pose of the markers on the rover base, in the OBC frame defined in Chapter 4, corresponding to each commanded displacement that will be measured in the laboratory. Since commands are issued in the BC frame, tracking the motion for the Group A test requires a rotation of BC and an evaluation of how this impacts OBC, while Groups B and D require a pure translation.

For each test, a number of N waypoints is defined. At each point a state vector is defined as $\mathbf{s} = \begin{pmatrix} x_{OBC} & y_{OBC} & z_{OBC} \end{pmatrix}^T$. Given the measured state at s_i^{meas} at point i resulting from OptiTrack readings and the expected state computed in the simulation $\mathbf{s}_i^{exp}$, the point-wise errors are computed as:

$$\mathbf{e}_i = \mathbf{s}_i^{meas} - \mathbf{s}_i^{exp} \qquad (5.2)$$

Table 5.5: Error metrics for Chaser Rover Base Motion for the conducted test

Test ID	**Mean [mm]**			**Std. Deviation [mm]**			**RMSE [mm]**		
	x	**y**	**z**	**x**	**y**	**z**	**x**	**y**	**z**
C1_A1_spin	-15.63	-1.22	21.11	24.42	1.60	22.37	28.99	2.01	30.76
C1_A2_spin_offset1	15.35	0.21	-24.27	34.43	3.70	39.79	37.70	3.71	46.61
C1_A3_spin_offset2	14.32	1.15	-24.73	75.54	3.81	63.67	76.89	3.98	68.31
C1_B1_linear_x_repeat	218.51	-1.92	-29.20	124.69	1.45	14.47	251.58	2.40	32.59
C1_B2_linear_y_repeat	40.51	5.02	233.89	28.74	3.84	197.48	49.67	6.32	306.11
C1_B3_linear_xy_repeat	240.18	3.43	202.05	155.52	2.48	263.53	286.14	4.23	332.07
C1_D1_circle_base_start1	-26.16	6.86	195.93	369.35	6.33	276.93	370.28	9.34	339.24
C1_D2_circle_base_start2	-35.16	6.64	194.02	342.32	6.39	205.82	344.12	9.21	282.85
C1_D3_circle_base_start3	-45.77	8.87	202.50	337.32	6.64	194.60	340.41	11.08	280.85

The statistical assessment of the error between expected and measured rover base motion employs three complementary metrics: the mean, the standard deviation and the Root Mean Square Error (RMSE).

Mean error The mean ē reveals any systematic offset between commanded and actual motion. A non zero mean indicates a persistent overshoot or undershoot in the base motion. If ē remains similar across repeated trials, suggesting modelling inaccuracy. For example, an incorrect translation in the relative positioning of coordinate frames such as a constant shift equal to the difference between the camera height and the OptiTrack marker height. Such biases must be corrected in the modelling described in Chapter 4. Alternatively, mean error may arise from other factors, including sensor bias or mechanical play of the hardware used. Because positive and negative deviations can cancel out in the mean, ē alone does not fully characterise the error distribution and must be supplemented by dispersion metrics which are discussed further. The mean error is then computed as:

$$\bar{\mathbf{e}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{e}_i \tag{5.3}$$

Standard deviation The standard deviation σ_e quantifies the scatter of individual errors around the mean. A low σ_e demonstrates that the rover's motion is repeatable and predictable, even if biased, as this can be corrected. However, a high σ_e indicates irregular performance. The standard deviation is then computed according to:

$$\sigma_e = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\mathbf{e}_i - \bar{\mathbf{e}})^2} \tag{5.4}$$

Root mean square error The RMSE aggregates bias and variability into a single quantity, penalising large deviations more strongly. It, therefore, provides a sensitive measure of overall accuracy, which is especially useful in applications where occasional large positioning errors must be avoided, such as when the rover base must be used to create a fully orbital trajectory in the laboratory. Therefore, this metric provides better insight into the performance of the rover base for the selected application. The RMSE is then computed as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \mathbf{e}_i^2} \tag{5.5}$$

5.3.3. Performance Evaluation Rover Base

The resulting errors between the commanded and measured positions, as recorded by the OptiTrack system, across all group tests for Case 1 are summarised in Table 5.5. This overview informs the general system behaviour of the base motion for the command input through waypoints, as discussed in Subsection 5.3.1. Further insight into individual scenarios is obtained by comparing commanded and measured trajectories in Figures 5.3 to 5.11. The following observations can be made about the rover base performance. Accuracy is measured axis-dependent, and the type of trajectory influences performance, with longer distances recording larger errors. Based on the observations, it is then attempted to distinguish the contributions of the software modelling from the hardware towards the error build-up. Lastly, at the end of the section, a discussion on the influence of these findings on creating trajectory-motion profiles is presented.

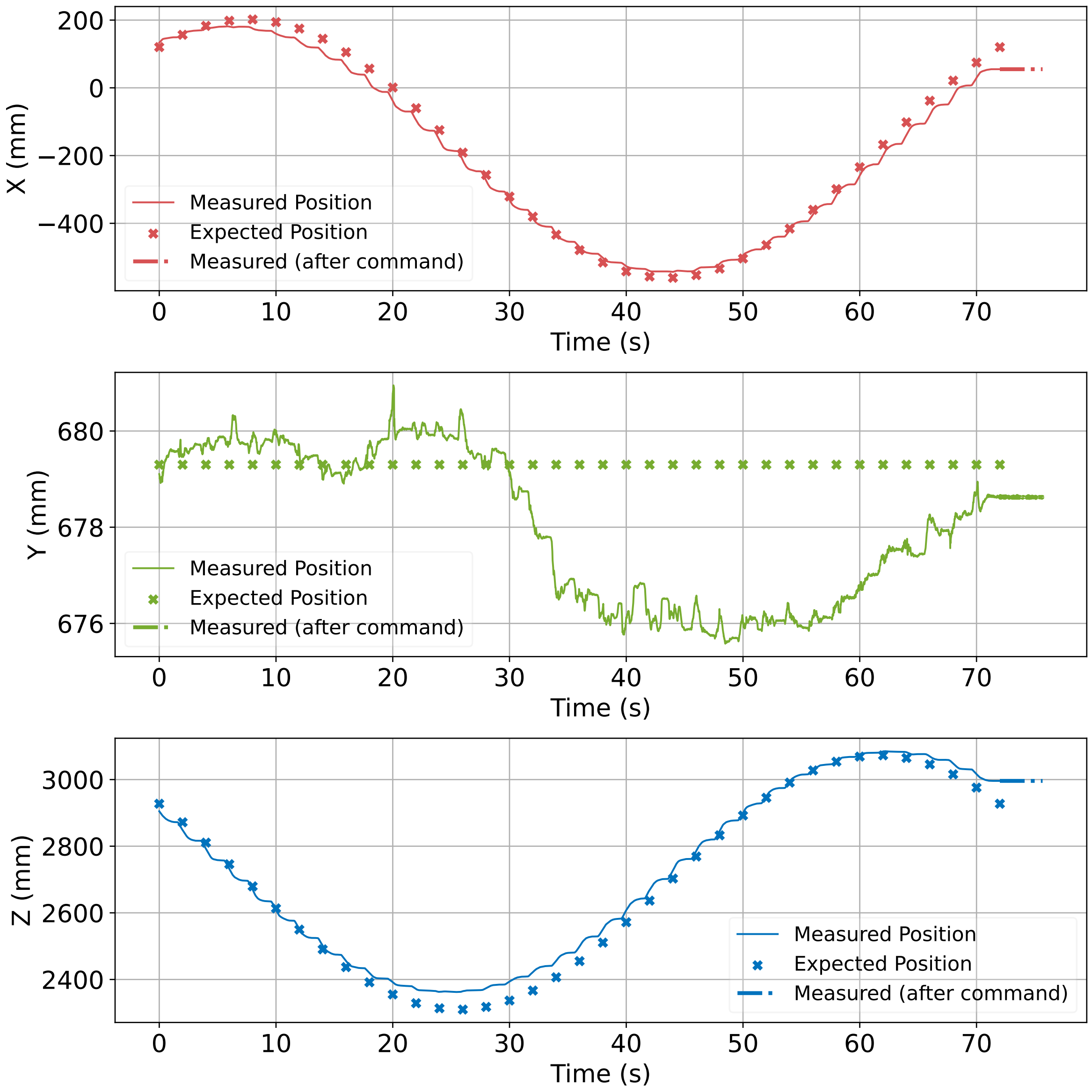


Figure 5.3: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_A1_spin)

Axis-dependent accuracy Errors along the vertical (y) axis, which corresponds to the marker height on the rover base, remain consistently below 10 mm across all tests. Performance along the laboratory length (z) is marginally worse than that along the width (x).

- **Vertical axis (y).** The mean and root-mean-square errors never exceed 10 mm. The small vertical error confirms that the height of the marker cluster with respect to the laboratory floor in the y-direction is modelled correctly. The remaining residual bias, as presented in Table 5.5, can be attributed to the OptiTrack calibration. Despite the initial marker pose being imported directly from the calibration measurement of the OptiTrack and used in the simulation to compute the expected position of the markers throughout the experiment, it has been observed that during different motion profiles, the recorded position of the height of the markers is also affected. This coupling between lateral motion and apparent vertical displacement is further analysed in Section 5.6

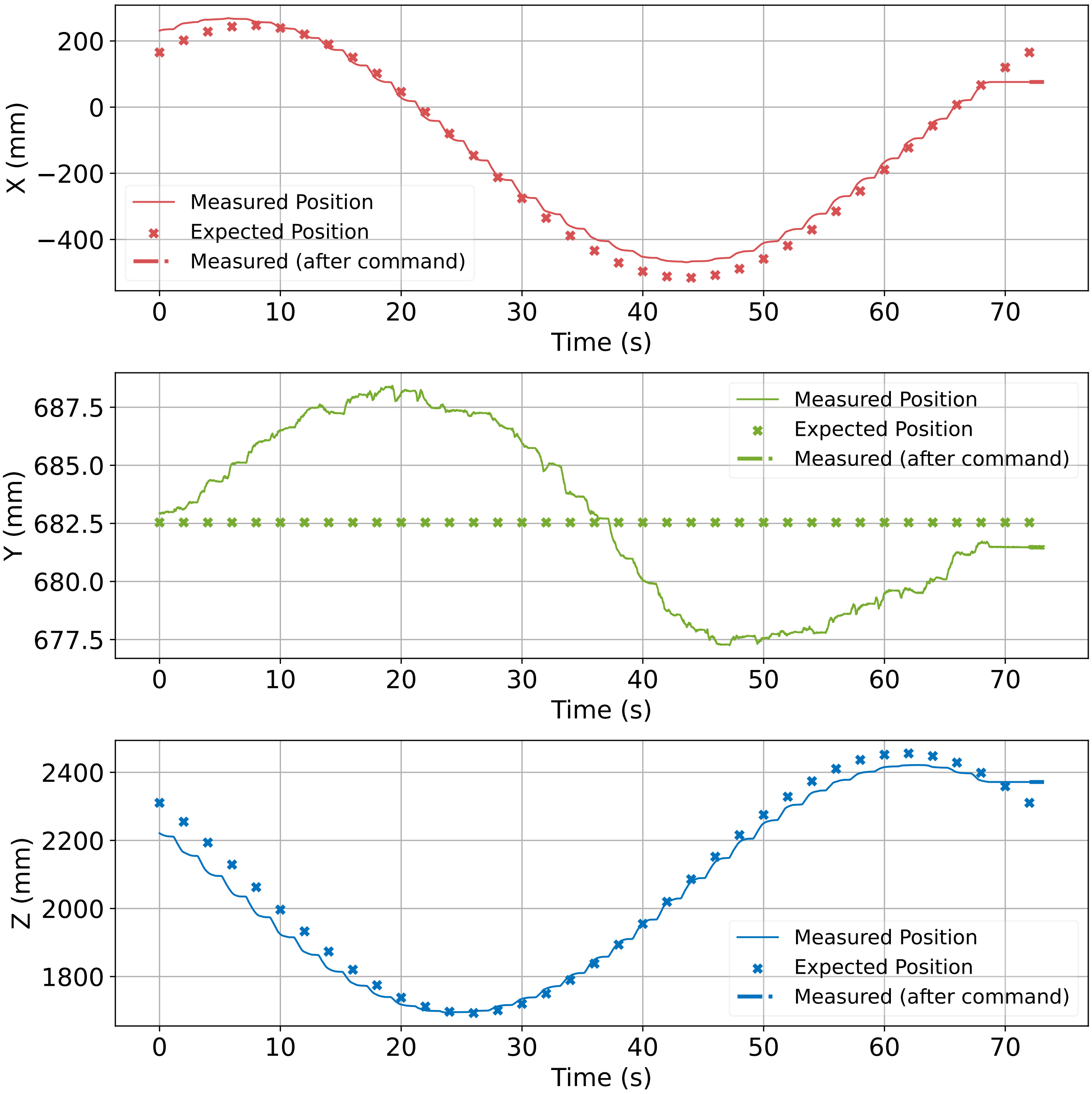


Figure 5.4: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_A2_spin_offset1)

- **Horizontal axes (x and z).** The largest deviations have been recorded for the horizontal biases. The lack of repeatability makes it difficult to evaluate where the potential sources of errors may stem from. Group A, for which the repeatability of angle commands has been analysed, has had similar performance across all three tests, with mean errors of up to 16 mm (x) and 25 mm (z). The recorded biases in this case may be attributed to either an initial misalignment during the manual command mode of the rover base or due to the slip of the wheels during the motion profile. Group B and Group D show increased standard deviations. Thus, dispersion grows with trajectory length, indicating that the errors increase over distance. This behaviour suggests implementing a closed-loop correction after at least every 0.1 m of travel to prevent error propagation along the trajectory.

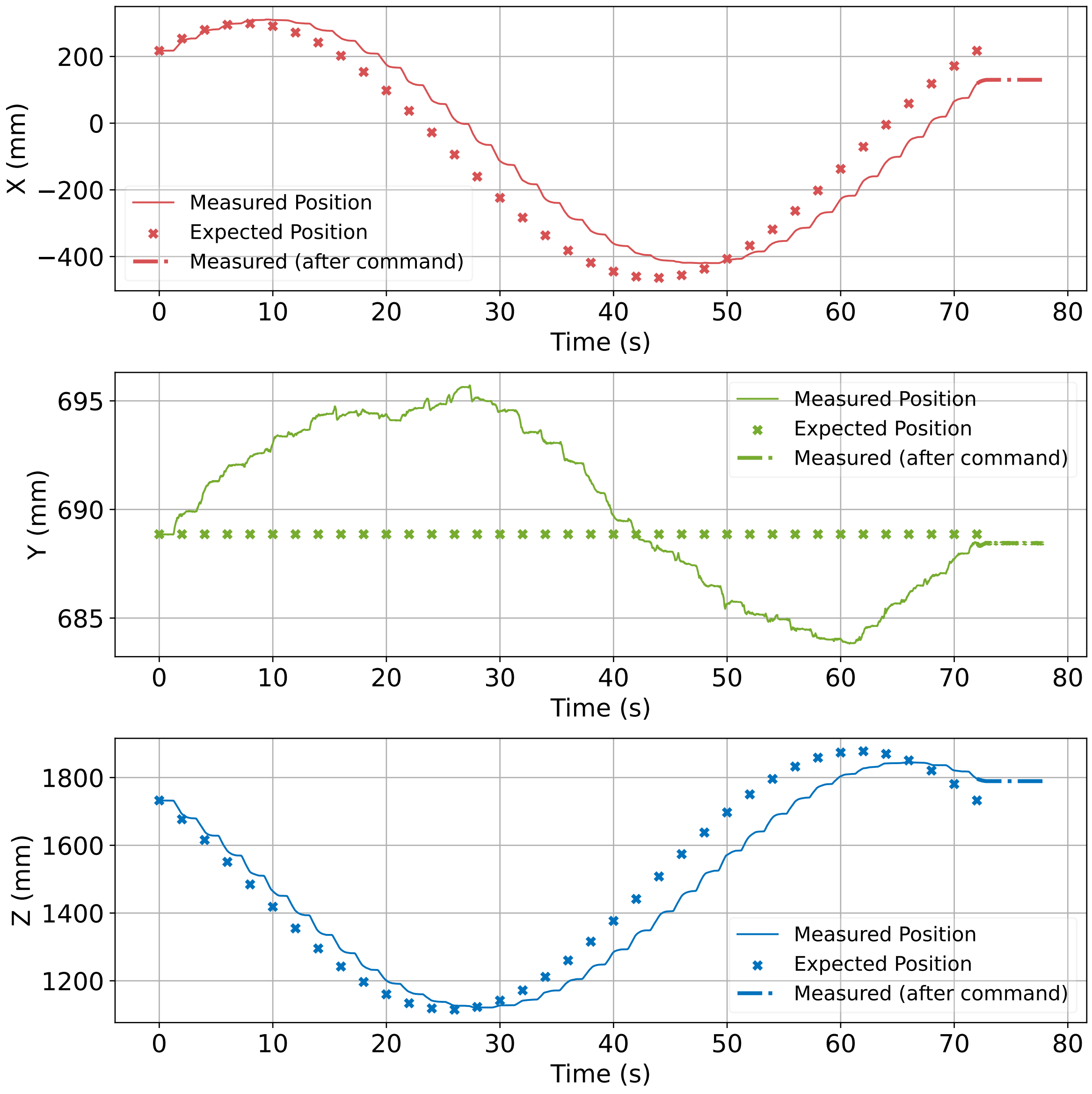


Figure 5.5: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_A3_spin_offset2)

Effect of trajectory type

Group A: Pure rotation (C1_A1–A3). Horizontal means remain within ±25 mm The recorded errors reveal that the body frame BC may have been rotated with respect to the laboratory axes which also explains the increased dispersion measured by the standard deviation and RMSE. The rover base falls short of completing the last command input for every test across all three tests. The reason why the last target is not reached may be an indication of the build-up of errors throughout the experiment and the slip of the rover wheels on the floor of the laboratory.

(a) **Systematic yaw bias - from initial calibration** The misalignment originates from the manual placement of the base at the start of each experiment. A 10° yaw offset between BC and the laboratory frame explains the non-zero horizontal means because each commanded rotation is interpreted in the misaligned body frame. Therefore, the bias propagates through the entire 37-step sequence and manifests as a uniform lateral displacement of the marker cluster.

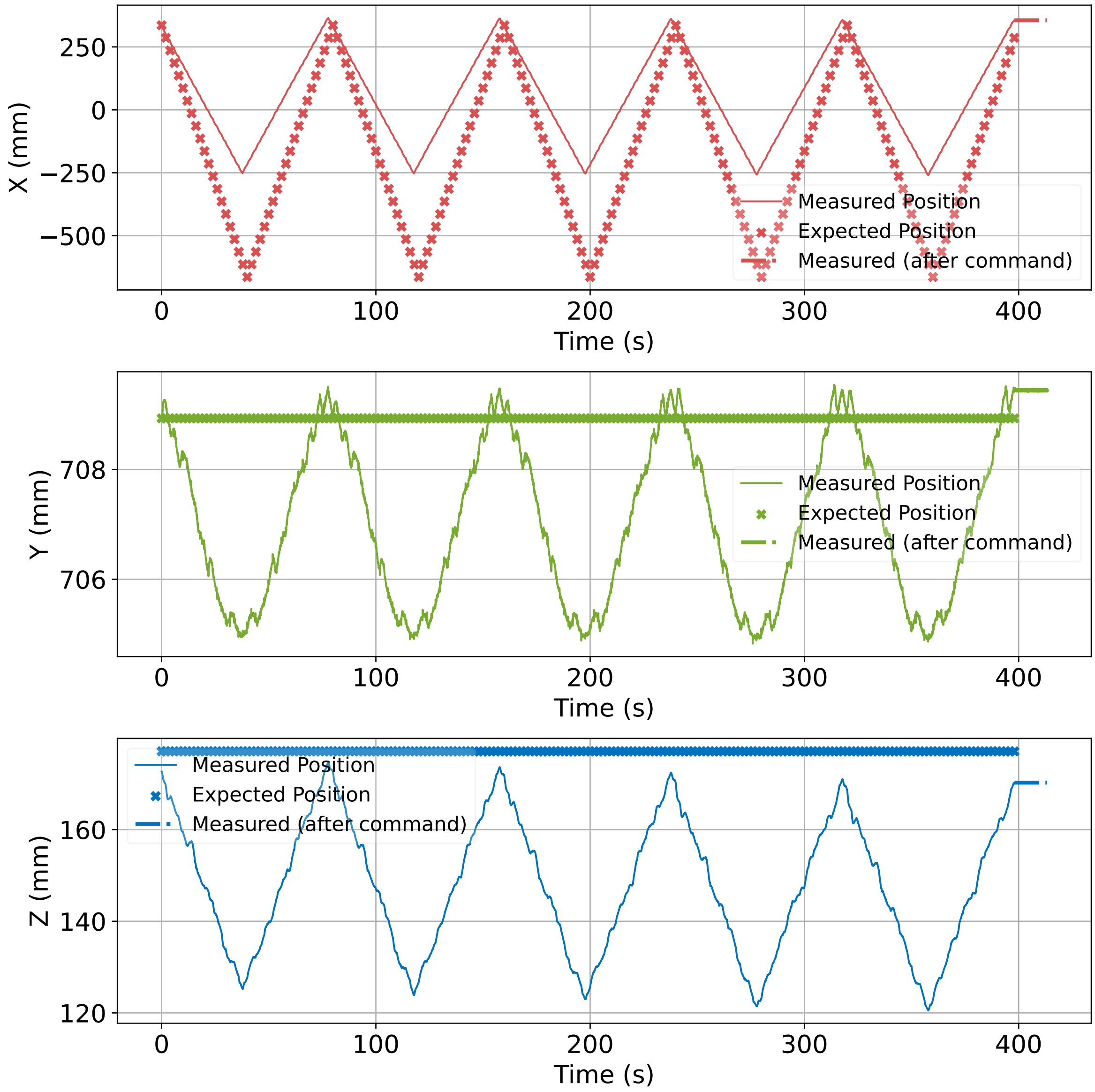


Figure 5.6: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_B1_linear_x_repeat)

(b) **Cumulative wheel slip** Static friction prevents the wheels from tracking this distance precisely and the discrepancy integrates over the 37 executed steps, so the final step is effectively unreachable.

To mitigate both effects, the next experimental campaign should (i) introduce an automated yaw alignment routine for the rover base and (ii) increase the incremental rotation to evaluate whether the cumulative wheel slip is present with the same order of magnitude also when the velocity profile is not limited due to the use of waypoints.

With an automated alignment routine that improves the rover base's parallel positioning to the laboratory's walls, the resulting errors should be closer to zero. Should that be the case, the next test could purposely induce a positive 10° rotation around the vertical axis of BC to evaluate if the reported positional errors match the ones presented in this work. Should both conditions be achieved, the hypothesis that the initial misalignment of the rover base is the leading cause of systematic yaw is confirmed.

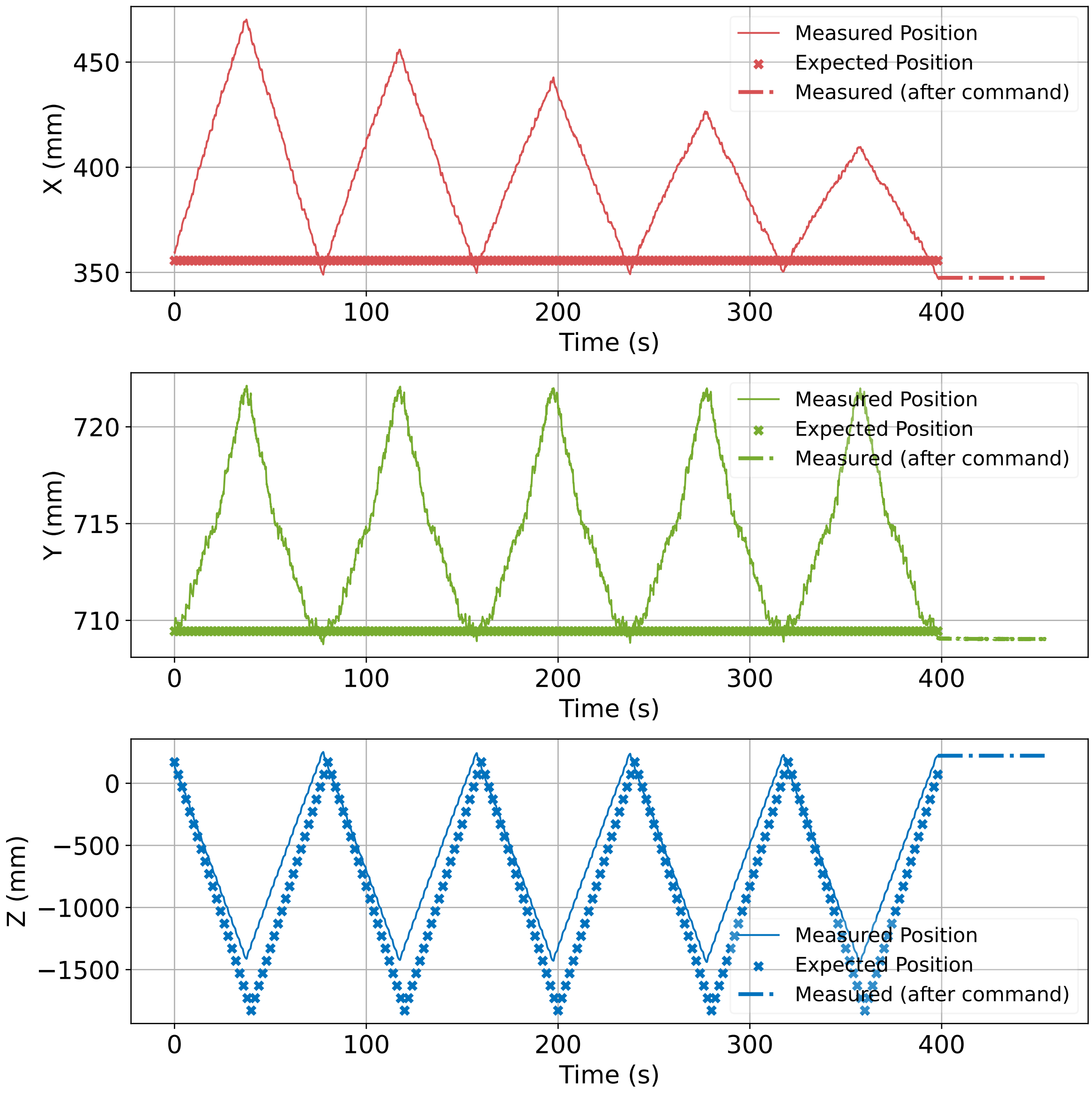


Figure 5.7: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_B2_linear_y_repeat)

Testing for wheel slip in the omnidirectional robots is particularly difficult. Previous work has decoupled the longitudinal slip from the lateral slip. For testing longitudinal slip, additional information from the rover base can be utilised, such as data from the internal encoder of the wheels (Reina, 2010), which provides information on wheel rotation. By modelling the velocity of each wheel, the expected values could be compared with data from the experiment. Here, the longitudinal command tests, such as C1_B1, could be recorded, and information from the sensors of the rover base can be gathered for this analysis. If variations in the recorded longitudinal velocities of wheels placed on the same side of the rover are recorded, the presence of longitudinal wheel slip is confirmed.

For lateral wheel slip, visual inspection methods are proposed to monitor the wheel traces of the rover base (Reina, 2010). In the case of the GNC laboratory, this approach would damage the ground surface and is therefore not recommended. One easier option would be to define start and end target locations for the rover base and asses the shift in position along

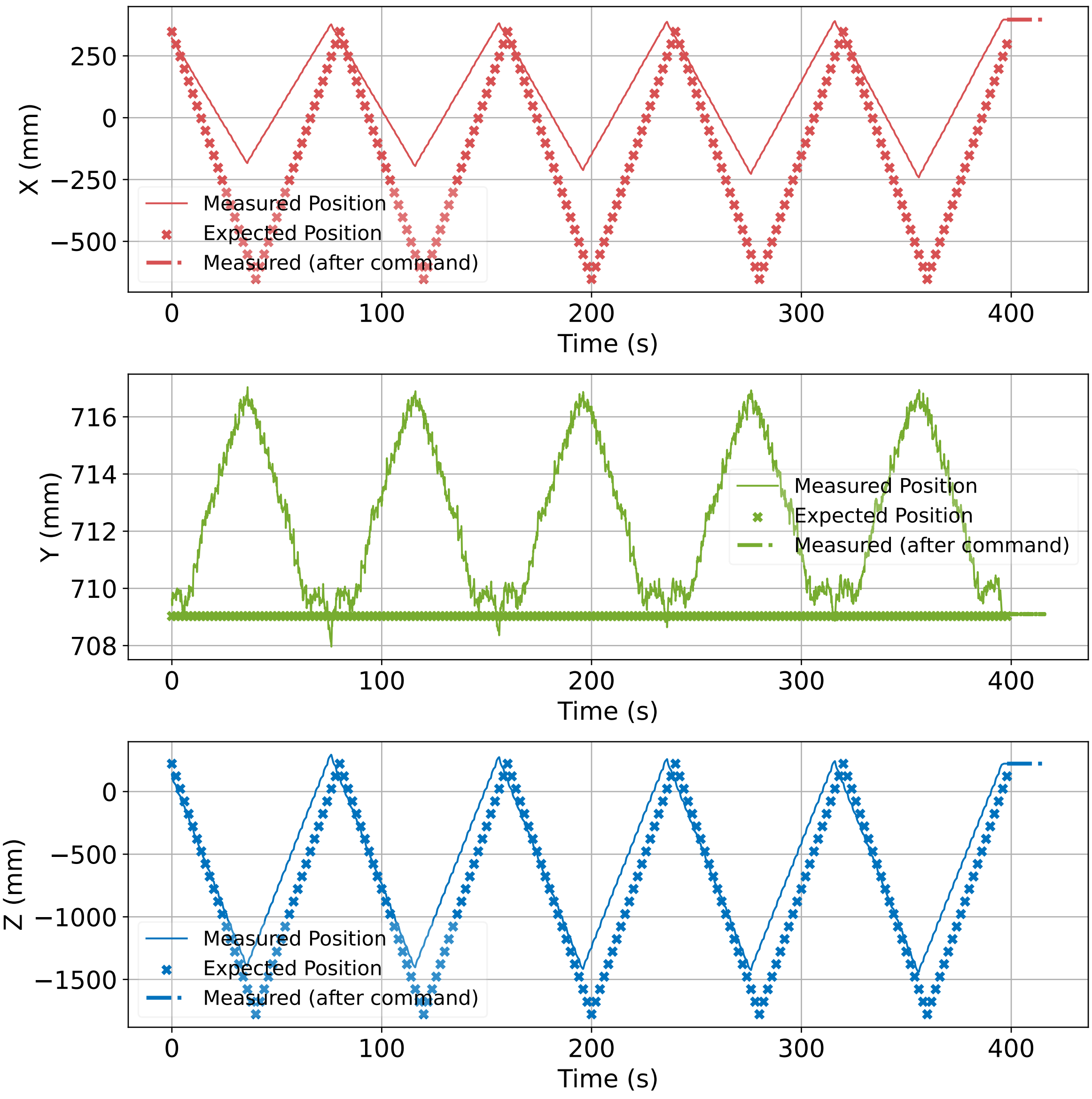


Figure 5.8: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_B3_linear_xy_repeat)

the length of the laboratory at the end of the trajectory. Should that occur, for a trajectory along the length of the laboratory, the lateral slip in the wheels of the rover base is confirmed.

Gorup B: Linear translation (C1_B1–B3). Systematic undershoot of target location dominates these tests. For a ±1 m command the base stops short by roughly 40% of the requested distance. Standard deviation in z climbs to 264 mm; the value is consistent with the expected lateral slip while steering the platform once lateral forces exceed the static friction threshold.

Despite the bias, the reverse commands return the base to the exact physical location, confirming that the wheel odometry integrator is internally consistent. Therefore, if further analysis is conducted on the type of motion commands that would remove or reduce the skid of the robotic base on the floor of the GNC lab, the rover base can later be used for creating motion profiles. Based on this initial performance analysis, it is believed that the additional corrections are required to prevent the positional drift seen along (z) axis (Figure 5.6) and (x) axis (Figure 5.7)

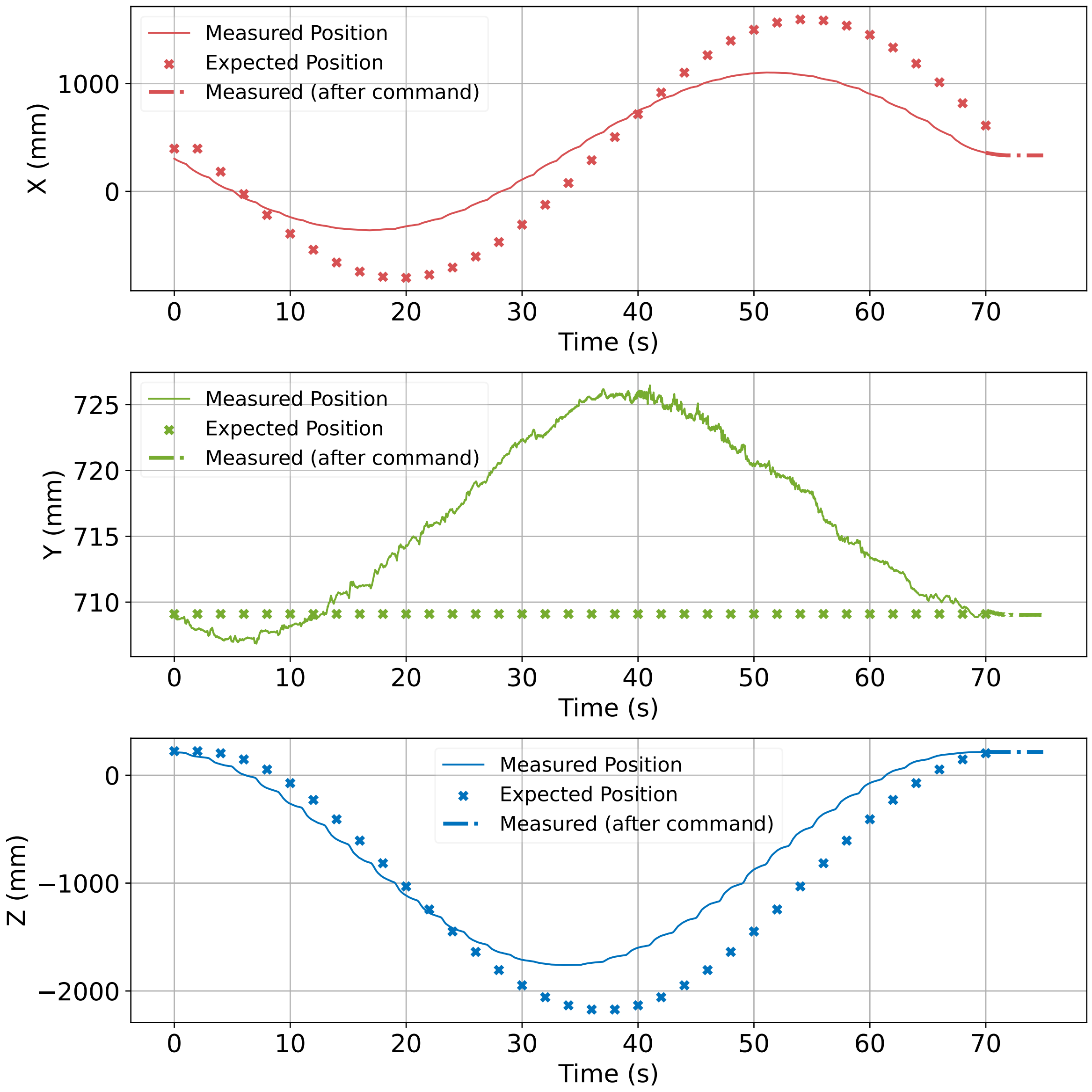


Figure 5.9: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_D1_circle_base_start1)

In test C1_B1_linear_repeat_x, the initial calibration appears successful as x and z values match the initial coordinates; however, despite a zero longitudinal command, the base is translating laterally. While the oscillation itself is relatively small at about 60 mm, the base continues drifting laterally throughout the test. This behaviour is also observed when only lateral commands are given Figure 5.7. The drift could indicate the wheels are skidding on the laboratory surface, making trajectory execution with small control inputs more difficult. When combining movement in the planar xy direction, the system overall behaves similarly with x and y as when individual commands were issued (Figure 5.8). The fact that the behaviour repeats when joint commands for x and y are provided, as when individual commands are given, confirms that the system's command repeatability is good. However, the expected command output does not yield the full expected movement and therefore warrants further investigation.

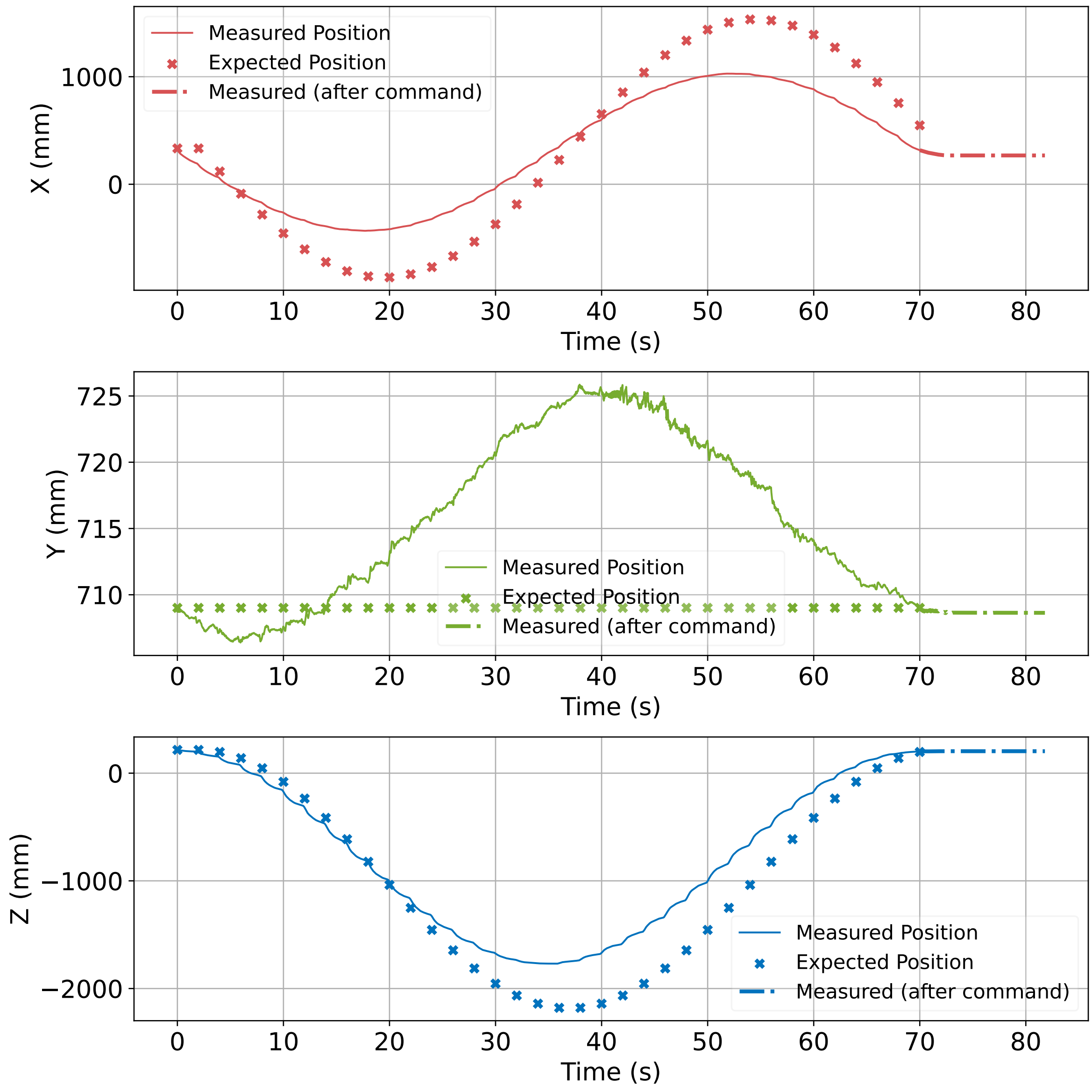


Figure 5.10: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_D2_circle_base_start2)

The systematic undershoot is hypothesised to originate from two coupled mechanisms:

(a) **Wheel floor interaction** Visual inspection during the tests revealed intermittent wheel slip, especially when the base reversed direction. Slip shortens the effective stride length so that the true travel distance is less than the commanded distance. A dedicated traction test will be required to quantify the static friction coefficient of the laboratory floor wheels and verify whether that value alone explains the observed undershoot. While in practice, the flooring of the GNC laboratory cannot be changed, as it has been painted and adjusted such that it can correctly mimic the dark conditions of space, it would still be beneficial to understand whether this type of coating poses limitations on the use the rover base. Testing the wheel floor interaction is performed in the proposed test for longitudinal and lateral slip assessment proposed in Group A.

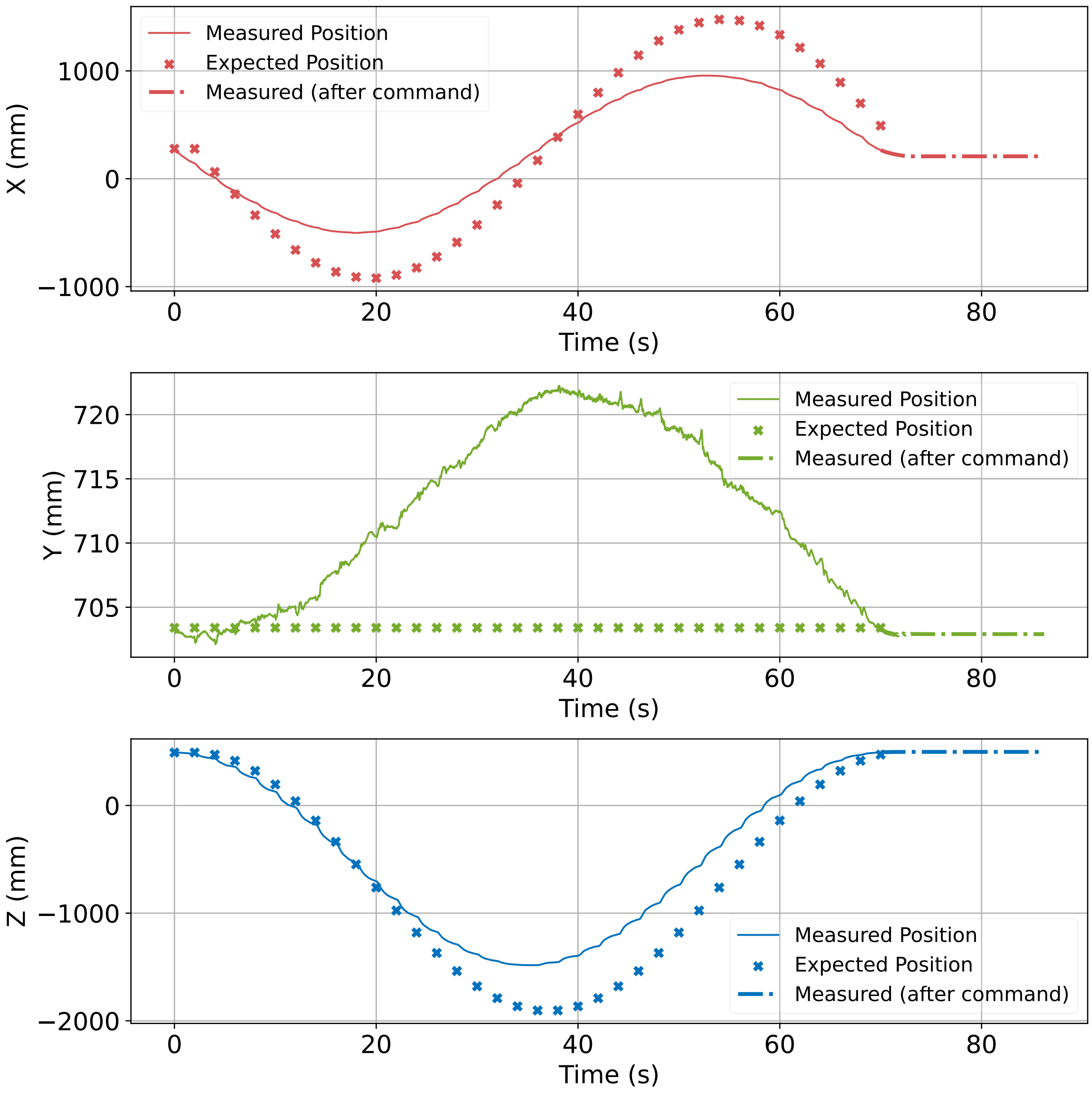


Figure 5.11: Expected versus measured movement of the Base Chaser OptiTrack markers (test C1_D3_circle_base_start3)

(b) **Integrator error in the low-level controller of the RB-Kairos** While the wheels are slipping, the wheel encoder count, responsible for recording how far each wheel has turned during the motion, does not increase. Therefore, this could be the reason why the rover base performed the same motion profile despite undershooting the target position.

Group D: Circular motion (C1_D1–D3). Deviations reach 45 mm (x) and 202 mm (z). The only velocity documented by the manufacturer is the maximum speed 1.5 m/s (Section 3.2). The path planner using discretisation in waypoints assumes that the differential kinematics used internally by the RB-Kairos to create the velocity commands is capable of handling distance of 100 mm to be executed within 2 seconds as the velocity stays well below the maximum. However, the actual wheel velocities appear to be capped. The low-level controller is believed to truncate the velocity profile at each waypoint. This motion profile is also affected by the same factors causing the undershoot in reaching the target location as Group B.

Distinguishing software and hardware contributions

It is essential to distinguish between errors arising from the software model and those imposed by hardware limitations. To this end, the system behaviour across all tests must be examined. The consistently small vertical errors confirm that the coordinate frame setup and alignment shown in Chapter 4 are well defined in the y–direction. Nonetheless, the OptiTrack system exhibits a location-dependent bias in height measurements within the laboratory volume (Section 5.6). Consequently, the pure rotation cases (C1_A1–A3), which involve minimal horizontal displacement, report the lowest vertical error, as the rover base has minimal translation across the volume of the laboratory. The vertical variation at different points in the laboratory space is a hardware limitation that must be evaluated further. In the simulation software, the assumption that the height of the markers will remain constant is not valid and has to be accounted for.

By contrast, the larger horizontal biases, evident in the non-zero mean errors, point to a residual miscalibration of the base orientation, which is assumed to be parallel to the laboratory walls. Specifically, the initial manual alignment yields displacements of approximately 15 mm along the x–axis and 25 mm along the z–axis, consistent with a rotation of roughly 10° about the base z–axis. Such misalignment causes the OptiTrack markers, mounted at the base corners, to record shifted positions compared to what is expected. This error is caused by an assumption made in the software that the frames can be aligned, which has to be improved by improving the calibration approach for the placement of the hardware.

Furthermore, the large scatter in horizontal error with a standard deviation up to 280 mm for a 1.2 m circular path indicates that the current kinematic and dynamic model does not capture the actual hardware behaviour. Although the manufacturer's specifications guarantee that the commanded position changes and timings yield feasible velocity profiles (Section 3.2), the rover base repeatedly fails to reach the target location. In the rotation tests (C1_A1–A3, see Figures 5.3-5.5), the number of motion commands is executed correctly, yet the system accumulates a residual error that prevents a complete 360° revolution. The residual error and incomplete target motion is a hardware-induced error, which is believed can be reduced by creating a closed-loop system in the GNC laboratory, which includes the positional error of the robotic system in the command generation process.

Implications of errors of the robotic base for subsequent cases

Because the RB-Kairos base undershoots translational commands and accumulates orientation error during curved motion, open-loop position control is insufficient for creating a full Keplerian path required in Case 3. While it has been attempted to evaluate whether the current modelling framework can use the command of the robotic arm to compensate the error of the robotic base and still reach the target waypoints of the Keplerian orbit using the end effector, it is clear that a closed-loop control loop is required for the robotic system to further improve its suitability for this type of experiments.

5.4. Case 2: Static Base, Mobile Arm

This test case evaluates the movement of the robotic arm when the rover base remains stationary. To generate the desired motion, a set of Cartesian waypoints is first defined in the laboratory frame L, starting from the current position of the robotic arm determined using the OptiTrack markers. These represent the trajectory to be followed by the robotic arm's end-effector. Each Cartesian point is then transformed into the arm mount frame AMC, which requires knowledge of the transformation between the lab reference frame and the arm mount explained in Section 4.2. Once the pose (position and orientation) of the end-effector is defined relative to the mount, an inverse kinematics algorithm is used to compute the corresponding joint angles that will achieve that pose.

5.4.1. Selection of the inverse kinematics solver

Three numerical optimisation methods are commonly employed to solve the Inverse Kinematics (IK) problem: Levenberg–Marquardt (LM), Gauss-Newton (GN), and Newton–Raphson (NR). Among these, LM was selected for this work due to its robustness and rapid convergence, particularly when a reliable initial guess is provided. For the considered robotic system, the current joint angles of the manipulator can be used as an initial guess for the solver. Joint positions may be obtained by querying the robot's current joint states via the command shown in Subsection 5.4.1. Additionally, a control node can be implemented to command the robotic arm to move to a predefined set of joint angles. In such cases, the onboard controller of the robotic arm automatically generates the joint trajectory required to transition from the initial configuration to the final one. Therefore, the same initial configuration can be reused as part of a calibration step. For computing the trajectory and the joint angles at each waypoint, the trajectory is defined as a sequence of evenly spaced out points. The solution obtained at each step is used as the initial guess for the following step, significantly improving the solver's stability and performance. The following command, issued in the robot's terminal, retrieves the current joint positions. The output provides the joint names and their corresponding positions, which can be directly used as the initial guess for the IK solver.

Command to retrieve joint states of UR16e

```
rostopic echo /robot/arm/joint_states
```

```
header:
  seq: 123
  stamp:
    secs: 123456
    nsecs: 789000
  frame_id: ''
name: ['shoulder_pan_joint', 'shoulder_lift_joint',
         'elbow', 'wrist1', 'wrist2', 'wrist3']
position: [1.57, -0.523, -2.617, 1.57, 1.57, 0.0]
velocity: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
effort: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

Listing 5.1: Output of `rostopic echo /robot/arm/joint_states` the values shown for position can be retieved and used for the inital guess of the inverse kinematics solver

For both linear and planar (xy) displacement tests, the average computation times of the Levenberg-Marquardt (LM), Gauss-Newton (GN), and Newton-Raphson (NR) methods were comparable, with the GN method occasionally achieving slightly faster solutions (see Table 5.6

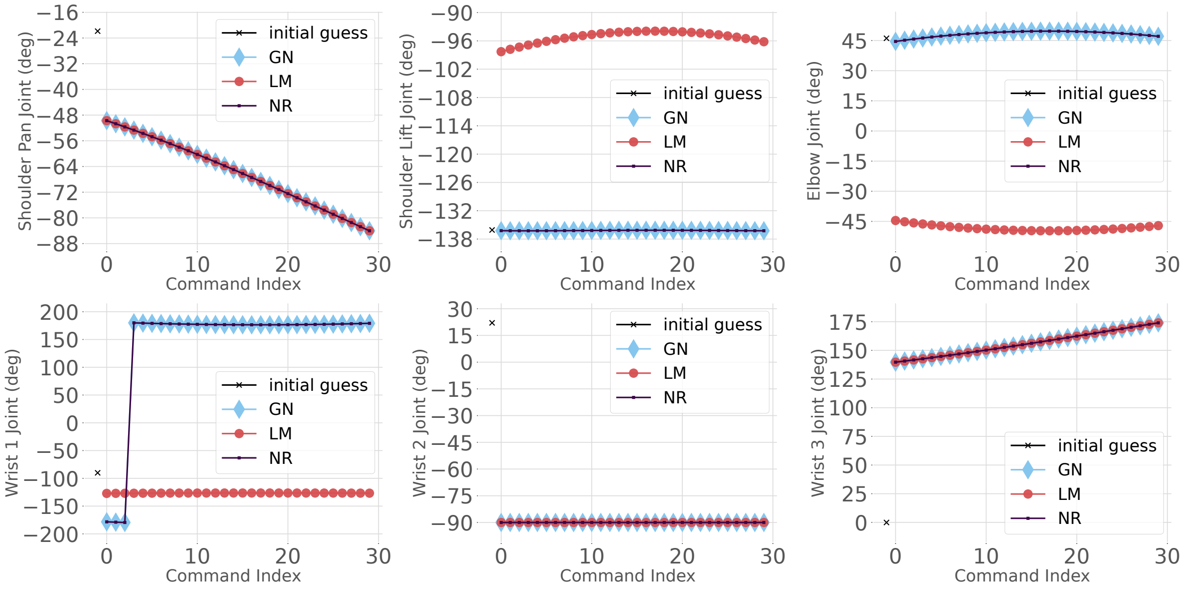


Figure 5.12: UR16e joint solutions for inverse kinematics solvers over a 0.3 m displacement along x axis with 30 waypoints

and Table 5.7). However, LM was selected as the default solver for all subsequent experiments due to its established robustness and convergence properties in the robotics literature and toolboxes (Corke, 2017). The LM algorithm adaptively interpolates between the fast convergence of GN and the global stability of gradient descent. It is significantly more reliable for ill-conditioned or poorly initialised problems. While the performance difference was negligible in these well-conditioned test cases, the improved reliability and community adoption of LM justified its use in this work. All three solvers have been implemented to solve for the joint commands associated with two different linear trajectories, respectively 0.3 m displacement along the x-axis with 40 waypoints (Figure 5.12) and (Figure 5.13)

Table 5.6: Timing statistics for inverse kinematics solvers over a 0.3 m displacement along x-axis with 40 waypoints

Solver	Mean Time per Call [ms]	Std [ms]	Max [ms]	Total [ms]
LM	132	59	337	3942
GN	124	40	275	3722
NR	136	52	263	4071

Table 5.7: Timing statistics for inverse kinematics solvers over a planar 0.3 m 0.12 m xy displacement trajectory with 40 waypoints

Solver	Mean Time per Call [ms]	Std [ms]	Max [ms]	Total [ms]
LM	105	25	189	3141
GN	119	58	372	3596
NR	119	53	301	3578

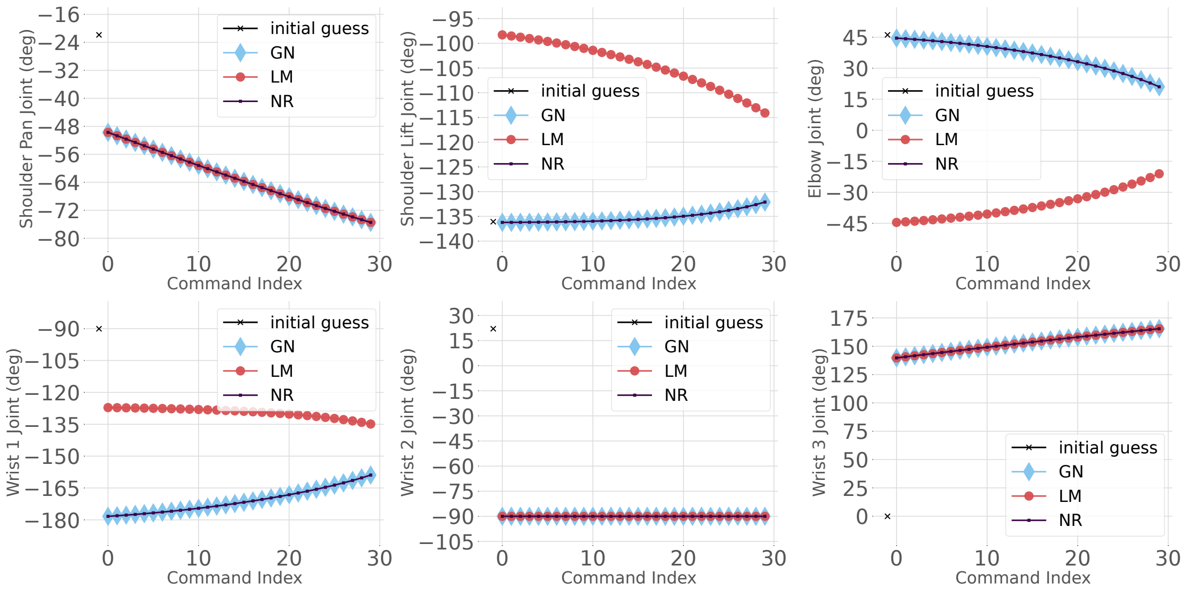


Figure 5.13: UR16e joint solutions for inverse kinematics solvers over a planar 0.3 m 0.12 m xy displacement trajectory with 30 waypoints

5.4.2. Creating the control input

The control input for Case 2 consists of joint angle commands. The base velocity is set to zero at every waypoint. The steps used to create a command sequence:

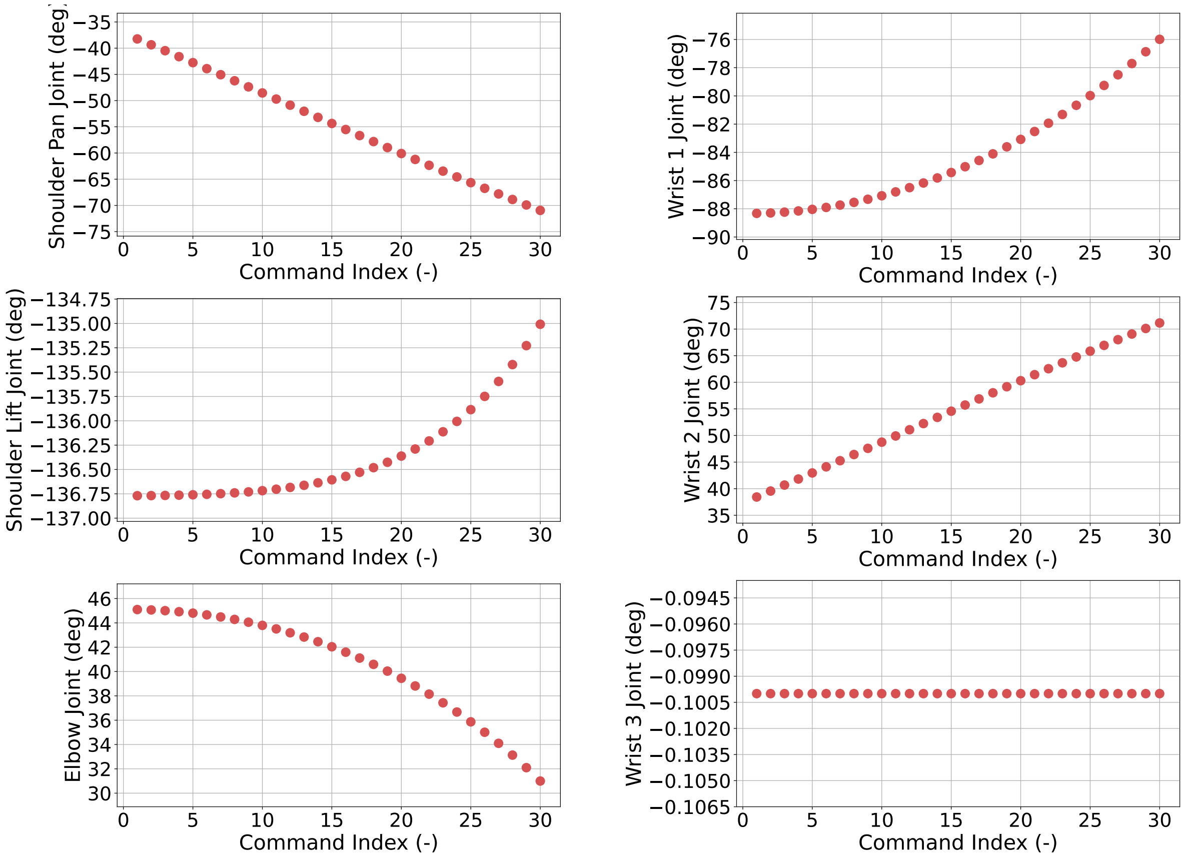
1. Generate the desired Cartesian waypoints in the laboratory frame L.
2. Transform each waypoint to the arm-mount coordinate frame AMC with the homogeneous transforms via the reference frame chained summarised in Table 4.1.
3. Define the pose of the end effector as a homogeneous transformation, as the position is the particular waypoint and corresponding orientation.
4. Solve the inverse kinematics for the pose set in Step 3 with the LM algorithm to obtain $\mathbf{q} = [q_1, \ldots, q_6]^T$.
5. Write the command index, set base commands to zero, append joint angle commands and camera-trigger flag to `inputFile.csv`.

Joint Commands are then visualised to verify smoothness and ensure no joint limits are violated before hardware execution.

Three test groups are defined. Group A commands straight-line translations, Group B prescribes circular paths, and Group C adds a pointing constraint that keeps the target centred in the end-effector camera. In Groups A and B, the end-effector orientation is frozen at the calibration orientation while in Group C, it is updated at every waypoint.

Group A: Linear End-Effector Motion

The robotic arm executes a linear trajectory of 0.3 m. The segment is discretised into N = 30 waypoints. The orientation of the end-effector is kept constant throughout the motion. As previously explained, the initial joint configuration of the robot arm can be retrieved from the robot's command line. For this purpose, an initial joint configuration has been established for the robot, which must always be commanded before starting the experiment. The initial guess for the inverse kinematics is this particular joint configuration: -0.381583, -2.375055, 0.805184, -1.571721, 0.385073, -0.001746 rad for the Base, Shoulder, Elbow, Wrist 1, Wrist 2 and Wrist 3, respectively. Using a kinematic function of these joint values makes it possible to compute the associated end effector position and orientation as homogeneous transformation. The resulting orientation is then commanded as the goal for the end effector along each waypoint.



(a) Part 1

(b) Part 2

Figure 5.14: Joint commands for test C2_A3_line_xy

The rotation matrix defining the orientation can then be maintained constant to ensure the focus remains solely on translational accuracy. The translational motion is discretised in waypoints.

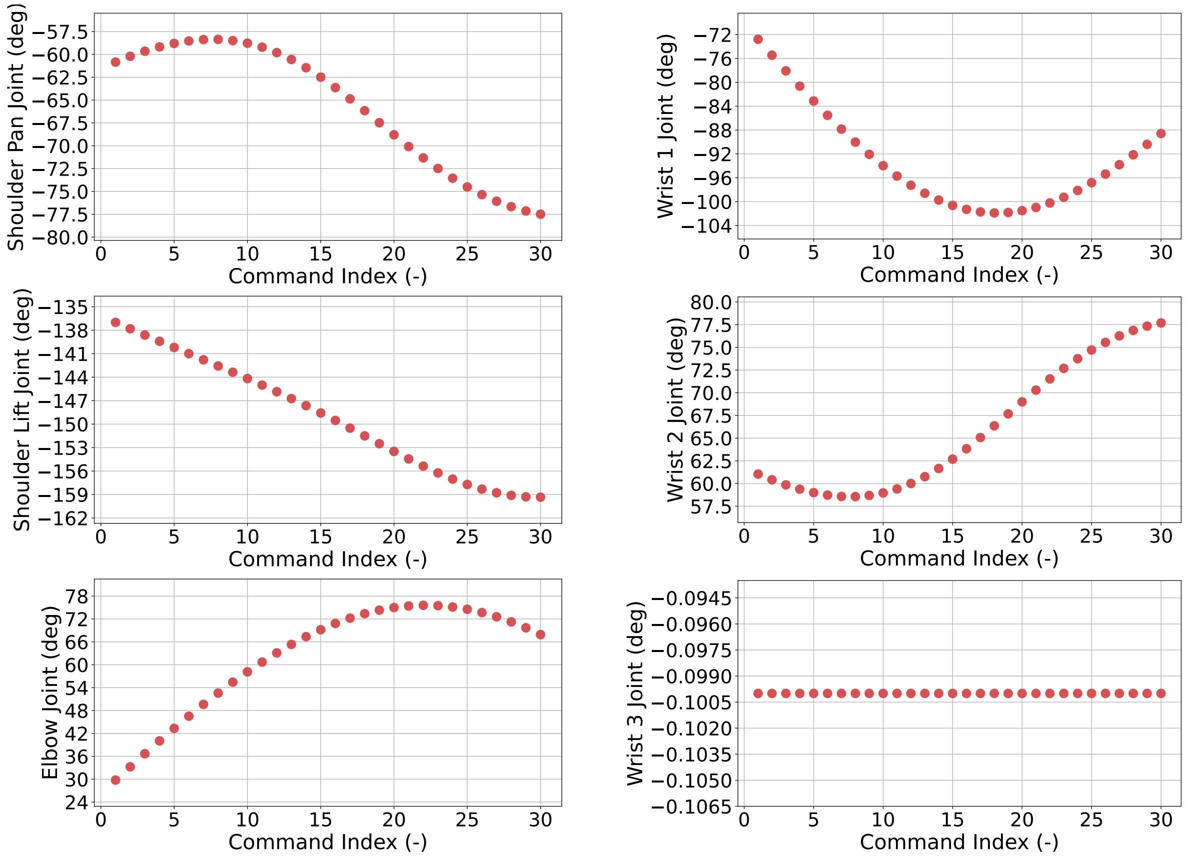
The k-th waypoint is defined as:

$$\mathbf{p}_k = \mathbf{p}_{\text{start}} + \frac{k}{N-1}\left(\mathbf{p}_{\text{end}} - \mathbf{p}_{\text{start}}\right), \qquad k = 0, \ldots, N-1. \tag{5.6}$$

The three variants for this group are:

- C2_A1_linear_x: 0.30 m along the negative laboratory x-axis;
- C2_A2_linear_y: 0.12 m along the negative laboratory y-axis;
- C2_A3_linear_xy: 0.20 m along –x and 0.12 m along –y simultaneously.

Joint angle evolution for each test can be visualised for each of the UR16e joints shown in Figure 3.4. Figure 5.14 shows the joint behaviour for test C2_A3_line_xy when the robotic arm has to create a linear trajectory. Figure 5.15 shows the behaviour of the robotic arm for creating an inclined half circle corresponding to test C2_B5_half_circle_xyz. Lastly, Figure 5.16 shows the joint behaviour when tacking is required in a horizontal half circle similar to what is required for the creation of a half orbit. One main difference to be observed while comparing the behaviour across the three different movements is with respect to Wrist 3, which in the first two tests maintains almost constant commands for the joints. That is no longer valid for test C2_C2_track_target_circle. In addition, the symmetry of the trajectory, such that the robotic arm creates a smooth half-orbit trajectory, is also reflected in the joint behaviour, where the commanded joint is symmetric around the half point of the trajectory.



(a) Part 1 **(b)** Part 2

Figure 5.15: Joint commands for test C2_B5_half_circle_xyz

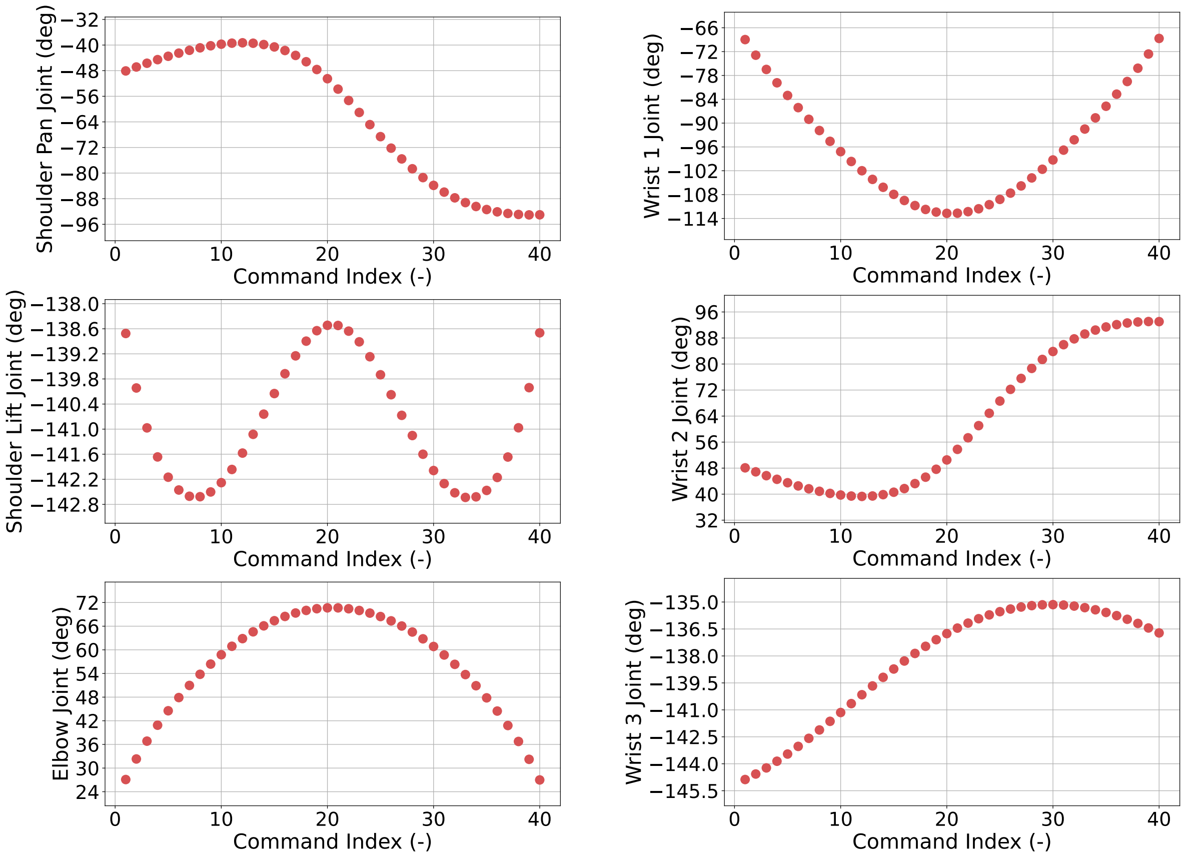
Group B: Circular End Effector Motion

Five modes were defined; only the half-circles were executed on the hardware.

- **C2_B1_circle_above_mount**: full circle, r = 0.5 m, centred 0.7 m above the mount;
- **C2_B2_circle_around_chaser**: full circle, r = 0.2 m, centred on the initial chaser position;
- **C2_B3_half_circle_xy**: half-circle, r = 0.2 m, in the xy-plane (Equation 5.8);
- **C2_B4_half_circle_yz**: half-circle, r = 0.1 m, in the yz-plane (Equation 5.9);
- **C2_B5_half_circle_xyz**: half-circle, r = 0.1 m, in a plane rotated 45° about y (Equation 5.10).

The full circle modes (C2_B1 and C2_B2) were implemented and validated in simulation but discarded for hardware tests, as they do not be able to reproduce the rotation around the asteroid. The three half circle modes were executed in both simulation and experiment. For the half-circles, the waypoints are defined as follows, where radius r is defined for each case, **c** represents the centre expressed in the AMC frame and θ_i is defined according to:

$$\theta_i = -\pi + \frac{(i-1)\pi}{N-1}, \qquad i = 1, \dots, N, \quad N = 40. \tag{5.7}$$



(a) Part 1

(b) Part 2

Figure 5.16: Joint commands for test C2_C2_track_target_circle

$$\mathbf{p}_i^{\mathrm{AMC}} = \begin{bmatrix} c_x + r\cos\theta_i \\ c_y + r\sin\theta_i \\ c_z \end{bmatrix} \qquad \text{(C2\_B3)} \tag{5.8}$$

$$\mathbf{p}_i^{\mathrm{AMC}} = \begin{bmatrix} c_x \\ c_y + r\sin\theta_i \\ c_z + r\cos\theta_i \end{bmatrix} \qquad \text{(C2\_B4)} \tag{5.9}$$

$$\mathbf{p}_i^{\mathrm{AMC}} = \mathbf{C}_y(45^\circ) \begin{bmatrix} r\cos\theta_i \\ r\sin\theta_i \\ 0 \end{bmatrix} + \mathbf{c} \qquad \text{(C2\_B5)} \tag{5.10}$$

Group C: Target Tracking

In this scenario, the end-effector must follow a trajectory while continuously orienting its z-axis toward a fixed target in space $\mathbf{t} \in \mathbb{R}^3$. This orientation imposes an additional constraint on the inverse kinematics problem: the end-effector orientation must dynamically update to point toward the target at each waypoint.

- C2_C1_track_target_line: End-effector moves in a straight line while pointing toward the target.
- C2_C2_track_target_circle: End-effector follows a circular path around the target while pointing toward the target.

Joint angle trajectories generated for linear and circular motions are shown in Figure 5.14 and Figure 5.15 respectively. These plots demonstrate that the commands remain within joint limits and evolve smoothly, indicating successful IK convergence. The remaining tests are shown in Appendix D to validate that the commanded joints have stayed within limits and had smooth behaviour across all tests conducted in this work.

Let **p** be the waypoint position and **t** target coordinates expressed in the laboratory frame L. The desired chaser frame, which defines the position and orientation of the camera C, is constructed as follows.

1. Determine target direction axis. The body z-axis of C is the unit vector that looks from the end-effector towards the target:

$$\hat{\mathbf{z}}_C = \frac{\mathbf{t} - \mathbf{p}}{\|\mathbf{t} - \mathbf{p}\|}. \tag{C.1}$$

2. Set the vertical axis. The vertical axis is chosen to match the vertical axis of the laboratory. It is important to ensure that the axis is not collinear with $\mathbf{z}_C$.

$$\hat{\mathbf{y}}_C = -\hat{\mathbf{z}}_L \tag{C.2}$$

3. Complete the orthonormal axis. For a right-handed frame, using the cross product provides the this axis of the reference frames. Nonetheless in the software simulation a last condition is applied for the previously derived axis to ensure the system is orthonormal.

$$\hat{\mathbf{x}}_C = \frac{\hat{\mathbf{y}}_C \times \hat{\mathbf{z}}_C}{\|\hat{\mathbf{y}}_C \times \hat{\mathbf{z}}_C\|} \tag{C.3}$$

$$\hat{\mathbf{y}}_C = \hat{\mathbf{z}}_C \times \hat{\mathbf{x}}_C \tag{C.4}$$

4.Apply AMC sign convention. As explained in Chapter 4 during the experiments, it has been observed that the robotic arm is mounted with inverted reference frames. A positive direction of the trajectory waypoints in the lab frame would have a negative orientation in the arm mount reference frame. Therefore, the two transverse axes have to be flipped:

$$\hat{\mathbf{x}}_C \leftarrow -\hat{\mathbf{x}}_C \qquad \hat{\mathbf{y}}_C \leftarrow -\hat{\mathbf{y}}_C \tag{C.5}$$

5. *Crate the rotation and homogeneous transform.* The rotation of the new C frame with respect to L is created and then converted to AMC with previously defined matrices. For each waypoint **p**, the corresponding orientation matrix of the end-effector **R** and the homogeneous transamination **T**(**p**) are computed as follows:

$$\mathbf{R}_{C/L} = \begin{bmatrix} \hat{\mathbf{x}}_C^T \\ \hat{\mathbf{y}}_C^T \\ \hat{\mathbf{z}}_C^T \end{bmatrix} \tag{C.6}$$

$$\mathbf{R} = \mathbf{R}_{C/L}\,\mathbf{R}_{L/AMC}\,\mathbf{C}_Z(\pi) \tag{C.7}$$

$$\mathbf{T}(\mathbf{p}) = \begin{pmatrix} \mathbf{R} & \mathbf{p} \\ \mathbf{0}^T & 1 \end{pmatrix} \tag{C.8}$$

Equations (5.6) and (5.10) supply the translational waypoints for the two modes. The orientation procedure (C.1)–(C.8) then computes the full pose **T**(**p**) required by the IK solver.

5.4.3. Performance Evaluation Robotic Arm

Table 5.8 and Figures 5.17–5.24 summarise the discrepancy between commanded and measured end-effector positions for all Case 2 tests. What is apparent from observing the figures

Table 5.8: Error Metrics for Robotic Arm Motion Test Cases

Test ID	Mean [mm]			Std. Deviation [mm]			RMSE [mm]		
	x	y	z	x	y	z	x	y	z
C2_A1_line_x	-66.64	-2.92	-63.13	20.45	1.28	24.95	69.71	3.19	67.88
C2_A2_line_y	-41.63	-4.14	-52.66	13.90	1.27	18.97	43.89	4.33	55.97
C2_A3_line_xy	-63.28	-3.11	-46.22	22.44	1.34	23.95	67.14	3.38	52.06
C2_B3_half_circle_xy	-58.66	0.76	-57.10	25.27	11.89	52.48	63.87	11.91	77.55
C2_B4_half_circle_yz	-57.39	2.65	-55.54	39.17	61.63	58.23	69.48	61.69	80.47
C2_B5_half_circle_xyz	-57.97	7.18	-56.60	15.64	4.95	52.97	60.05	8.72	77.52
C2_C1_track_target_line	-38.65	-2.71	-68.10	10.77	0.90	15.69	40.12	2.86	69.89
C2_C2_track_target_circle	-36.91	12.96	-52.36	38.93	13.85	66.92	53.65	18.96	84.97

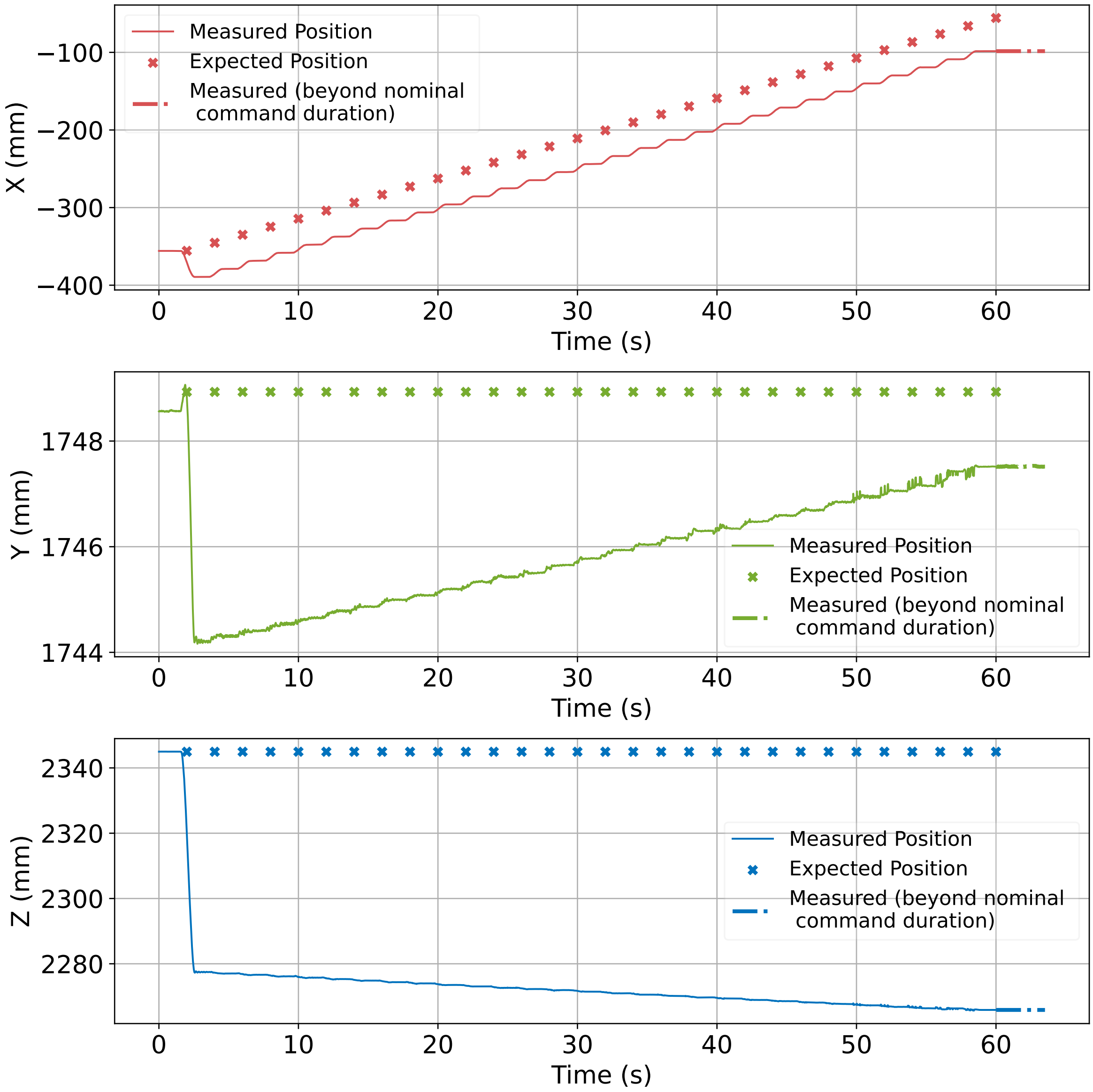


Figure 5.17: Expected versus measured movement of the chaser markers (test C2_A1_line_x)

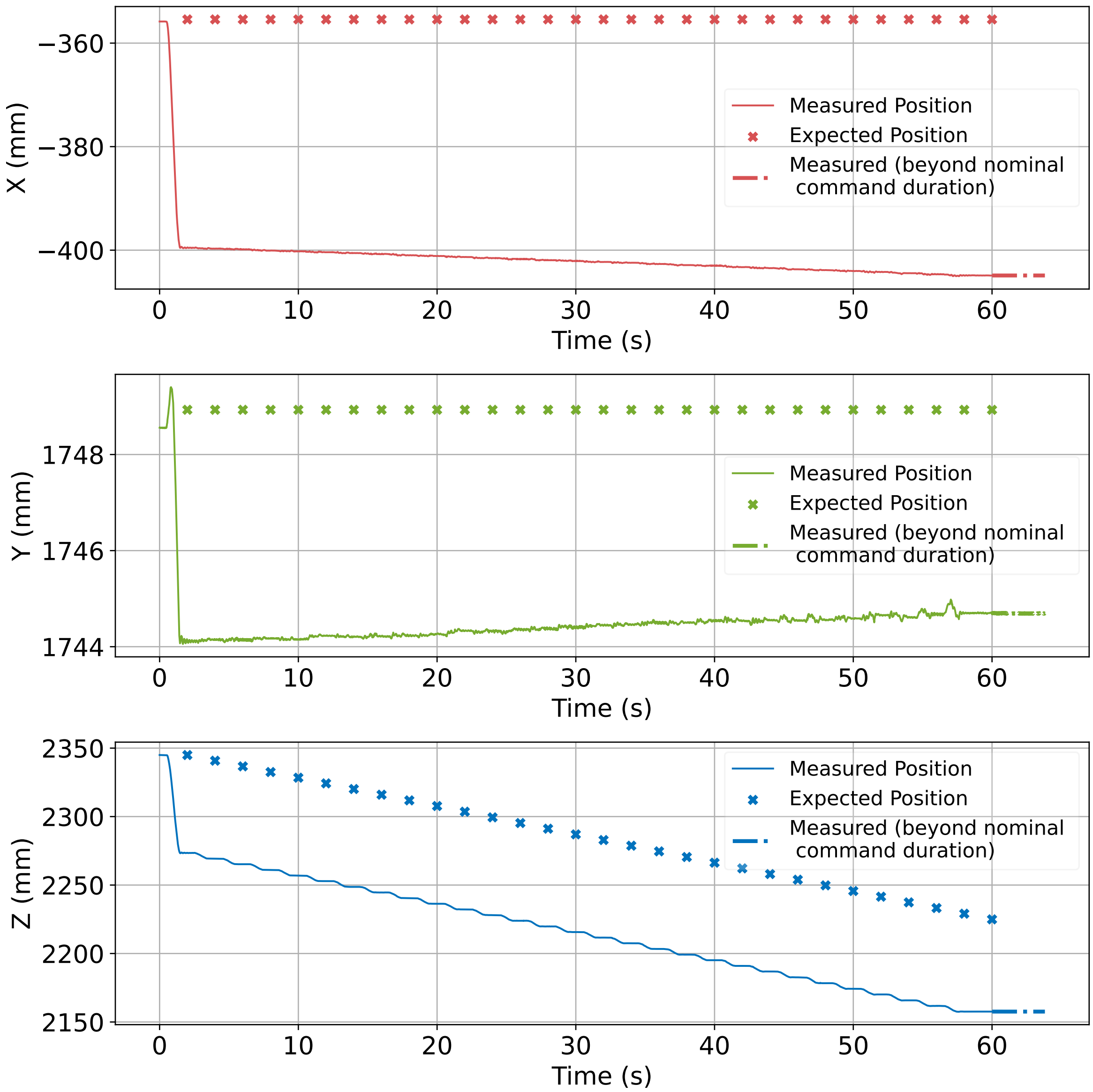


Figure 5.18: Expected versus measured movement of the chaser markers (test C2_A2_line_y)

is that each test starts with a jump before continuing the waypoint trajectory with the expected trend. The jump may point to a misalignment in the frame transformations. It is important to remember that when creating the control input commands, it is assumed that the OBC frame will have the exact alignment as the OptiTrack frame. The position of the arm mount of the chaser rover AMC is then retrieved using the measurements from the OptiTrack to get the position of OBC and then using the physical dimensions of the rover base to place the middle of the rover BC and the arm mount point on the surface of the rover using the chain described in Table 4.1. Given that there was no available alternative to determine the exact position of the hardware in the laboratory, the choice of using OptiTrack readings has induced a frame chain transformation sensitivity, which may be the main cause of the errors reported in Table 5.8.

To determine whether the observed initial jump in the end-effector position is caused by a misalignment in the frame transformations between the OptiTrack and OBC frames, a static pose test is proposed. The robotic arm should be placed in a known, repeatable joint config-

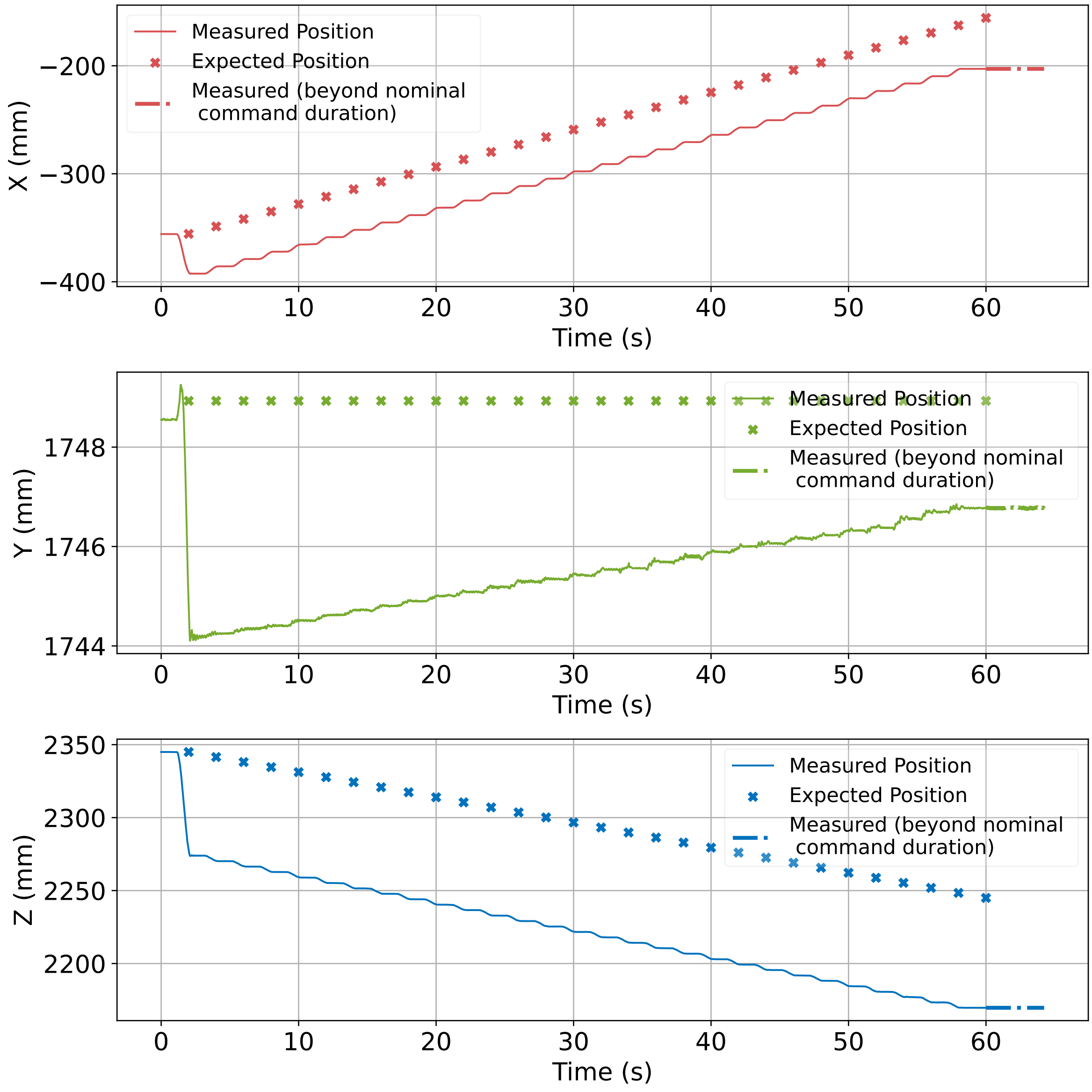


Figure 5.19: Expected versus measured movement of the chaser markers (test C2_A3_line_xy)

uration, such as the same initial configuration used during command generation. The corresponding end-effector position is then recorded in the laboratory frame using the OptiTrack system. Using the same joint angles, the forward kinematics of the robotic arm can be computed, applying the complete transformation chain from the arm mount frame (AMC) to the laboratory frame, as defined in Table 4.1. By comparing the new end-effector pose with the OptiTrack measurement, it is possible to identify translational or rotational offsets.

This method allows the quantification of displacement and orientation errors in the arm mount, allowing for correction within the trajectory generation framework. However, this approach is only valid when the base remains stationary, as any repositioning would require the procedure to be repeated. Consequently, a more robust solution would involve investigating calibration methods for determining the rover base position using additional sensors. The automated alignment of the rover base with respect to the laboratory environment provides a more reliable initialisation of the arm mount location for use in subsequent command generation.

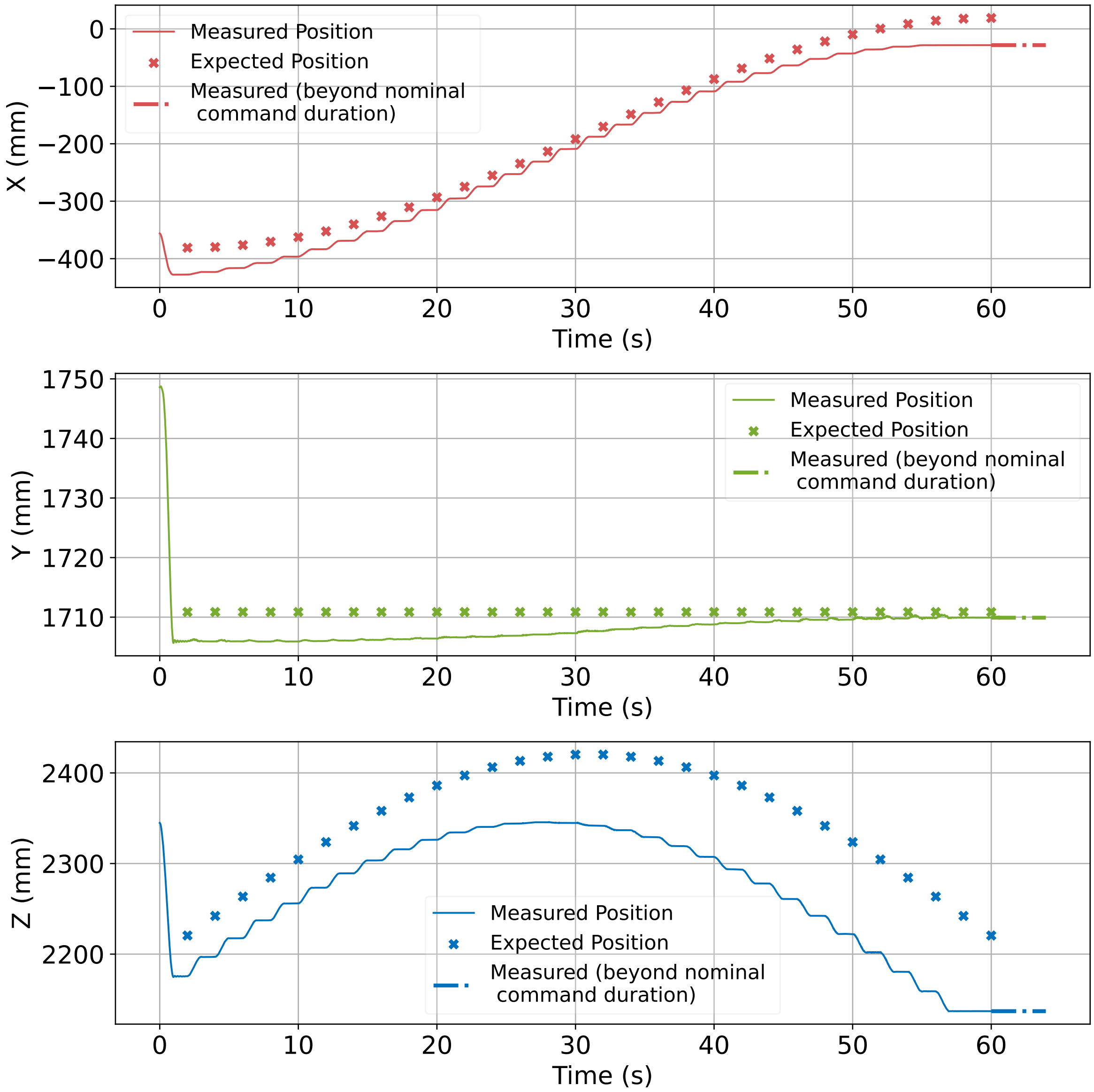


Figure 5.20: Expected versus measured movement of the chaser markers (test C2_B3_half_circle_xy)

Frame transformation chain sensitivity The measured pose of the mount of the robotic arm on the chaser robot (AMC) is recovered through the sequence OBC $\rightarrow$ BC $\rightarrow$ AMC. If the base is placed with a yaw error $\Delta\psi$, the AMC frame inherits *both* a translational offset $(\Delta\text{x},\ \Delta\text{z}) \approx \psi\,\mathbf{p}_{\text{markers}}$ and an angular offset $\Delta\psi$. Frame AMC is then further used to express the coordinates of the chaser frame (C), as explained in the previous section. Consequently, the first motion command, computed in the erroneously aligned frame, produces an immediate jump visible across all conducted tests. The trajectory continuation follows as expected, and the bias remains constant, as shown in Table 5.8. This behaviour reinforces the reasoning that the errors are caused by the initial misalignment at the base-frame alignment stage rather than from the arm kinematics itself.

Performance Group A: Linear End-Effector Motion

In tests C2_A1–A3 (straight-line translations), the mean position errors along x and z range from –66.6 mm to –41.6 mm, with RMSE values up to 69.7 mm. The results are shown

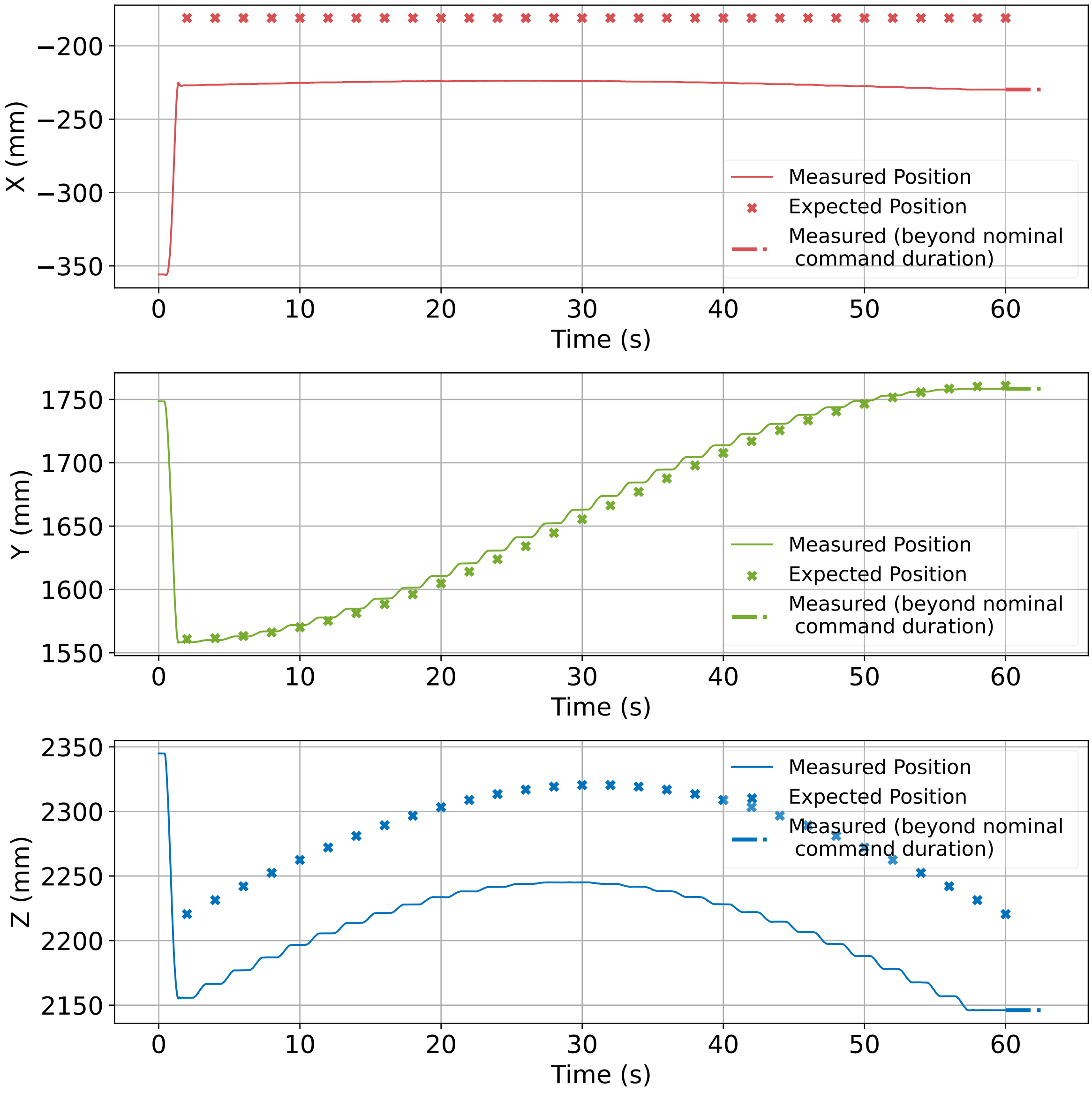


Figure 5.21: Expected versus measured movement of the chaser markers (test C2_B4_half_circle_yz)

in Figures 5.17, 5.18, 5.19). Standard deviations remain low ($\sigma_x \leq 22.4$ mm, $\sigma_z \leq 24.0$ mm), indicating that aside from the constant offset, the arm consistently tracks the interpolated points defined in Equation (5.6). The vertical axis error stays below 5 mm, confirming that the homogeneous transforms for L $\rightarrow$ AMC preserve y alignment when orientation is held fixed via the retrieved orientation for the joint configuration. The standard deviation is also smaller than 5 mm across all three tests. The robotic arm observation is impacted less by the sensitivity to the robot's position in the laboratory observed in Case 1, as the base is stationary and the distances covered are shorter.

Performance Group B: Circular End-Effector Motion

Half circle tests C2_B3–B5 introduce curvature and the manipulator continues to replicate each circular arc with minimal distortion of shape (Figures 5.20, 5.21, 5.22). Mean errors remain comparable to Group A (–57 mm in x, –56 mm in z), but standard deviations increase for out-of-plane motion (σ_y = 61.6 mm in test C2_B4). A similar increase is observed also in

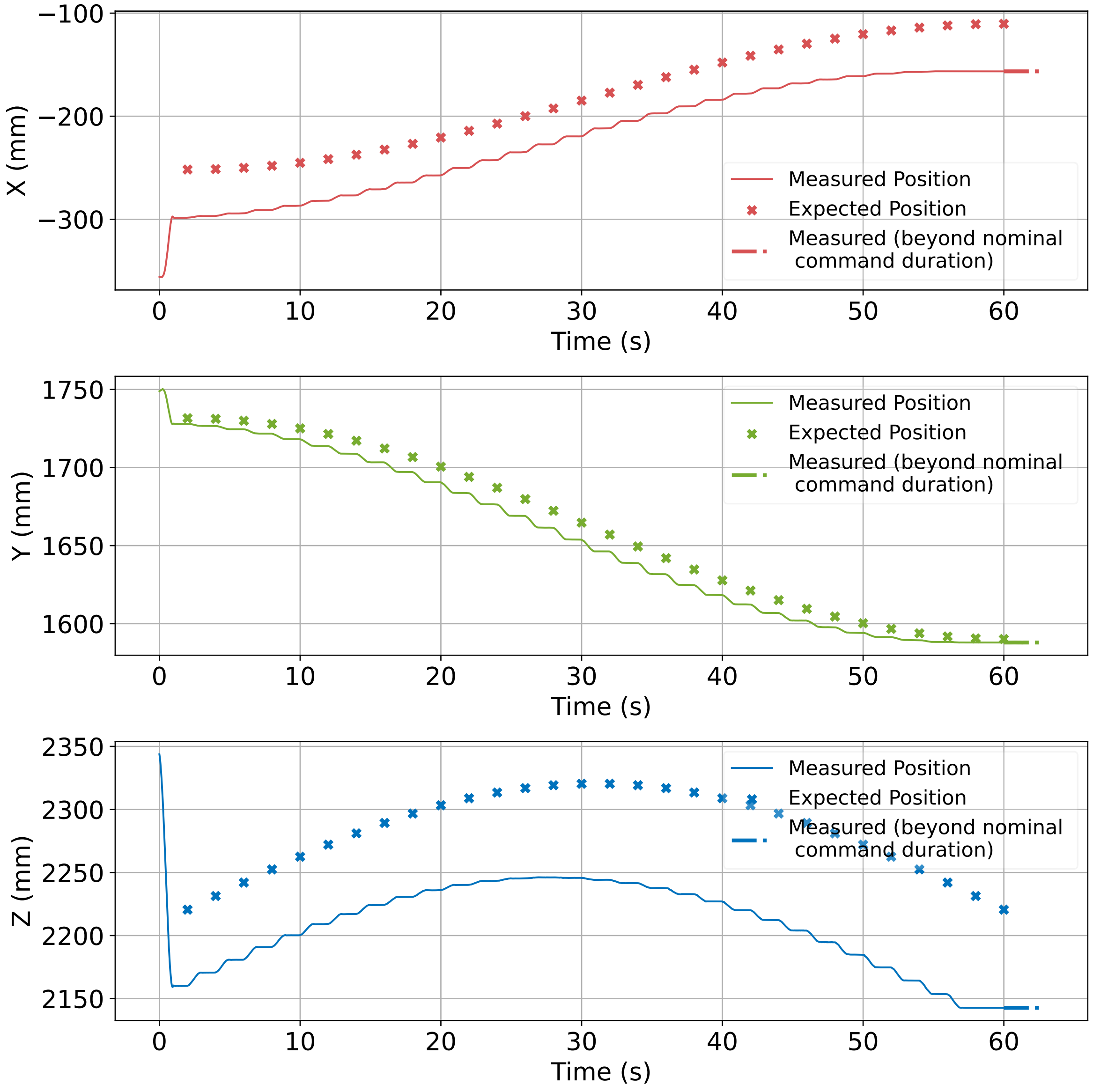


Figure 5.22: Expected versus measured movement of the chaser markers (test C2_B5_half_circle_xyz)

C2_C2 when the end-effector is performing the same circular motion while tracking the target asteroid. The increase can thus be associated with the motion type aggravating the influence of the misalignment of frame AMC in which the target locations of the end-effector are defined.

Performance Group C: Target Tracking

When imposing the dynamic orientation constraint using Equations (C.1)–(C.8) to point the end-effector at **t**, tracking accuracy improves slightly in the translation axis, with mean x bias reduced to –38.7 mm and RMSE to 40.1 mm in test C2_C1_track_target_line (Figure 5.23). The circular target-tracking test C2_C2_track_target_circle achieves similar performance with mean biases –36.9 mm to +12.96 mm and RMSE up to 84.97 mm (Figure 5.24). Unlike test C2_C2, which includes a vertical displacement along the y-axis of the OptiTrack in its movement and therefore sees an increase in the measured positional error, test C2_C1 maintains a constant height. Therefore, the mean errors and standard deviation are close to 0.

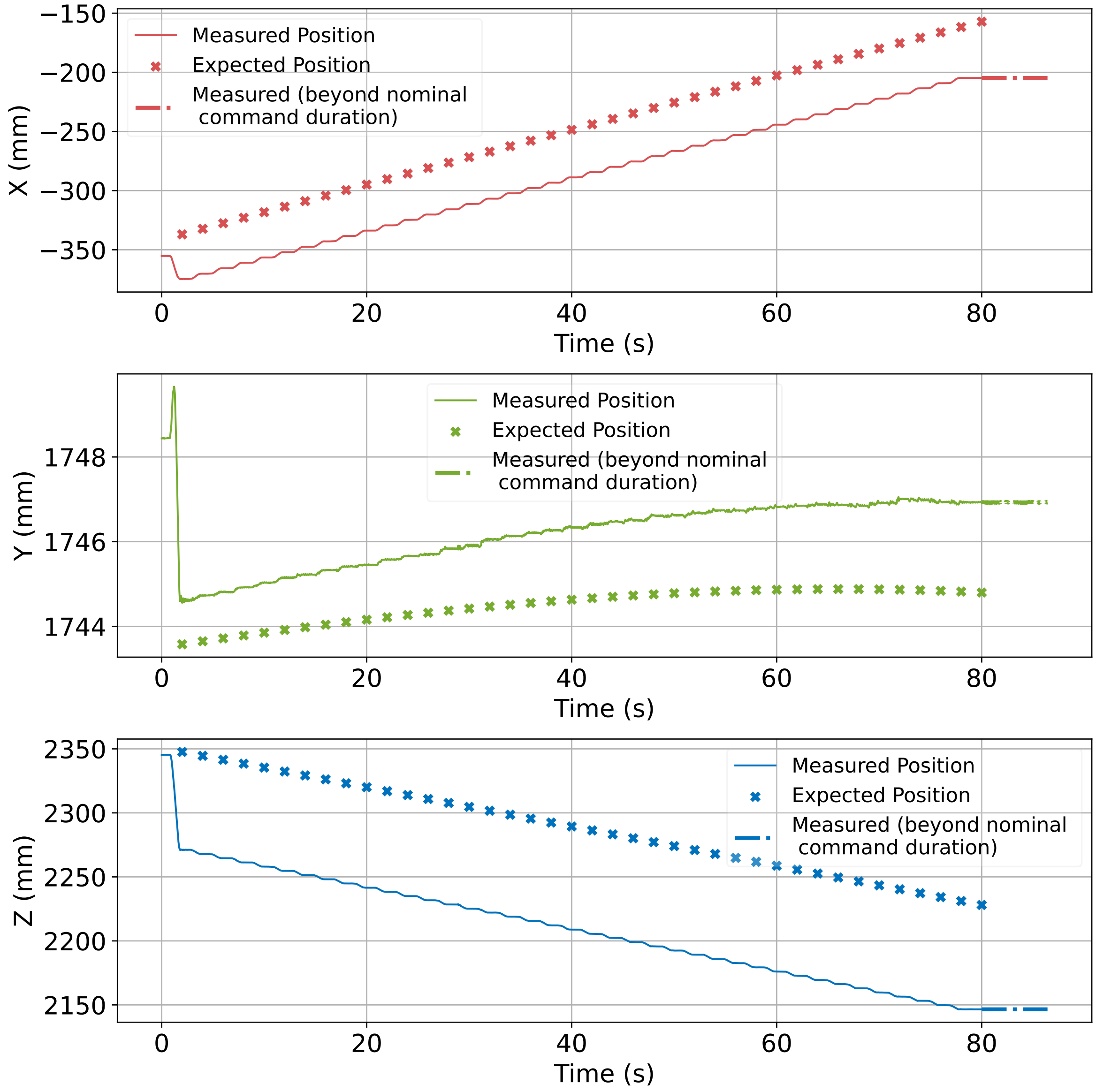


Figure 5.23: Expected versus measured movement of the chaser markers (test C2_C1_track_target_line)

Implications of errors of the robotic arm for subsequent cases

Overall, the robotic arm demonstrates consistent performance in both linear and curved trajectory tracking once initial frame misalignment is considered and corrected. The systematic bias arising from the alignment of the transformation chain OBC → BC → AMC underscores the importance of precise initial calibration. For subsequent applications, it is crucial to incorporate an automated calibration procedure capable of accurately determining and compensating for the offsets introduced by small angular and positional discrepancies. By achieving this, the sub-millimetre repeatability characteristics of the UR16e robotic arm can be fully used.

The employed LM solver exhibits strong robustness and sub-millimetre repeatability once systematic bias from frame misalignment is mitigated. Given its reliability and precision in converging to accurate solutions, the continued use of the LM solver is justified without the need for alternative optimisation methods.

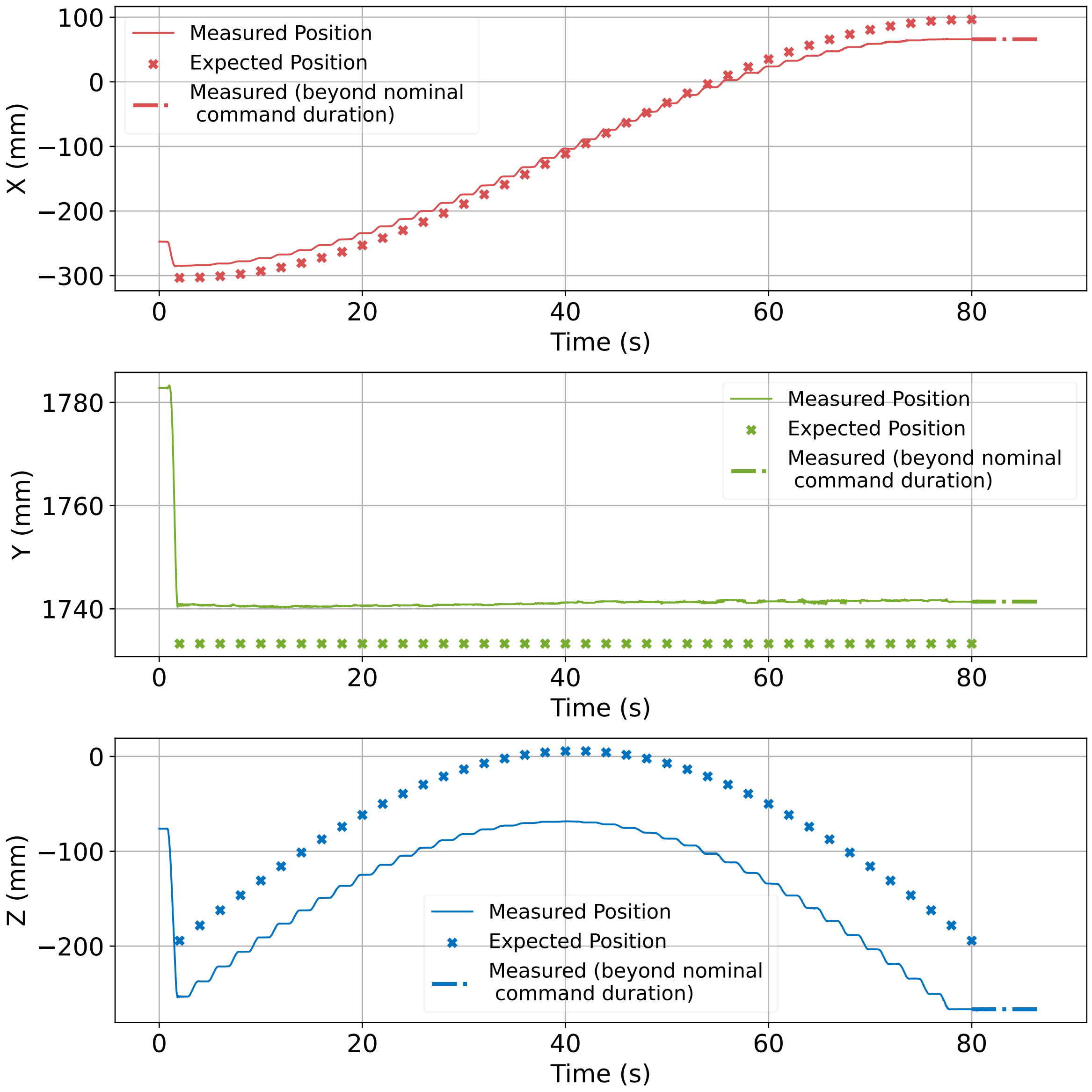


Figure 5.24: Expected versus measured movement of the chaser markers (test C2_C2_track_target_circle) - test was redone with the correct command input at a different time

For future practical applications and experimental scenarios, focusing on limited segments of an orbit or specific mission phases is recommended, given the limitation of accurately positioning the arm mount of the robotic arm within the laboratory's space. For instance, conducting surface scans on asteroid topographies and larger surface models of the asteroid could be an alternative to the available robotic system and its current capabilities. This approach uses the robotic arm's repeatability and the robustness of the solver while avoiding the issues associated with relying on OptiTrack for hardware localisation within the laboratory. For full orbital trajectory generation, accurately localising the hardware within the laboratory's space is crucial for generating the command input for both the robotic base and the robotic arm. Therefore, in this scenario, the robotic arm alone cannot compensate for the errors of the robotic base discussed in Section 5.3. The tendency of the robotic base to undershoot the target locations commanded would only further increase the misalignment of the AMC frame, in which the control inputs of the robotic arm are expressed.

5.5. Case 3: Coordinated Base and Arm Movement

In previous sections, linear and circular motions were individually analysed for the rover base and robotic arm. However, coordinated movement between these robotic components is essential to replicate orbital trajectories accurately within the laboratory. Due to the notable tracking errors observed separately for the rover base and robotic arm, achieving fully autonomous and coherent motion has proven more complex than initially anticipated using the open-loop control framework proposed in this work. This section details the intended orbital movement scenarios, their mathematical implementation, associated assumptions, and limitations, and outlines methods for generating trajectory commands tailored to the laboratory constraints.

Concept Mission

Chapter 2 provided an overview of the phases of the NEAR Shoemaker mission, highlighting the different orbital stages undertaken to approach, orbit, and observe asteroid 433-Eros. Most notably, this includes circular orbits at altitudes of 200 km, 100 km, 50 km, and 35 km. One of the main objectives for the design of the GNC lab is to create a testing and validation environment for various orbital scenarios. This environment will be a benchmark for evaluating the performance of software modules dedicated to optical navigation around small bodies.

Before testing these modules in complex or perturbed conditions, it is essential to establish a baseline by assessing the accuracy that can be replicated in the laboratory using the robotic platform introduced in Chapter 3. Therefore, the modelled environment of the spacecraft has to be adjusted accordingly.

Environment

When defining a simulation, it is important that the environment modelling represents the physical problem, thus including the properties of the bodies involved so that the acceleration or perturbations present can be correctly estimated. Selecting the environment model is a trade-off between attempting to mathematically describe the problem as accurately as possible and ensuring that computational demand can be met by the hardware available both in a simulation and an experimental hardware-in-the-loop environment. Including a model that is as detailed as possible can lead to a highly non-linear system being evaluated. This approach has the benefit of accounting for all known perturbations and forces present in the space environment and their impact on the state estimation of the spacecraft. Which parameters can be approximated while maintaining the realism of the problem described will be discussed in this section. If those conditions cannot be replicated in the lab environment, some environmental characteristics are excluded from the simulation and can thus not be tested.

When considering how to model the space environment, different parameters can be important, such as orbital mechanics of the Sun, Earth, Moon, asteroid or other bodies involved in the problem to properties such as gravity and magnetic fields, atmosphere, solar-radiation pressure or third-body perturbations (Mooij, 2024; Wakker, 2015). Previous research where orbits around 433-Eros have been modelled (Bourgeaux, 2020) (Spee, 2022) (Munuera Vilalta, 2024) have shown that the most impactful perturbations on the orbit are the gravitational field of 433-Eros are third-bodies perturbations due to the Sun in terms of gravitational accelerations and Solar Radiation Pressure (SRP) in terms of non-gravitational ones.

Therefore, as a first step in this study, it is selected to start by first modelling the gravitational field of 433-Eros. While more advanced methods to determine the gravity model of the asteroid, such as Spherical Harmonics Modelling (Sinem Ince et al., 2019) or Gravity Field Polyhedron Modelling (Mooij et al., 2024) in the lack of a benchmark for what is the achievable accuracy of creating a trajectory in the hardware environment it is chosen first to model the gravity filed as Point Mass. The first scenario investigated in this thesis is a circular, constant-velocity Keplerian orbit around 433-Eros. This scenario provides an analytical benchmark to assess and quantify the experimental performance of the hardware. Additional perturbations can be modelled once a benchmark is established for this type of orbit creation in the laboratory. Therefore, for this application, the following assumptions will be made for modelling the environment of a spacecraft orbiting 433-Eros.

Assumptions and Limitations of Environmental Modelling

- Gravity is modelled as a point mass field (central potential).
- No perturbations (non-spherical gravity, third-body forces, etc.) are included in this initial phase.
- The asteroid is considered fixed in space (non-rotating) for the baseline experiment.
- The orientation of the spacecraft is not propagated. The orientation of the camera point from the end effector will be modelled on the robotic setup such that it always points towards a target point representing the asteroid retrieved from OptiTrack readings.

Kepler Orbit

The orbit selected for this case study is a circular orbit at a 100 km altitude above the surface of 433-Eros. This particular orbit has been employed in past studies to evaluate the accuracy of the asteroid's spherical harmonics model and improve the robustness of navigation filters (Munuera Vilalta, 2024). Additionally, it has served as a test case for experimental imaging studies conducted within the department (Terwindt, 2025). The chosen orbit aligns conveniently with the available laboratory space. Given the laboratory scaling factor of 10:1,000,000, a 100 km orbital radius translates precisely to a 1 m laboratory radius, providing sufficient operational manoeuvre space for the rover base around the target asteroid model, as outlined in Section 3.4.

Before incorporating higher-order gravitational perturbations, the focus is placed on unperturbed two-body motion, modelled as a point mass central force problem. This simplification helps isolate the robotic system inaccuracies from the limitations of the physical model. It also provides a quantitative benchmark for system performance that is crucial for identifying sources of trajectory deviation in later, more realistic tests.

The gravitational interaction between 433-Eros and a spacecraft can be described by Newton's law of gravitation, as shown in Equation (5.11):

$$F = G\frac{m_1 m_2}{r^2} \tag{5.11}$$

where m_1 and m_2 are the masses of the spacecraft and 433-Eros respectively (in kg), r is the distance between their centres of mass (in km), and $G = 6.67 \times 10^{-11}\,\mathrm{m^3\,kg^{-1}\,s^{-2}}$ is the gravitational constant. The gravitational parameter of Eros is approximated by $\mu = Gm_2$. Using the estimated mass from Table 5.9, the value used in this simulation is $\mu = 480,240\,\mathrm{m^3\,s^{-2}}$. Additional asteroid properties are given in Table 5.9.

This method assumes a spherical mass distribution which is inaccurate for significant mass anomalies or elongated bodies, as in the case of 433-Eros (see Figure 6.14). Therefore, once a benchmark for expected hardware performance in generating Keplerian trajectories is established, additional environmental accelerations must be included. These accelerations directly affect the orbital shape and manoeuvre time. For more elliptical or perturbed orbits, the velocity is no longer constant, which would have to be reflected in the control command time of the robotic arm.

Keplerian orbits represent ideal two-body motion under Newtonian gravity, where the spacecraft follows a conic trajectory around a central point mass. No external forces such as third-body perturbations, solar radiation pressure, or non-spherical gravity fields are considered.

For this case study, a circular orbit is chosen at an orbital radius $r_0 = 100$ km. The orbital velocity is computed using:

Table 5.9: 433-Eros Specifications[6]

Property	Value	Unit
Size	33 x 13 x 13	km
Approximate mass	7.2×10^{15}	kg
Rotation Period	5.270	hrs
Orbital Period	1.76	yrs
Spectral Class	S	-
Semimajor Axis	1.458	AU
Perihelion Distance	1.13	AU
Aphelion Distance	1.78	AU
Orbital Eccentricity	0.223	-
Orbital Inclination	10.8	deg
Geometric Albedo	0.16	-
Rotational rate	$\omega_{cb} = (0, 0, \omega_{cb,z}) = (0, 0, 3.3244 \times 10^{-4})^T$	rad/s

$$v_0 = \sqrt{\frac{\mu}{r_0}} = 2.19\,\mathrm{m/s} \tag{5.12}$$

The orbit is simulated by solving the equations of motion:

$$\begin{aligned} \dot{x} &= v_x \\ \dot{y} &= v_y \\ \dot{z} &= v_z \\ \dot{v}_x &= -\mu \frac{x}{(x^2+y^2+z^2)^{3/2}} \\ \dot{v}_y &= -\mu \frac{y}{(x^2+y^2+z^2)^{3/2}} \\ \dot{v}_z &= -\mu \frac{z}{(x^2+y^2+z^2)^{3/2}} \end{aligned} \tag{5.13}$$

These equations are integrated numerically using the Dormand-Prince method, an explicit Runge-Kutta scheme of order 5(4) with adaptive step size (RK45)[4]. The initial conditions for position and velocity are chosen such that the spacecraft begins along the x-axis and travels tangentially along the y-direction. The solution of the initial value problem of the set of ordinary differential equations shown in Equation (5.13) is visualised in Figure 5.25. Here, the point mass location of 433-Eros is assumed to be centred at the origin of the plot.

To implement this in a constrained lab environment, the orbital solution is normalised using a scaling factor according to Equation (5.14). The scaled orbit is visualised in Equation (5.26). From the scaled trajectory, a series of keypoint locations are sampled to be tracked by the robotic system. The first test considered has zero inclination, and it is for this reason that only the x and y coordinates have been visualised. However, should an inclination be added, the same reasoning can be applied to the z coordinates.

$$x_{lab} = x_{orbit} \cdot S, \quad y_{lab} = y_{orbit} \cdot S \tag{5.14}$$

where S is the minimum of the lab width and length scaling factors. The scale applied correlates to the scale of the asteroid model of 10:1,000,000, where 1 m in the laboratory space

[4] RK45 in `SciPy`:https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html (accessed: 6 June 2025).

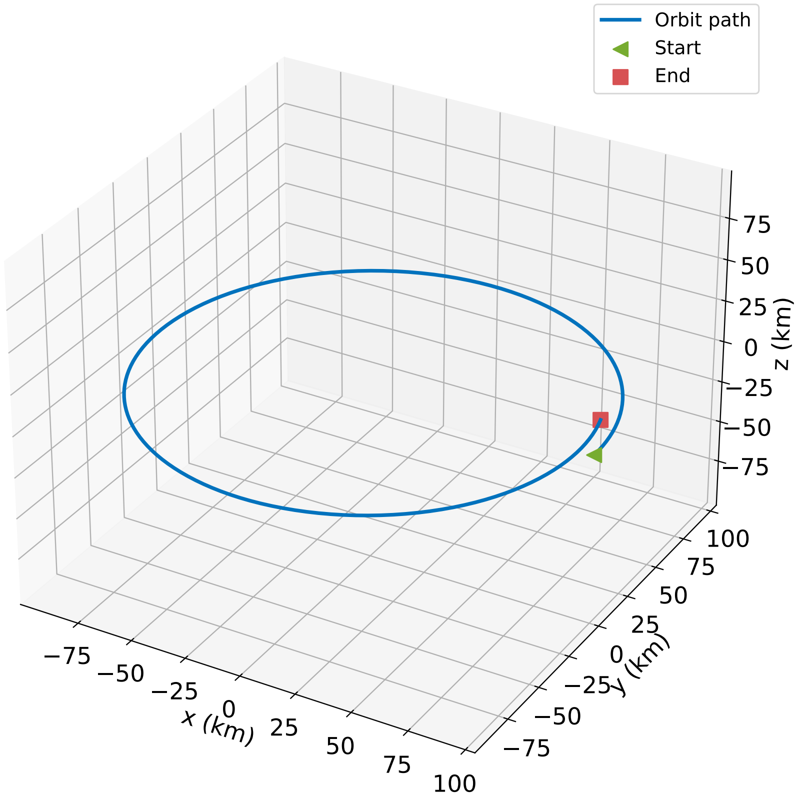


Figure 5.25: 100 km Kepler orbit around 433-Eros modelled as point mass

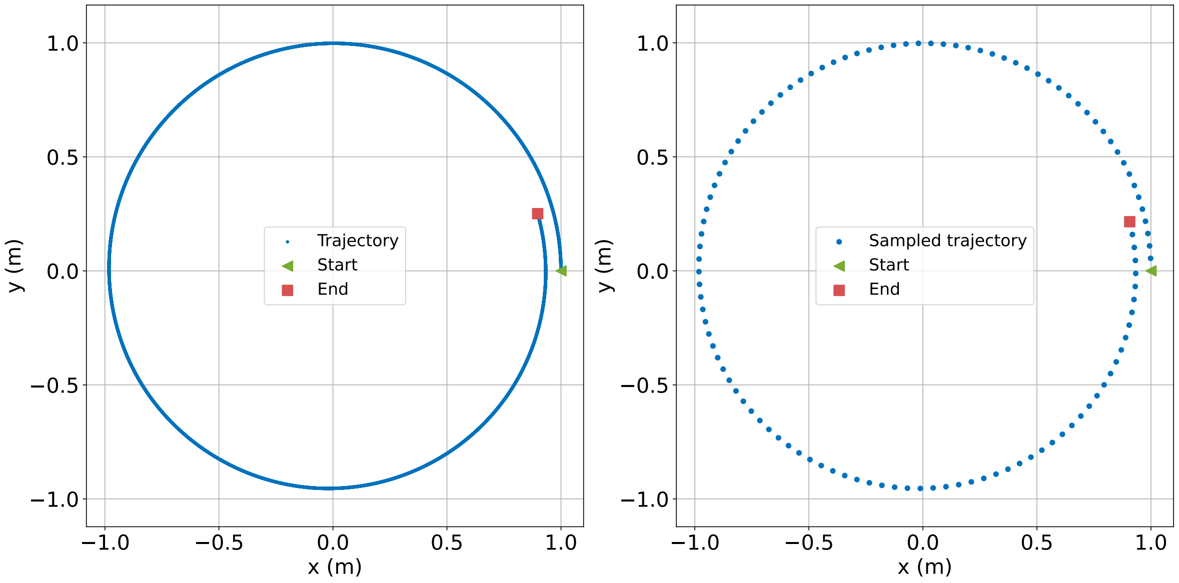


Figure 5.26: (left) projection of the Kepler orbit scaled to the laboratory dimensions (right) sampled points 1 every 100 along the scaled trajectory

corresponds to 100 km in the space environment. Therefore, 1 cm of the observed asteroid model in the laboratory experiment corresponds to 1 km of the real asteroid.

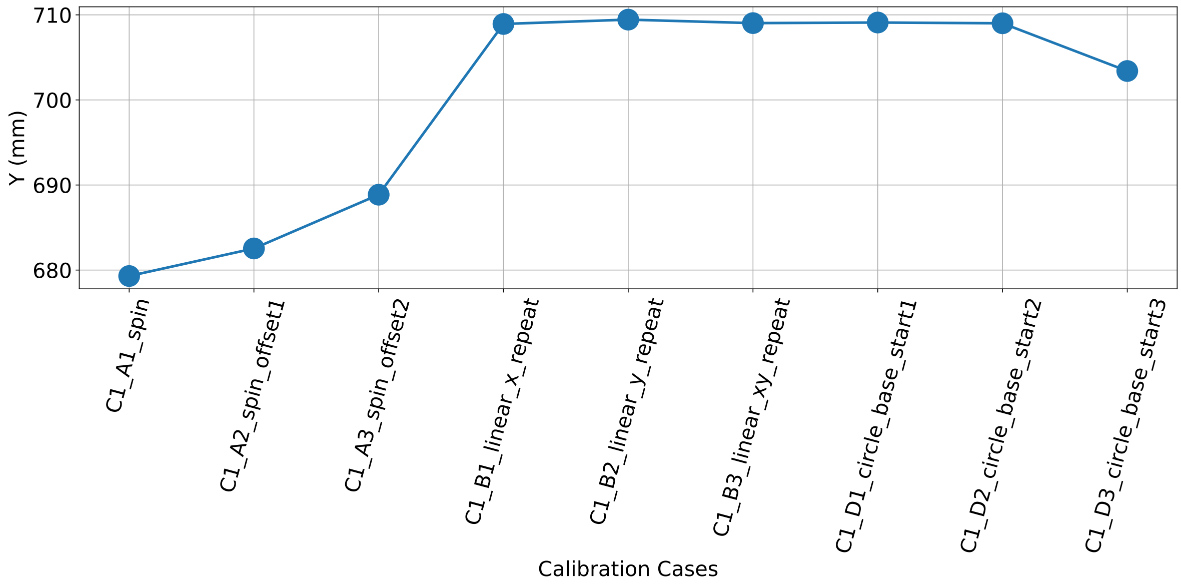


Figure 5.27: Variation in the Y-coordinate (height indication) of the markers placed on the rover base, as measured by the OptiTrack system, across experiments

5.6. Uncertainty Sources

Initial Calibration: This study revealed that the initial calibration of the hardware within the laboratory's space was one of the leading causes of errors. Positioning the rover base to orient with the walls of the GNC lab perfectly is not yet automatic, and manual alignment introduces errors. For this reason, some labs have wall-mounted platforms that guide movement and allow the robot to travel along. As discussed in Section 5.3, this approach of manually positioning the robotic arm can induce errors of up to 30 cm in the initial estimation of the position of the rover base mainly due to orientation shifts. The error is one limitation of the lab that can be fixed by constraining the robotic base within a space during the initial calibration space.

Marker placement: It is possible that some slight movement may be recorded in the marker position during the operation of the rover base. However, this movement should not be significant. It is thus hypothesis that the small vertical oscillations recorded in the height of the position of the markers of the rover base in C1_A1_spin, when the rover base is not translating in the volume of the laboratory, may be caused by these small disturbances. In this case, the expected error would be smaller than 5 mm.

Variations in the marker identification at different locations in the laboratory: During the experiments, it was observed that for the same laboratory calibration, the markers placed on the rover base were identified at different heights across different tests. To evaluate how large the variation in the recorded height is, the variation in the y-coordinate of the markers placed on the rover base, as measured by the OptiTrack system, across all the experiments in which the rover base is moving is shown in Figure 5.27. The difference in the recorded height is 30 mm.

It is then interesting to assess whether there is a correlation between the recorded height and the position of the rover base in the volume of the laboratory. The variation of y measurements with respect to x measurements corresponding to the position along the width of the laboratory is shown in Figure 5.28. Similarly, the height measurements are plotted against the position along the laboratory length shown in Figure 5.29. While the number of tests is

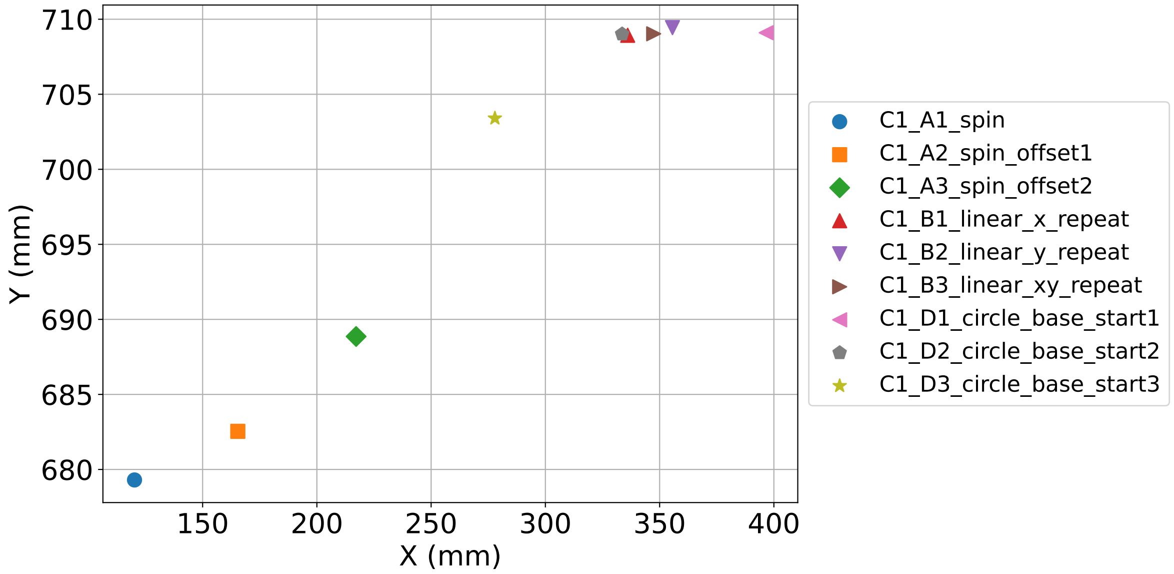


Figure 5.28: Variation in the Y-coordinate (height indication) of the markers placed on the rover base, as measured by the OptiTrack system, across experiments with varying base positions along the X-axis (lab width). The X-axis represents the corresponding OptiTrack X-coordinate readings.

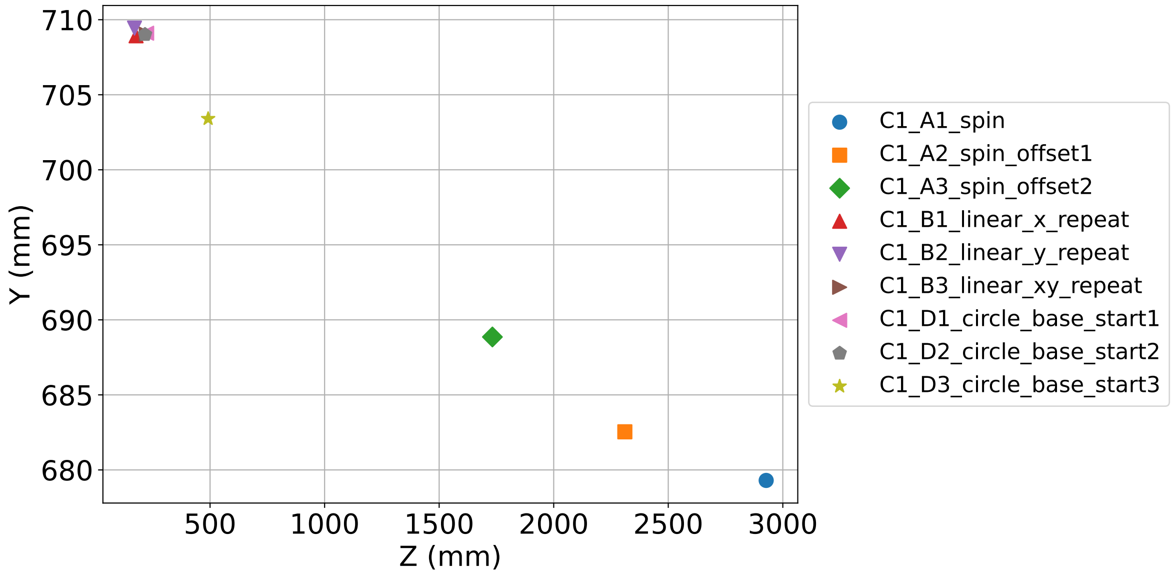


Figure 5.29: Variation in the Y-coordinate (height indication) of the markers placed on the rover base, as measured by the OptiTrack system, across experiments with varying base positions along the Z-axis (lab length). The Z-axis represents the corresponding OptiTrack z-coordinate readings.

too small to draw rigorous conclusions, it can be hypothesised that being closer to the origin of the OptiTrack system along the length of the laboratory would yield the measured height to be higher, whereas being further away from the height would be measured to be smaller. An opposite trend is observed for the variation of the rover base along the width of the laboratory. Here, being closer to the origin, as for C1_A1_spin, leads to lower height values. Being closer to the edges of the rooms, as is the case for C1_D1_circle_base_start2 leads to slightly higher measurements.

However, the room width is only 3900 mm. For a rover base with a length of 760 mm, the variation of the laboratory's width would be proportionally much smaller compared to a translation along the laboratory's length. For this reason, it is hypothesised that the position along the laboratory's length is the leading factor. This hypothesis would have to be verified through additional tests. In addition, it can also be checked whether the quality of the room calibration impacts these results.

Incomplete rover base movement: As discussed in Section 5.3, the rover base fails to reach the target location over larger distances. The number of initiated displacement commands matches the expected number of commands sent to the robotic system for each test. As mentioned, this may indicate that the rover base is skidding on the laboratory surface. It would have to be investigated further through additional tests to see if that is true and if the friction with the coating applied on the darkroom floor is preventing the wheels from performing the full commanded movement, or if there is another underlying issue.

Conclusions on Robotic Movement Modelling

This chapter addressed the experiment design subquestion by modelling the robotic system's movement and evaluating which motion parameters must be accurately replicated to support relative asteroid navigation. In doing so, the influence of the OptiTrack motion capture system on reference frame alignment and command generation has also been examined. Sufficient data have been collected to inform the subsequent analysis on experiment evaluation, which is discussed in Chapter 7.

The test campaign evaluated the capability of the rover base to execute rotation (Group A), translation (Group B), and circular motion (Group D) commands using a waypoint-based control framework. Results indicate that while the vertical tracking error remained below 10 mm, confirming correct modelling of the marker height and OptiTrack frame alignment in the y-direction, significant horizontal errors were observed. Translational tests exhibited systematic undershoot of up to 250 mm, and rotational profiles accumulated angular drift, with RMSE values exceeding 300 mm in circular trajectories. These findings suggest that the horizontal plane performance is limited by cumulative wheel slip, encoder integration errors, and initial misalignments due to manual command placement of the rover base.

All commanded trajectories were defined with respect to the BC frame, with motion captured through OptiTrack measurements in the OBC frame. As the marker placement and calibration biases introduce a static offset, the reconstruction of expected marker motion relies on frame transformations that proved sensitive to yaw alignment and localisation in the laboratory volume, which has been analysed as an uncertainty of the OptiTrack motion capture system and for which additional tests are proposed.

These limitations highlight the need for closed-loop correction within the robotic testbed. The current open-loop control architecture results in a non-negligible propagation of error, particularly for circular or arc-like motions. For this reason, Case 2 and Case 3 were designed to examine whether arm or coordinated arm base motion, respectively, can compensate for and mitigate base-level inaccuracies. The previously mentioned alignment issues also affected Case 2 in defining the performance of the robot arm commands, where it has been observed that an initial jump occurs at the start of each trajectory. Therefore, the resulting errors cannot be compensated by combined movement, and the error and localisation have to be integrated into the command generation logic moving forward.

Finally, the choice of circular trajectories is justified both from a mission and experimental perspective. These arcs provide a simplified representation of Keplerian motion in the absence of perturbations and serve as an operational benchmark for evaluating system suitability during initial global characterisation phases. Furthermore, the arc geometry satisfies overlap requirements for surface scanning, further discussed in Chapter 6, aligning the laboratory methodology with navigation tasks performed during proximity operations.

6
Surface Reconstruction

The work so far focused on configuring the robotic testbed for trajectory generation in relative asteroid navigation. This chapter addresses the next stage: surface characterisation. Once accurate keypoints are reached, detailed surface models become essential for reliable navigation and tracking. The feasibility of using the GNC laboratory for 3D surface reconstruction is assessed to determine the achievable model accuracy and its suitability for navigation filters. Photogrammetry is proposed as the reconstruction method in Section 6.1. The image datasets are introduced in Section 6.2, and the implementation of the required open-source software is detailed in Section 6.3. Finally, Section 6.4 discusses the laboratory's reconstruction capabilities and their relevance to developing a testbed for relative asteroid navigation.

6.1. Photogrammetry

Photogrammetry is the process of recreating three-dimensional surface geometry from two-dimensional images by detecting and matching features across overlapping photographs (Aber et al., 2010). Photogrammetry uses the principle of creating structure from motion.

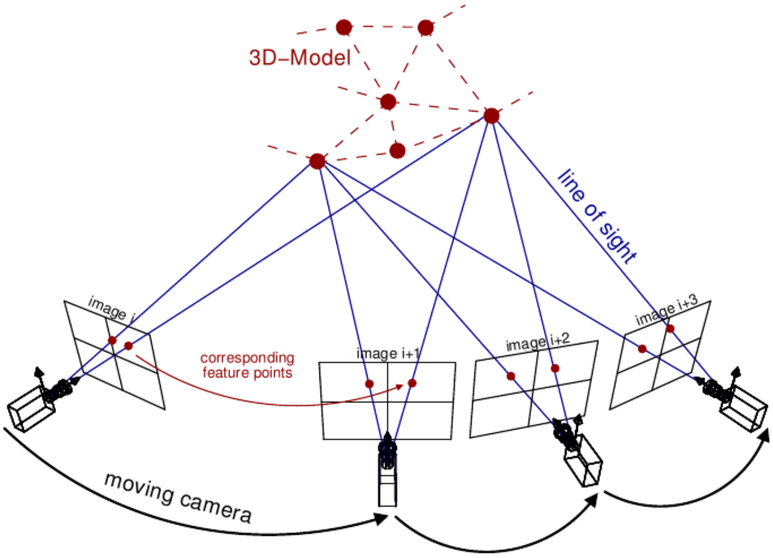
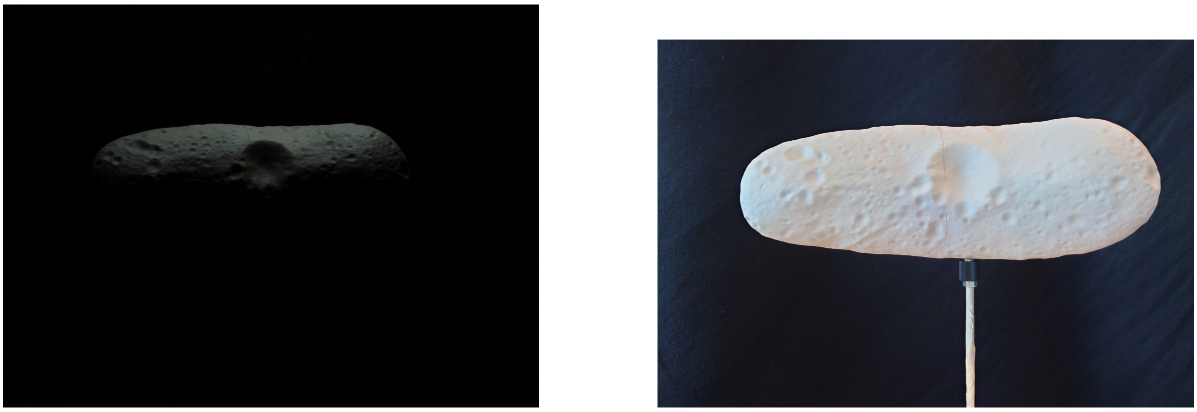


Figure 6.1: Overview of the structure-from-motion pipeline[1]

[1] Theia Vision Library: http://theia-sfm.org/ (accessed: 9 May 2025).



(a) 433-Eros 100 km experimental orbit (Dataset 1)

(b) 433-Eros (Dataset 2)

Figure 6.2: Comparison of the two datasets used for surface reconstruction of the asteroid model

By moving the camera at different angles around the object that will be reconstructed, the algorithm can scan for corresponding features across consecutive image frames. With this approach, it is possible to triangulate the location of the point on the object being scanned and recreate the 3D model. This is visualised in Figure 6.1.

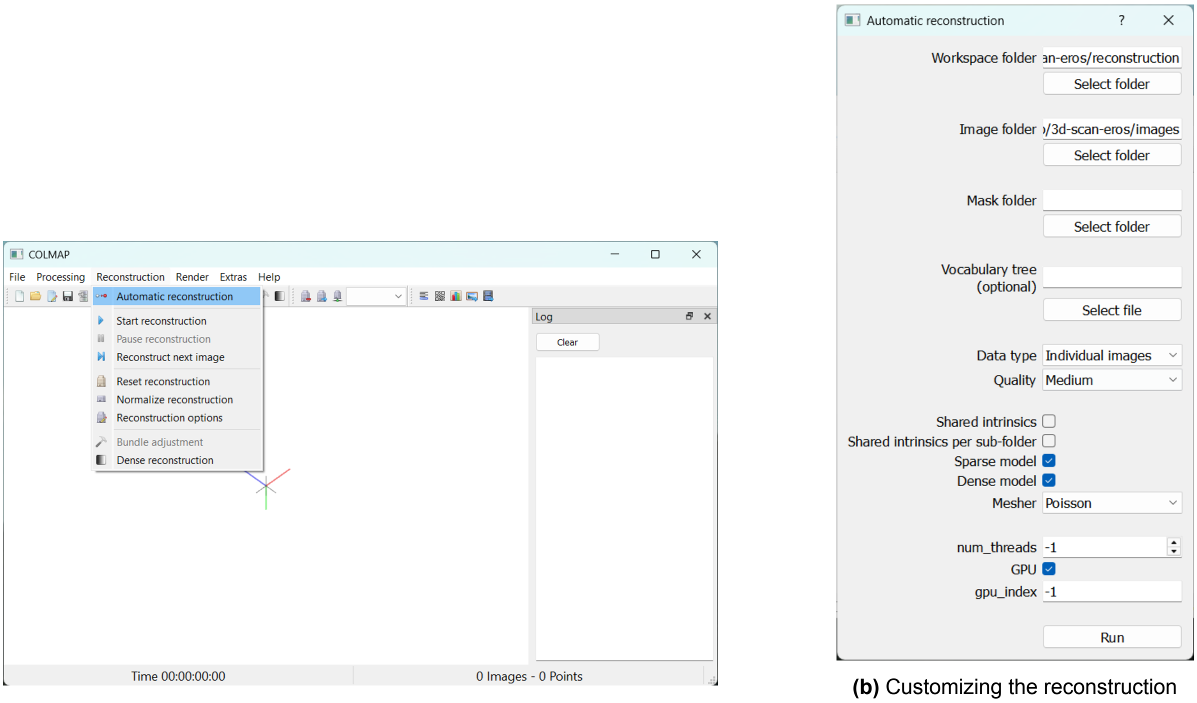
The workflow consists of image acquisition, camera calibration, sparse feature extraction, bundle adjustment, dense stereo reconstruction, and mesh generation. For image acquisition in this work, reconstructing 433-Eros will be the objective but it is important to note that the process described is not bound to the 433-Eros model. The steps explained in this chapter can be taken to create a different surface reconstruction of a different celestial body or object of interest. Feature detection and matching are performed using scale-invariant key points automatically identified by the software both on the object to be reconstructed and on the background. Subsequent bundle adjustment of the matched features refine camera poses and the sparse point cloud geometry. A dense multi-view stereo step then produces a high-resolution point cloud, which serves as the basis for surface reconstruction and the recreation of the 3D surface model.

6.2. Image Datasets

Two sets of Images of the 433-Eros model were evaluated for surface reconstruction. One set was taken in the control room of the GNC laboratory using diffuse natural lighting conditions. In contrast, the other set used experimental images with the model under simulated orbital lighting conditions.

- **Experimental Dark Room Images (Dataset 1):** Images obtained from the experimental navigation setup described by Terwindt (2025).
- **Smartphone Images (Dataset 2):** Photographs of the 433-Eros model, captured with a smartphone, under natural lighting conditions in the control room

Figure 6.2 presents a sample of the input images to the photogrammetry software from the two different datasets showing the same view of 433-Eros. Due to the light source in the experiment conducted in the darkroom being placed above the asteroid, the lighting on half of the asteroid model is minimal, reducing the features that can be detected by the software. This type of hard shadow is detrimental for the photogrammetry reconstruction. One hypothesis was that the system would not be able to reconstruct this area, but this should not be the case for the dataset of images gathered within the laboratory.



Time 00:00:00:00

0 Images - 0 Points

Workspace folder: an-eros/reconstruction

Image folder:)/3d-scan-eros/images

Mask folder:

Vocabulary tree (optional):

Data type: Individual images

Quality: Medium

Shared intrinsics ☐

Shared intrinsics per sub-folder ☐

Sparse model ☑

Dense model ☑

Mesher: Poisson

num_threads: -1

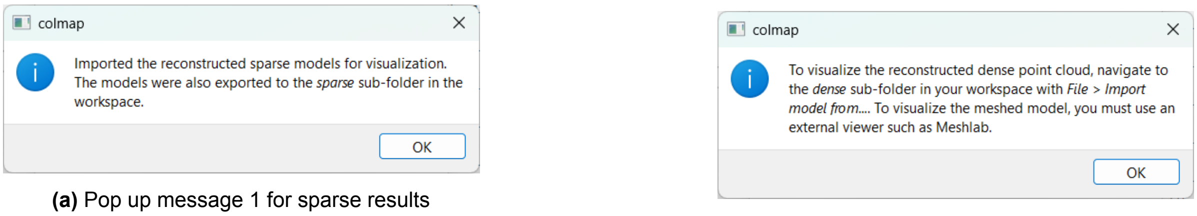
GPU ☑

gpu_index: -1

(a) Starting the Colmap application (Step1)

(b) Customizing the reconstruction (Step 2 - 7)

Figure 6.3: Setting up the automatic reconstruction of the object from images



Imported the reconstructed sparse models for visualization. The models were also exported to the *sparse* sub-folder in the workspace.

To visualize the reconstructed dense point cloud, navigate to the *dense* sub-folder in your workspace with *File > Import model from...*. To visualize the meshed model, you must use an external viewer such as Meshlab.

(a) Pop up message 1 for sparse results generation

(b) Pop up message 2 for dense results generation

Figure 6.4: Results of a successful auto reconstruction

6.3. Open-source Photogrammetry Software

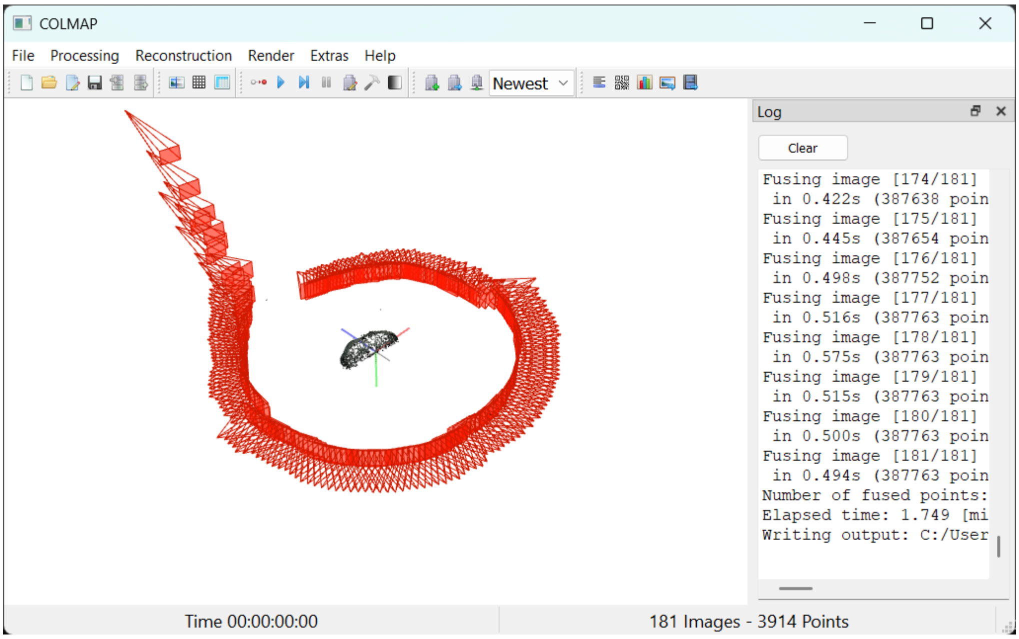
Prusa, the 3D printing company, is offering guidelines on available resources for creating 3D scans of objects[2]. Two different software are required to achieve a 3D scan: a photogrammetry software that can process the images and create an initial mesh of the 3D model and subsequently a software to refine the created mesh.

Colmap

Colmap is an open-source Structure-from-Motion (SfM) and Multi-Video Stereo (MVS) pipeline offering an automatic reconstruction tool[3]. It is important to note that this software requires a Compute Unified Device Architecture (CUDA) capable Graphics Processing Unit (GPU). For the result presented in this chapter, an NVIDIA Quadro P1000 GPU was used. While it is possible to use the software without a CUDA system, several features will not be available, including the dense point cloud reconstruction required for the creation of the 3D surface model.

[2] Photogrammetry – 3D scanning https://blog.prusa3d.com/photogrammetry-3d-scanning-just-phone-camera_7811/ (accessed: 1 May 2025)

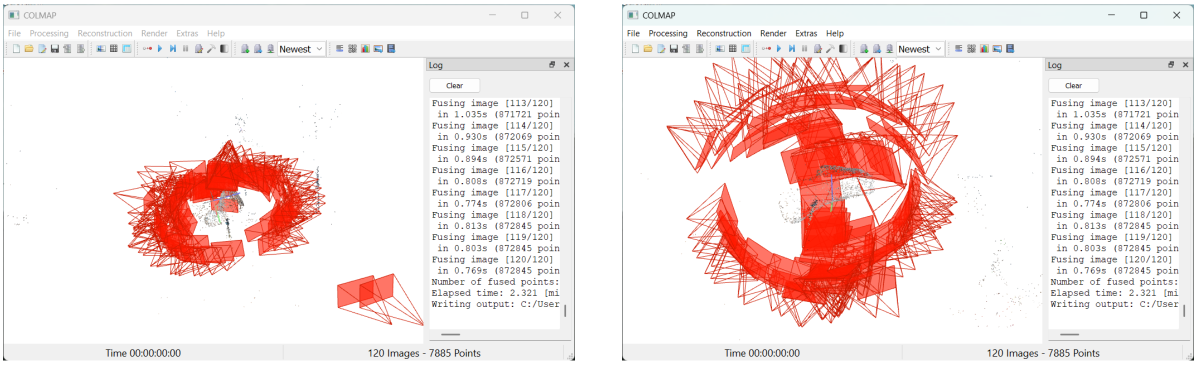
[3] Colmap Cuda release: https://github.com/colmap/colmap/releases/tag/3.11.1 (accessed 11 April 2025)



```
Fusing image [174/181]
 in 0.422s (387638 poin
Fusing image [175/181]
 in 0.445s (387654 poin
Fusing image [176/181]
 in 0.498s (387752 poin
Fusing image [177/181]
 in 0.516s (387763 poin
Fusing image [178/181]
 in 0.575s (387763 poin
Fusing image [179/181]
 in 0.515s (387763 poin
Fusing image [180/181]
 in 0.500s (387763 poin
Fusing image [181/181]
 in 0.494s (387763 poin
Number of fused points:
Elapsed time: 1.749 [mi
Writing output: C:/User
```

Time 00:00:00:00 181 Images - 3914 Points

Figure 6.5: Finished reconstruction of 433-Eros using experimental images from a 100 km orbit under orbital lighting conditions



```
Fusing image [113/120]
 in 1.035s (871721 poin
Fusing image [114/120]
 in 0.930s (872069 poin
Fusing image [115/120]
 in 0.894s (872571 poin
Fusing image [116/120]
 in 0.808s (872719 poin
Fusing image [117/120]
 in 0.774s (872806 poin
Fusing image [118/120]
 in 0.813s (872845 poin
Fusing image [119/120]
 in 0.803s (872845 poin
Fusing image [120/120]
 in 0.769s (872845 poin
Number of fused points:
Elapsed time: 2.321 [mi
Writing output: C:/User
```

Time 00:00:00:00 120 Images - 7885 Points

```
Fusing image [113/120]
 in 1.035s (871721 poin
Fusing image [114/120]
 in 0.930s (872069 poin
Fusing image [115/120]
 in 0.894s (872571 poin
Fusing image [116/120]
 in 0.808s (872719 poin
Fusing image [117/120]
 in 0.774s (872806 poin
Fusing image [118/120]
 in 0.813s (872845 poin
Fusing image [119/120]
 in 0.803s (872845 poin
Fusing image [120/120]
 in 0.769s (872845 poin
Number of fused points:
Elapsed time: 2.321 [mi
Writing output: C:/User
```

Time 00:00:00:00 120 Images - 7885 Points

(a) Finished view of the reconstructed scene

(b) Rotated scene with visible point cloud of 433-Eros

Figure 6.6: Finished reconstruction of 433-Eros using images of the asteroid model under natural lighting conditions

The following steps were taken to process the images using Colmap:

1. Launch `Colmap.bat` and select Reconstruction → Automatic Reconstruction (Figure 6.3a).
2. Set the **Workspace Folder** to a directory adjacent to the images folder. This folder stores all intermediate files and final outputs (Figure 6.3b).
3. Set the **Image Folder** to the directory containing the images (Figure 6.3b).
4. Select *Medium* quality to balance reconstruction fidelity against the stability of creating a solution. *High* quality mode frequently induces software crashes, in particular on lower-performing computers (Figure 6.3b).
5. Check that both Sparse and Dense model generation are selected (Figure 6.3b). The sparse model is not sufficient to create the mesh of the 3D object. However, its outputs are directly used to create the dense model. The dense model mesh includes the fused elements required to finalise the mesh, which will be further refined using Meshlab.
6. Check that GPU is selected. The dense model will not be generated without a GPU setting activated.
7. Retain all other parameters at their defaults and run the program.

Depending on the number of images and the specifications of the host computer, the process requires between a few minutes and several hours. To generate the results showcased in this work, the generation time was around one hour. After a successful generation of the models, two pop-up windows should appear in the application showcased in Figure 6.4a and Figure 6.4b.

Figures 6.3a - 6.5 illustrate the key stages: application launch, parameter configuration, sparse and dense reconstruction prompts, and the coloured point cloud of 433-Eros. The successfully reconstructed point clouds from the two different datasets used in this work are visualised in Figure 6.6 for the diffuse natural lighting conditions and Figure 6.5 for the experimental images from the darkroom of the GNC-Lab. The saving destination of the dense model created for each case is restated on the right-hand side panel, and it should match the Workspace destination folder selected under Step 2. The image frames relative positioning and orientation used in the reconstruction process are visible in red around the point cloud. It is interesting to note the difference between the experimental dataset, where only equatorial images were taken of the asteroid model, and the other data set, using images obtained in the control room, which uses images angled from above and below the asteroid model. How this influences the results is further discussed in Section 6.4.

Meshlab

MeshLab[4] provides a suite of mesh processing and surface reconstruction filters. To refine the meshes generated by Colmap, the following steps are necessary:

1. Import the Colmap-generated mesh via File → Import Mesh (Figure 6.7).
2. Remove outlier vertices and isolated components manually using selection tools (Figure 6.8).
3. Apply Filters → Remeshing, Simplification and Reconstruction → Poisson Surface Reconstruction (Figure 6.9). The key parameter is reconstruction depth, which starts as default as 8 (Figure 6.10b). Increasing this parameter improves the surface generation smoothness and the associated number of faces and vertices generated. However, the generation time of the mesh also increases. It has been observed that for values above 16, the application crashes. The cleaned mesh used as input for the Remeshing, Simplification and Reconstruction step is shown in Figure 6.10a.
4. Inspect and export the reconstructed surface in .ply format to preserve the mesh or .stl for 3D printing.

Figures 6.8 to 6.11b show the intermediate cleaning, parameter settings and the final reconstructed mesh from experimental data (Dataset 1). The same steps have been performed for Dataset 2. For this dataset, additional steps were required to filter the initial point cloud reconstructed from Colmap as shown in Figure 6.12. The reconstructed results are shown in Figure 6.13. The particularities of the process for each dataset, along with the discussion on the quality of the generated results, are presented in Section 6.4.

[4] Meshlab download https://www.meshlab.net/ (accessed:9 May 2025)

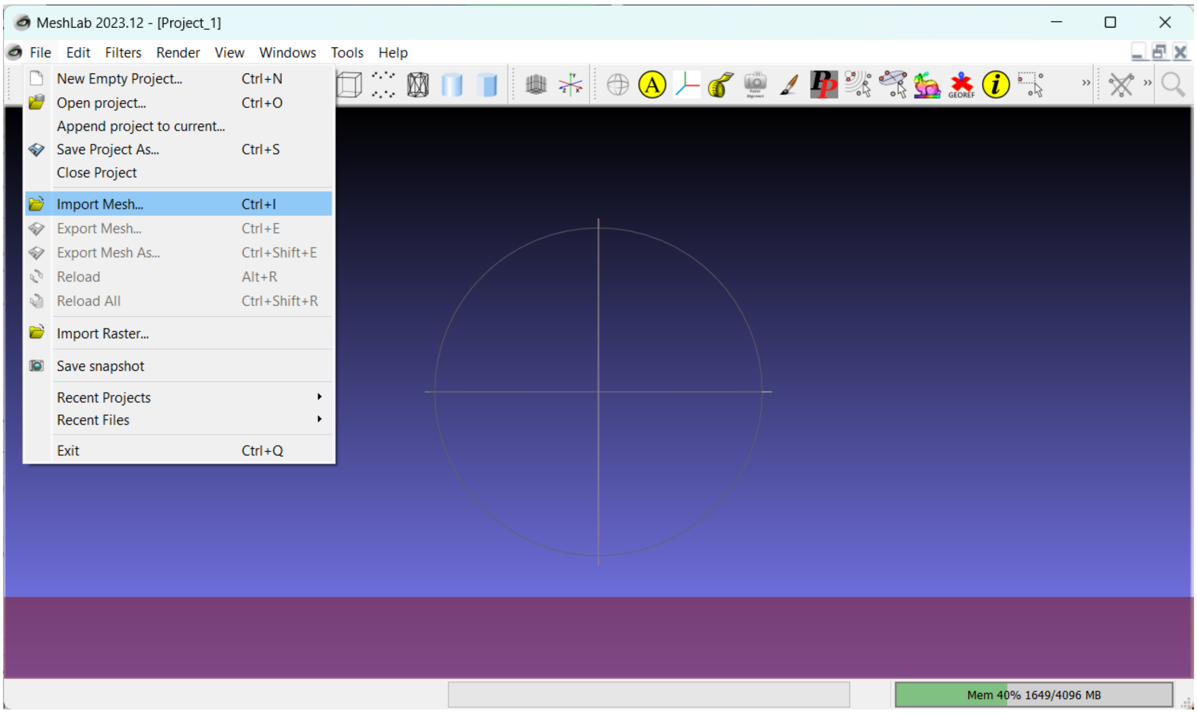
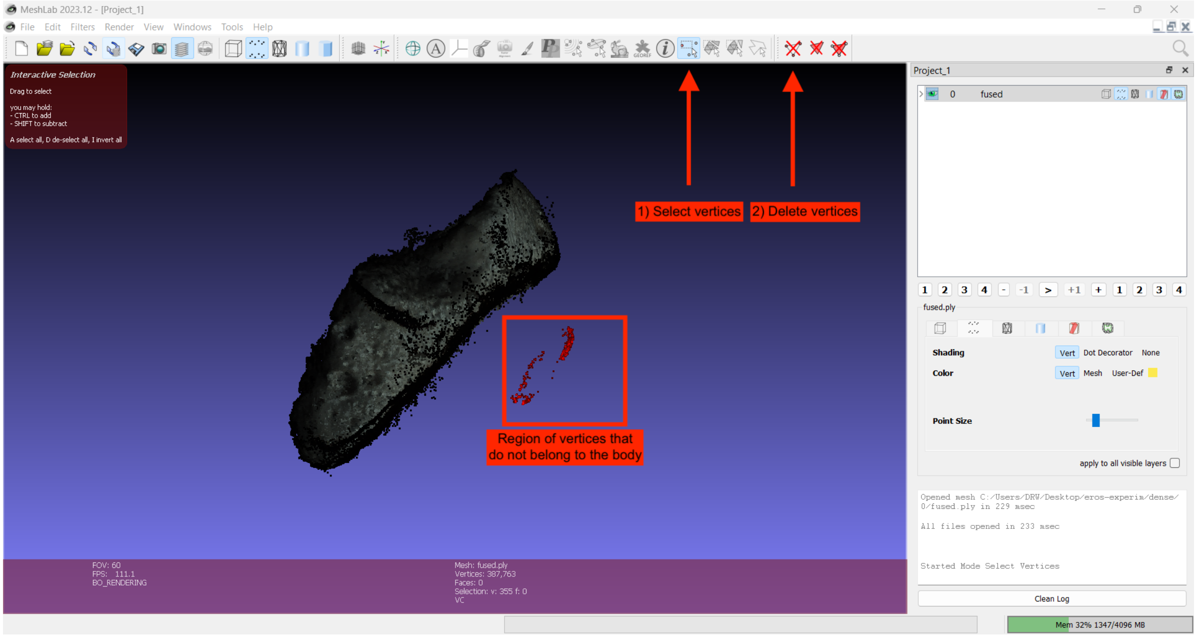


Figure 6.7: Meshlab launch window showing how to import the mesh generated from Colmap (Step 1)



Interactive Selection

Drag to select

you may hold:
- CTRL to add
- SHIFT to subtract

A select all, D de-select all, I invert all

FOV: 60
FPS: 111.1
BO_RENDERING

Mesh: fused.ply
Vertices: 387,763
Faces: 0
Selection: v: 355 f: 0
VC

1) Select vertices

2) Delete vertices

Region of vertices that do not belong to the body

fused.ply

Shading: Vert | Dot Decorator | None

Color: Vert | Mesh | User-Def

Point Size

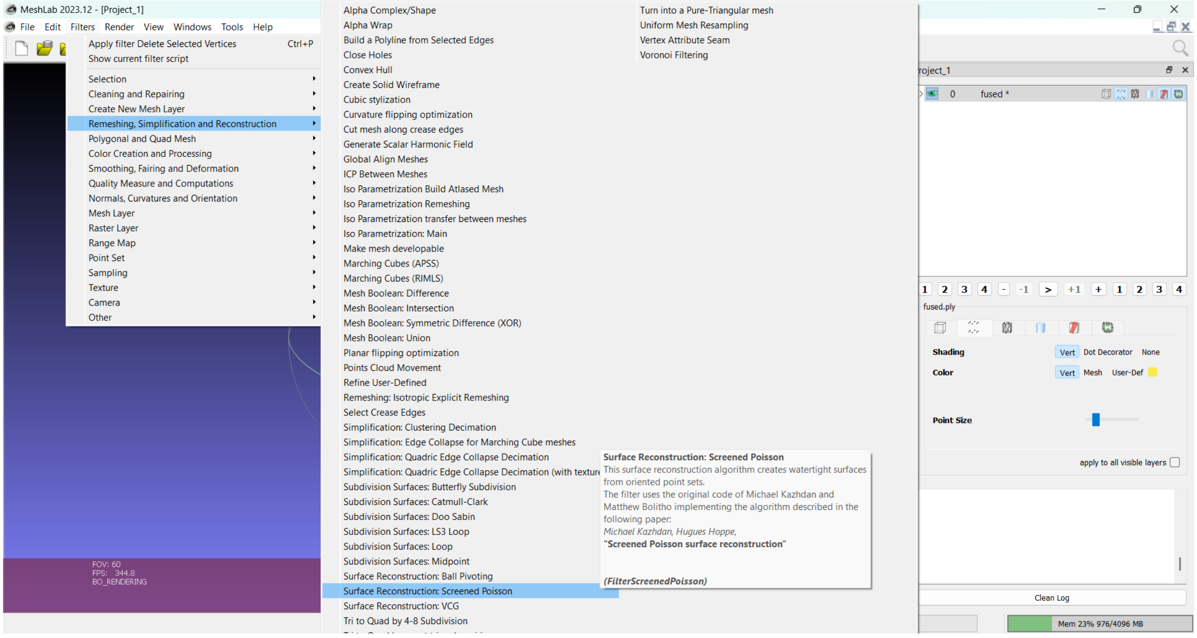
apply to all visible layers □

```
Opened mesh C:/Users/DRW/Desktop/eros-experim/dense/
0/fused.ply in 229 msec

All files opened in 233 msec


Started Mode Select Vertices
```

Figure 6.8: Removing points that do not belong to the reconstructed body (Step 2)

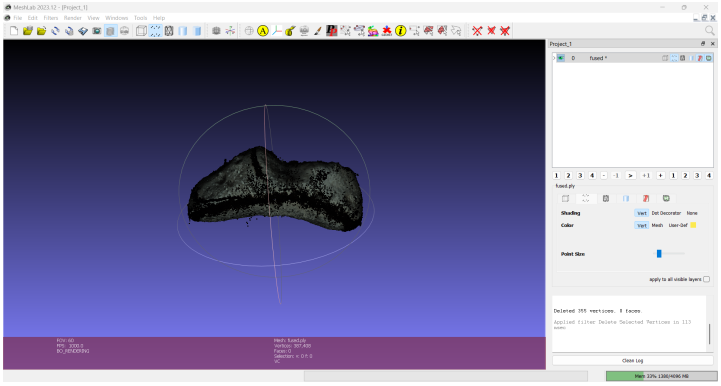

MeshLab 2023.12 - [Project_1]
File Edit Filters Render View Windows Tools Help
Apply filter Delete Selected Vertices Ctrl+P
Show current filter script
Selection
Cleaning and Repairing
Create New Mesh Layer
Remeshing, Simplification and Reconstruction
Polygonal and Quad Mesh
Color Creation and Processing
Smoothing, Fairing and Deformation
Quality Measure and Computations
Normals, Curvatures and Orientation
Mesh Layer
Raster Layer
Range Map
Point Set
Sampling
Texture
Camera
Other
Alpha Complex/Shape
Alpha Wrap
Build a Polyline from Selected Edges
Close Holes
Convex Hull
Create Solid Wireframe
Cubic stylization
Curvature flipping optimization
Cut mesh along crease edges
Generate Scalar Harmonic Field
Global Align Meshes
ICP Between Meshes
Iso Parametrization Build Atlased Mesh
Iso Parametrization Remeshing
Iso Parametrization transfer between meshes
Iso Parametrization: Main
Make mesh developable
Marching Cubes (APSS)
Marching Cubes (RIMLS)
Mesh Boolean: Difference
Mesh Boolean: Intersection
Mesh Boolean: Symmetric Difference (XOR)
Mesh Boolean: Union
Planar flipping optimization
Points Cloud Movement
Refine User-Defined
Remeshing: Isotropic Explicit Remeshing
Select Crease Edges
Simplification: Clustering Decimation
Simplification: Edge Collapse for Marching Cube meshes
Simplification: Quadric Edge Collapse Decimation
Simplification: Quadric Edge Collapse Decimation (with textur
Subdivision Surfaces: Butterfly Subdivision
Subdivision Surfaces: Catmull-Clark
Subdivision Surfaces: Doo Sabin
Subdivision Surfaces: LS3 Loop
Subdivision Surfaces: Loop
Subdivision Surfaces: Midpoint
Surface Reconstruction: Ball Pivoting
Surface Reconstruction: Screened Poisson
Surface Reconstruction: VCG
Tri to Quad by 4-8 Subdivision
Turn into a Pure-Triangular mesh
Uniform Mesh Resampling
Vertex Attribute Seam
Voronoi Filtering
roject_1
0 fused *
1 2 3 4 - -1 > +1 + 1 2 3 4
fused.ply
Shading Vert Dot Decorator None
Color Vert Mesh User-Def
Point Size
apply to all visible layers □
Clean Log
Mem 23% 976/4096 MB
FOV: 60
FPS: 344.8
BO_RENDERING

Surface Reconstruction: Screened Poisson

This surface reconstruction algorithm creates watertight surfaces from oriented point sets.
The filter uses the original code of Michael Kazhdan and Matthew Bolitho implementing the algorithm described in the following paper:
Michael Kazhdan, Hugues Hoppe,
"Screened Poisson surface reconstruction"

(FilterScreenedPoisson)

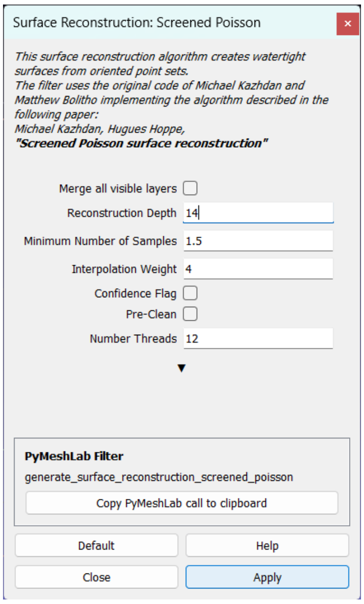
Figure 6.9: Guidelines on how to apply surface reconstruction to the imported mesh (Step 3)


MeshLab 2023.12 - [Project_1]
File Edit Filters Render View Windows Tools Help
Project_1
0 fused *
1 2 3 4 - -1 > +1 + 1 2 3 4
fused.ply
Shading Vert Dot Decorator None
Color Vert Mesh User-Def
Point Size
apply to all visible layers □
Clean Log
Mem 33% 1380/4096 MB

```
Deleted 355 vertices, 0 faces.

Applied filter Delete Selected Vertices in 113
msec
```

(a) 433-Eros mesh from experimental data after most vertices that did not belong to the body have been removed (Input mesh for Step 3)



Surface Reconstruction: Screened Poisson

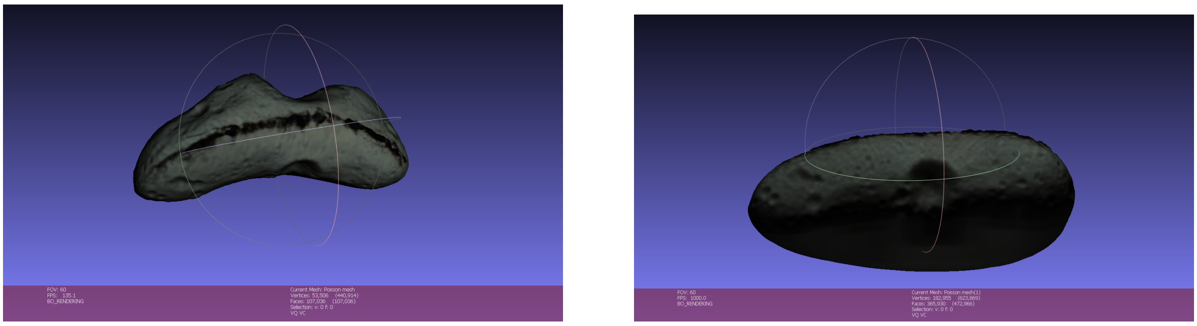
This surface reconstruction algorithm creates watertight surfaces from oriented point sets.
The filter uses the original code of Michael Kazhdan and Matthew Bolitho implementing the algorithm described in the following paper:
Michael Kazhdan, Hugues Hoppe,
"Screened Poisson surface reconstruction"

Merge all visible layers □
Reconstruction Depth 14
Minimum Number of Samples 1.5
Interpolation Weight 4
Confidence Flag □
Pre-Clean □
Number Threads 12

PyMeshLab Filter
generate_surface_reconstruction_screened_poisson

(b) Surface reconstruction settings

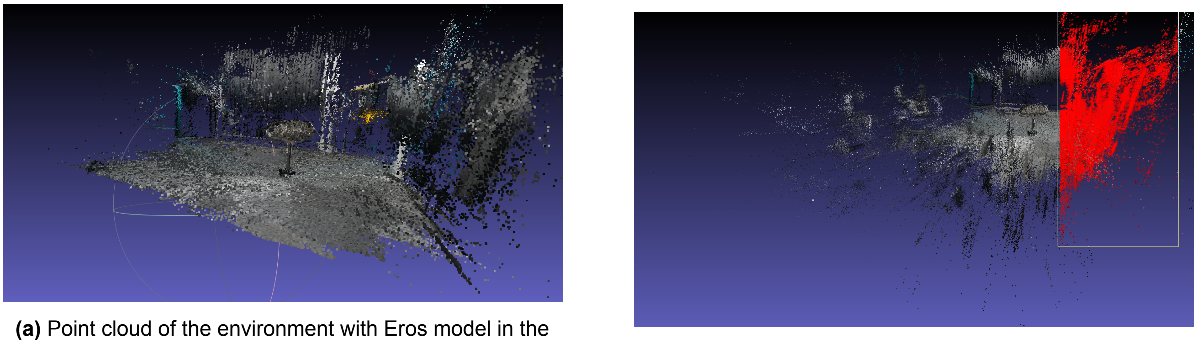
Figure 6.10: Tuning of the reconstruction process (Step 3)



(a) Top view

(b) Side view

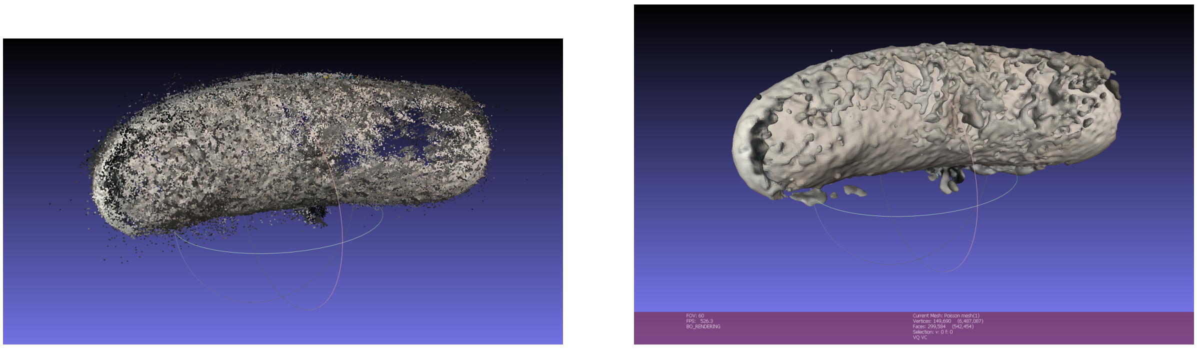
Figure 6.11: 433-Eros surface reconstruction from experimental images Dataset 1 (Step 4)



(a) Point cloud of the environment with Eros model in the middle

(b) Removing vertices process for Dataset 2

Figure 6.12: Initial point cloud produced from Dataset 2



(a) Point cloud used as input to Step 3 for Dataset 2

(b) Reconstructed surface for Dataset 2

Figure 6.13: 433-Eros surface reconstruction from smartphone images Dataset 2 (Step 4)

6.4. Discussion on 3D Surface Reconstruction Capabilities

Experimental Dark Room Images (Dataset 1) The results of the proposed reconstruction method using Dataset 1 are presented in Figure 6.11. A reconstruction depth of 14 was selected. Although the hard shadows in the dataset create some reconstruction artefacts, the overall top surface and general asteroid shape have been successfully reconstructed. The shadow introduces an artificial feature on the top surface where no direct image measurements are available. Moreover, the central cut-out of the asteroid has not been reconstructed.

Hard shadows along the lower half prevent the software from identifying features. When the asteroid's sides are visible, the mesh-refinement stage encloses the surface, as illustrated in Figure 6.11b, yet the outcome does not match the true shape of the model. This limitation must be considered when designing laboratory experiments. A light source placed directly above the asteroid is insufficient for surface reconstruction, and greater luminosity is required. One alternative is to place an additional light source near the camera. This approach has also been encountered in the literature. During the Hayabusa mission, for instance, the spacecraft carried a flashlight to enhance the visibility of artificial landmarks deployed on the surface of the asteroid (Ogawa et al., 2020). Nonetheless, the Colmap–MeshLab workflow proved robust under varying lighting, as it was able to recreate a shape model even under the limited lighting conditions from the experimental dark room.

To evaluate the photogrammetric reconstruction, the shape model produced by Gaskell (2020) from NEAR Shoemaker data is adopted as a reference. The original model contains 49,152 faces (Figure 6.14). Earlier work simplified this mesh to 5,144 faces to reduce gravity modelling computational costs (Spee, 2022). By contrast, the reconstruction obtained here comprises 182,955 vertices and 365,930 faces (Figure 6.15) owing to the chosen depth. A higher vertex count preserves fidelity, especially in regions of pronounced curvature such as concavities and ridges, and was therefore preferred to recover detail in areas with lower image

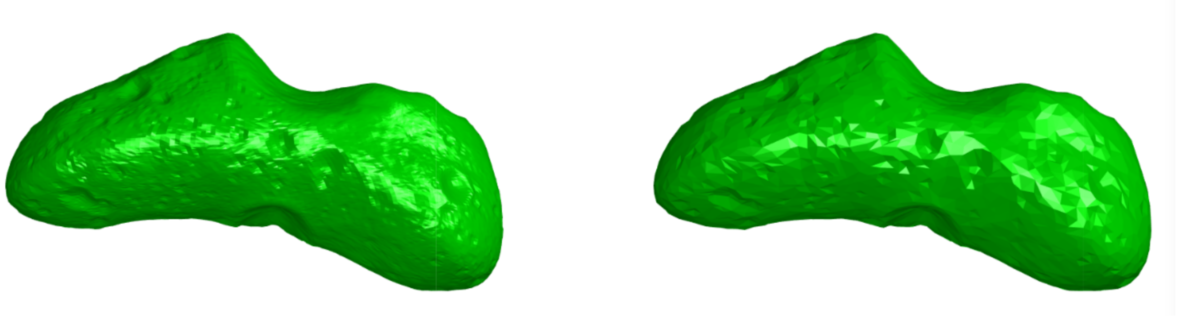


Figure 6.14: Original shape model of 433-Eros from Gaskell et al. (2006) (left) Eros Shape Model 49,152 faces; (right) Simplified Eros Shape Model 5,144 faces (Spee, 2022)

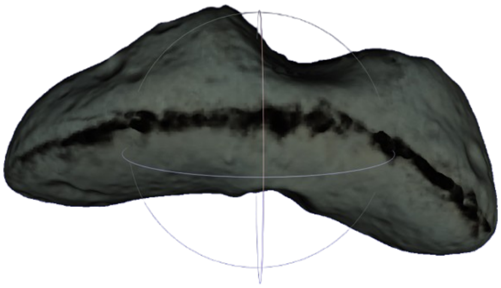


Figure 6.15: Shape model of 433-Eros reconstructed from experimental measurements images. The reconstructed model has 182,955 vertices and 365,930 faces.

coverage, and to assess how well this proposed method can reproduce features from the 3D model. Nonetheless, the reconstruction depth can be adjusted to reduce the number of faces for different applications.

Smartphone Images (Dataset 2) Dataset 2 was captured under lighting adjusted to remove dark shadows. It was anticipated that this would improve reconstruction. However, inspection of the resulting point cloud in MeshLab (Figure 6.12) reveals a different issue. The software reconstructs features from the surrounding laboratory environment, so extensive cleaning is required to retain only points belonging to the asteroid. Because the environmental points dominate the point cloud, the object of interest is not centred after removal. Centre the remaining points can be done via Filters → Normals, Curvatures and Orientation → Transform: Translate, Centre, set Origin. The preview checkbox to select to visualise the desired translation. Deselect preview before applying the transform. Further vertex removal can be performed after the translation, as it is easier to identify which points belong to the body.

Laboratory reflections and shiny surfaces severely degrade the asteroid point cloud, which is considerably sparser than that derived from Dataset 1. The irregular reflections are interpreted as surface texture, introducing artefacts after remeshing (Figure 6.13). Although increased illumination helps to capture the previously shadowed lower hemisphere, the cluttered reflective background introduces false features. Future datasets should therefore be acquired against a simpler background, supplemented by an additional light source.

Degree of autonomy of surface reconstruction As summarised in Chapter 2, surface scanning and 3D reconstruction constitute the first stage of asteroid navigation. The resulting mesh defines landmarks that enable relative state estimation. Achieving full on-board autonomy for surface reconstruction and relative position has not yet been achieved, and it is one of the main objectives of the Hera mission. The current practice is to generate the shape model on the ground using high-performance computers. This approach is also recommended for the development of a robotic testbed for relative asteroid navigation, given the current stage of development of the GNC laboratory. The resulting 3D model can then be uploaded to the navigation filter. If the image of the model is transmitted from the robotic system, then it can be compared to the asteroid model previously reconstructed.

Photogrammetry limitations A few mentions should be made about the limitations of this proposed method. Most importantly, photogrammetric models have proportion but do not have scale. In order to scale the model, there must be at least one known distance[5]. Therefore, while the model produced maintains the proportions observed from the image dataset, in order to use the 3D model for further analysis, it is imperative that the mesh structure of the asteroid is scaled.

A second constraint is that the object must remain stationary while the camera moves around it. With the current laboratory hardware and trajectory generation capabilities, this assumption holds, and the method yields 3D shape models whose face and vertex counts are comparable with those reported by Gaskell (2020). As the facility advances and experiments reproduce realistic asteroid spin states, however, the body's rotation will have to be considered. In that case, additional topographic and albedo information will become necessary for surface reconstruction.

[5] Artec3D Photogrammetry https://www.artec3d.com/learning-center/what-is-photogrammetry (accessed: 1 June 2025)

Constructing a full topography model would require an auxiliary lidar sensor for altimetry. More advanced image processing techniques would be required, such as stereophotoclinometry SPC. In SPC stereo (Adam et al., 2019) refers to the use of multiple images to determine feature location, as was the case with the Photogrammetry model proposed in this work. The second term, Photoclinometry (Adam et al., 2019), describes a method of recreating shapes by using the shading of the different features in the images. This approach can be used for the creation of DTMs, which are then used as inputs for the surface reconstruction model as it has been described by the work of Gaskell et al. (2007) for Eros and Walsh et al. (2022) Bennu. Nonetheless, additional optimisation pipelines have to be created to evaluate the rotational rate. For this a reference frame has to be created from the measurements and additional optimisation process has to be implemented to best approximate the rotational rate of the observed body as described by the work of Preusker et al. (2015) for comet 67P/Churyumov-Gerasimenko and then evaluate its impact on the surface reconstruction model.

In summary, open-source photogrammetry pipelines can reconstruct detailed asteroid surfaces to be further used in robotic relative asteroid navigation. Careful dataset preparation is required to ensure that the asteroid covers a sufficiently large proportion of the image. Further research should explore hybrid methods that combine passive photogrammetry with laser scanning to mitigate the shadow-induced reconstruction errors or additional lighting sources.

Suitability for Ongoing Research within the Department The 3D surface models reconstructed in this chapter provide a functional basis for vision-based navigation tasks in the GNC laboratory. Their application requires accurate scaling and the removal of environmental artefacts to ensure compatibility with landmark-based navigation filters. For image-based pose estimation, such as landmark tracking or Perspective-n-Point (PnP) methods, the mesh must be referenced to a common coordinate system and include sufficient surface detail.

Although direct trajectory control of the camera on the robotic arm remains limited, static poses can be used to capture images from known configurations. From a single image, the camera pose relative to the asteroid can be estimated using the PnP algorithm. This approach requires identifying 2D–3D correspondences between features visible in the image and points on the surface mesh. A practical method is to manually select surface landmark such as concavities or ridges on the 3D model and associate them with visible features in the captured image. Once the correspondences are defined, the PnP algorithm can compute the six-degree-of-freedom pose of the camera within the laboratory reference frame.

Current research in the department evaluates how navigation filter performance varies depending on the source of image data. Comparative studies are being conducted using images acquired in the laboratory under controlled lighting and synthetic views generated from the original 433-Eros model in Blender. These experiments aim to quantify the effects of lighting, resolution, and noise on relative pose estimation.

The surface model created in this study can extend this research. Once scaled and cleaned, it can be used to generate synthetic datasets for neural network training or feature detection pipelines. Rendering the model under varying viewpoints and illumination conditions enables the creation of datasets that closely replicate the laboratory environment. This supports the validation of perception algorithms and provides a bridge between experimental and simulated navigation scenarios.

7 Conclusions and Recommendations

The work performed in this master thesis has set out to investigate:

How can the development of autonomous asteroid navigation be supported in a robotic-lab environment?

Asteroid mission planning is characterised by three main phases: initial global characterisation, detailed global characterisation, and execution of scientific objectives. This study focuses on the initial global characterisation phase, evaluating how laboratory-based systems can support the development of autonomous navigation.

Support for such development is achieved by configuring the robotic laboratory as a kinematic testbed capable of replicating translational and rotational motion. To this end, the performance of hardware within TU Delft's Guidance, Navigation, and Control (GNC) laboratory has been assessed. The testbed comprises a 3 degrees of freedom (dof) omnidirectional RB-Kairos rover, a 6 dof UR16e robotic arm, a 10:1,000,000 scale model of asteroid 433-Eros, and an OptiTrack motion-capture system. A python framework was developed to define reference frame transformations across all hardware components, enabling generation of synchronised command sequences for the robotic platform. A modular Gazebo simulation environment was implemented to validate commands and assess trajectory feasibility prior to physical deployment, enhancing the demonstration capabilities of the GNC laboratory.

The open-loop kinematic performance of the system was evaluated through trajectory execution experiments, including circular paths around the asteroid model and target tracking paths with the robotic arm. These tests revealed that the open-loop control strategy introduces notable positional errors. For a 1200 mm radius circular path of the rover base, deviations reached up to 370 mm (width) and 276 mm (length). Additional uncertainties in the robotic arm's mounting position produced mean errors of -39 mm to 66 mm (width), -46 mm to 57 mm (length), and 0.76 mm to 12.96 mm (height). These findings indicate that closed-loop control is required to mitigate cumulative trajectory errors and improve positional accuracy.

In summary, the developed framework confirms that autonomous asteroid navigation can be supported in a robotic-lab environment, provided that positional control is improved. This work establishes a baseline open-loop performance of the robotic system for simulating asteroid navigation tasks and sets the foundation for future expansion of the laboratory's experimental capabilities.

7.1. Addressing Research Sub-Questions

Experiment Design

1(a) **How can a robotic system be designed and utilised to simulate the navigation around a 3D-printed model of Eros?** The proposed testbed integrates available hardware, respectively: RB-Kairos rover, UR16e robotic arm equipped with a camera, a 10:1,000,000 scale model of 433-Eros, and an OptiTrack motion-capture system. The base provides translational motion to simulate orbital paths, while the arm enables fine positioning and target tracking of the 433-Eros asteroid model. The robotic system is commanded using a command file executed via Robot Operating System (ROS) nodes, coordinating base and arm motion with camera data acquisition. A Gazebo digital twin supports pre-execution of trajectory commands, enabling safe and reliable verification of the commands

1(b) **What parameters need to be replicated in the laboratory trajectory to be representative of the space environment?** Representative trajectories require accurate correlation between the asteroid model scale, trajectory dimensions, and laboratory workspace. This work employs a 10:1,000,000 scale model of asteroid 433 Eros. Within the physical constraints of the testbed, circular Keplerian trajectories at 100-300 km altitude can be safely simulated using constant-velocity commands.

1(c) **What are the requirements for the external monitoring of the setup using a tracking system like OptiTrack?** High-frequency, accurate external monitoring is required to capture motion trajectories and detect positional deviations. The OptiTrack system, operating at 240 frames per second (fps), provides adequate temporal resolution. While the manufacturer specifies a positional accuracy of 0.3 mm, experimental results showed vertical variations of up to 30 mm in the rover base markers. This discrepancy requires further investigation to identify error sources and define a reliable calibration benchmark for the GNC laboratory workplace.

Model Accuracy and Detail

What level of detail and accuracy is required in the 3D-printed model of Eros to ensure reliable navigation and tracking? A minimum of 49,152 faces is required in the surface mesh of 433-Eros to ensure sufficient geometric fidelity for landmark recognition in asteroid relative navigation. The photogrammetry method introduced in this study enables assessment of model accuracy by generating reconstructions of comparable resolution. For gravity field modelling, lower mesh complexity is acceptable. The photogrammetry pipeline allows resolution to be adapted to experimental needs, from high-detail tracking to reduced-resolution gravity field estimations required for spacecraft state estimation in perturbed orbits.

Experiment Evaluation

3(a) **To what extent can the robotic system accurately replicate the trajectory profiles required for relative asteroid navigation?** Trajectory replication was evaluated in three cases: base-only motion (Case 1), arm-based target tracking (Case 2), and coordinated motion planning (Case 3). In Case 1, circular paths with a 1200 mm radius showed RMSE up to 370.28 mm (x) and 339.24 mm (z), with vertical errors under 10 mm. Translational tests yielded consistent undershoot, with RMSE of 306.11 mm (x) and 332.07 mm (z). Case 2 demonstrated improved accuracy using the UR16e arm, achieving RMSE between 43.89 mm and 69.71 mm (x), and 52.06 mm to 77.55 mm (z). During dynamic camera tracking of the asteroid model, average errors were –36.91 mm (x), 12.96 mm (y), and –52.36 mm (z), with RMSE values of 53.65 mm, 18.96 mm, and 84.97 mm, respectively. These results confirm that the testbed can reproduce representative trajectories for the initial global characterisation phase. However, open-loop control and sensor calibration drift introduce cumulative errors, motivating the implementation of closed-loop correction for improved repeatability and accuracy.

3(b) **Which features of autonomous navigation around an asteroid can be tracked using the available set-up?** The experimental setup supports key elements of the initial global characterisation phase of asteroid missions. The robotic base enables circular and arc-like trajectories around the 433-Eros model, emulating simplified Keplerian motion. The UR16e robotic arm supports coordinated camera pointing and target tracking, enabling dynamic adjustment of sensor orientation with respect to the asteroid model. Surface reconstruction was demonstrated using a photogrammetry pipeline, confirming the ability to generate surface reconstructions suitable for visual navigation. The testbed, therefore, enables experimental replication of initial target body characterisation, providing a foundation for testing relative asteroid navigation.

Returning to the main research question, the combined results from the experiment design, hardware characterisation, and surface reconstruction lead to the following answer to: *How can the development of autonomous asteroid navigation be supported in a robotic-lab environment?*

Autonomous asteroid navigation can be supported in a robotic-lab environment by configuring a modular kinematic testbed that reproduces simplified orbital motion and optical tracking conditions relevant to initial global characterisation. This study demonstrates the feasibility of executing representative circular and arc trajectories with open-loop Root Mean Square Error (RMSE) errors ranging from 306 mm to 370 mm, and improves accuracy through camera tracking with the robotic arm achieving RMSE below 85 mm. Surface reconstruction via photogrammetry confirms the capability to simulate landmark-based navigation, establishing a foundation for closed-loop control and autonomous navigation algorithm development in future work.

7.2. Specification Document

The main challenge of developing a robotic testbed for relative asteroid navigation was defining requirements. Due to the link between mission type and system design, exploratory work assessed feasible missions and required accuracy for relative navigation results. This provides a foundation for future experimental campaigns.

MISSION

REQ-MIS-01: The spacecraft (S/C) shall orbit asteroid 433-Eros at orbital altitudes between 100 km and 300 km

REQ-MIS-02: The navigation system shall determine the S/C position with an accuracy of better than 10 m (3σ) during the orbit.

REQ-MIS-03: The S/C shall capture at least 180 images of Eros during one full rotation period, ensuring a minimum image overlap of 70% between consecutive frames.

SYSTEM

REQ-SYS-01: The lab simulator shall provide at least 3 degrees of freedom (DoF): 2 linear translations along the horizontal plane and 1 rotation about the local vertical axis.

REQ-SYS-02: The simulator shall replicate the relative geometry for manoeuvrers with one fixed object and one actuated vehicle, ensuring positional error <5 mm (3σ) between the expected and observed positions. **REQ-MIS-01**

REQ-SYS-03: The generated spacecraft trajectories shall not exceed the physical lab boundary (3.9 m × 8.7 m area). **REQ-MIS-01**

REQ-SYS-04: The trajectories shall ensure continuous camera asteroid line-of-sight.

REQ-SYS-05: The light source shall have a colour temperature of 5800 K ± 200 K.

REQ-SYS-06: The OptiTrack marker placement shall ensure no obstruction of camera measurements.

REQ-SYS-07: The conversion of trajectories into robot joint space shall remain within the robots' safe working envelope defined by the manufacturer, maintaining joint angles and velocities under specified safety limits.

REQ-SYS-08: Hardware components: RB-Karios, UR16e, MER2 commands and OptiTrack measurements shall synchronize state readings within <10 ms latency from each other.

REQ-SYS-09: The end-effector commanded position shall match the measured position within ±0.1 mm. **REQ-MIS-02**

REQ-SYS-10: The chaser RB-Karios rover base shall create revolutions around the target RB-Karios rover base with orbital distances between 1.089 m and 1.481 m. These values are justified in Section 3.2.

REQ-SYS-11: The stationary target rover (Robot 1) shall be preconfigured with joint angles of [3.080, –2.6279, 2.6279, –1.570796, 1.570796, 3.1415] rad for Base, Shoulder, Elbow, Wrist 1, Wrist 2, and Wrist 3 joints respectively when holding the Eros-433 model. These values are justified in Section 3.3.

REQ-SYS-12: The robotic system (Robot 2) shall be able to track trajectories with a minimum height of 162 cm in the space of the laboratory. These values are justified in Section 3.3.

REQ-SYS-13: The motion-capture system shall use 240 fps for recording the motion profiles during experiments. These values are justified in Section 3.5.

TEST

REQ-TEST-01: No human presence shall be allowed within the defined lab safety perimeter during automatic robotic operations.

REQ-TEST-02: The OptiTrack system shall be calibrated daily before the first experiment by using the manufacturer's calibration tools (magic wand and calibration square).

REQ-TEST-03: Test scenarios shall be repeatable with less than ±5% variation in position and attitude accuracy for trajectory tracking over at least three identical runs.

7.3. Recommendations and Future Work

This section outlines recommendations for further development of the robotic testbed, based on the limitations identified in this work. The proposed improvements are categorised into system configuration, software development, navigation autonomy, and laboratory infrastructure. Together, they define a roadmap for expanding the Guidance, Navigation, and Control laboratory's capabilities to support autonomous asteroid relative navigation research.

System Configuration

C.1 **Remount the robotic arm.** The UR16e arm should be remounted so that its coordinate frame aligns the positive direction with the axes of the rover base. This alignment would simplify command interpretation, as the positive direction for both robotic components would align. Replacing the arm would remove the need to artificially flip the positive direction of generating waypoints in the software.

C.2 **Improve hardware localisation.** The initial manual alignment of the rover base and robotic arm introduces uncertainty. A laser alignment system mounted on the rover base could assist with accurate parallel placement relative to lab walls. To eliminate the need for the operator to be part of the calibration process, autonomous base alignment using lidar and odometry should be further investigated as a long-term solution.

C.3 **Evaluate and document OptiTrack accuracy.** The OptiTrack system has demonstrated vertical inconsistencies of up to 30 mm across the laboratory workspace, while variations below 0.3 mm were expected per the specifications provided by the manufacturer. A full calibration campaign should be conducted to quantify positional accuracy throughout the volume. The campaign should include testing at various heights and distances from the calibration square under varied lighting conditions.

C.4 **Investigate hardware start-up optimisation.** Frequent communication failures between the robotic arm and the rover base require system restarts, increasing test time significantly. It is recommended to investigate startup procedures, ROS driver synchronisation, and URCaps configurations to minimise protective stops and connection delays.

Software Development and Simulation Environment

S.1 **Continue development of the Gazebo simulation.** The digital twin environment developed in this work provides a safe pre-execution layer for trajectory planning. As interest in space robotics grows, an expanded simulation capability will support parallel projects and encourage collaborative development across the department.

S.2 **Create a closed-loop control architecture.** The current open-loop control strategy introduces trajectory errors. Incorporating closed-loop feedback, either through OptiTrack or on-board odometry, would allow the system to adjust commands dynamically and reach target positions with higher precision.

S.3 **Continue development of human robot interface.** A centralised Human Robot Interface (HRI) would enable teleoperation of both the UR16e and RB-Kairos. Tools such as Foxglove Studio, already tested in simulation Appendix C, can be integrated to visualise sensor data and trajectory plans in real time.

S.4 **Remove dependency on OptiTrack for positioning.** While useful for measurement, integrating OptiTrack data into the robot's command generation introduces complexity and error propagation. Alternatives include improved sensor fusion or object-relative navigation using camera feed. This approach contributes to relative asteroid navigation, however, it would not command the robotic system to absolute coordinates in the laboratory, which is the standard in the hardware simulators researched.

Navigation Autonomy

N.1 **Investigate fully relative asteroid navigation.** Rather than commanding robots to absolute lab coordinates, navigation could be based on the relative position of the camera with respect to the asteroid model. Vision-based object detection models may enable such relative positioning using real-time object detection with the camera mounted on the end effector.

N.2 **Estimate spherical harmonic coefficients.** The reconstructed shape model of Eros can be processed to compute its gravitational field and compare the resulting spherical harmonic coefficients to those from the benchmark model Gaskell (2020). This would enable validation of surface reconstruction fidelity for gravitational field estimation.

Laboratory Infrastructure and Illumination

L.1 **Implement adaptive lighting for the darkroom.** Lighting in the darkroom currently produces hard shadows that impair surface reconstruction. A ceiling-mounted lighting system with adjustable position would improve allow the robotic system greater placement flexibility. Mobile or rail-mounted lighting supports should be considered over the standing support.

L.2 **Improve surface reconstruction datasets.** Future image acquisition campaigns will require sunlight conditions, as has been done before. In addition, artificial lighting should be adopted to increase surface visibility and ensure sufficient asteroid coverage within each frame. The additional light source would reduce point-cloud contamination and facilitate consistent surface reconstruction quality.

References

Aber, J. S., Marzolff, I., & Ries, J. B. (2010). Chapter 3 - Photogrammetry. In *Small-Format Aerial Photography* (p. 23-39). Amsterdam, Netherlands: Elsevier. doi: doi.org/10.1016/B978-0-444-53260-2.10003-1

Adam, C. D., McCarthy, L. K., Sahr, E. M., Nelson, D. S., Pelgrift, J. Y., Lessac-Chenen, E. J., … Lauretta, D. S. (2019). Transition from Centroid-Based to Landmark-Based Optical Navigation During OSIRIS-REx Navigation Campaign at Asteroid Bennu (Tech. Rep.). Saratoga Springs, NY, United States: Goddard Space Flight Center. (Report Number: GSFC-E-DAA-TN74664)

Antreasian, P., Chesley, S., Miller, J., Bordi, J., & Williams, B. (2001). The design and navigation of the NEAR-Shoemaker landing on Eros. In (Vol. 109). (AAS 01-372)

Auguste, R., & Lumière, L. (2023). RB-KAIROS+ User Manual Revision v.1.1 (Tech. Rep.). Valencia, Spain: Robotnik Automation S.L.L.

Boge, T., Benninghoff, H., Zebenay, M., & Rems, F. (2012). Using Robots for Advanced Rendezvous and Docking Simulation. In *The Workshop on Simulation for European Space Programmes (SESP).* Noordwijk, Netherlands.

Boge, T., & Schreutelkamp, E. (2002). A new commanding and control environment for rendezvous and docking simulations at the EPOS-facility. In *Proc. 7th Int. Workshop on Simulation for European Space Programmes (SESP).* Noordwijk, Netherlands.

Bourgeaux, A. (2020). Autonomous estimation of the gravity field for asteroid missions (MSc Thesis). University of Bologna, Delft, Netherlands.

Brandt, D. (2024). Universal Robots User Manual UR16e (Tech. Rep.). Odense, Denmark: Universal Robots A/S. (Retrieved: https://www.universal-robots.com/download/manuals-e-seriesur20ur30/user/ur16e/56/user-manual-ur16e-e-series-sw-56-english-international-en/)

Cattani, B. (2022). Enhancing Collision Avoidance: Software and Hardware Techniques to Support and Validate GNC Developments (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Corke, P. (2017). *Robotics, Vision and Control: Fundamental Algorithms In MATLAB Second, Completely Revised, Extended And Updated Edition*. Queensland University of Technology (QUT), Brisbane, Australia: Springer.

Gaskell, R. (2020). Gaskell Eros Shape Model Bundle V1.0. In *NASA Planetary Data System.* urn:nasa:pds:gaskell.ast-eros.shape-model::1.0. doi: 10.26033/0hpf-4e64

Gaskell, R., Barnouin-Jha, O. S., & Scheeres, D. J. (2007). Modeling Eros with Stereophotoclinometry. In *Lunar and Planetary Science XXXVIII* (p. 1333). The Woodlands, Texas: Lunar and Planetary Institute.

Gaskell, R., BarnouinJha, O., Scheeres, D., Mukai, T., Hirata, N., Abe, S., … Kominato, T. (2006). Landmark navigation studies and target characterization in the Hayabusa encounter with Itokawa. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit.* Keystone, Colorado. (AIAA 2006-6660) doi: 10.2514/6.2006-6660

Hashimoto, T., Kubota, T., Kawaguchi, J., Uo, M., Shirakawa, K., Kominato, T., & Morita, H. (2010). Vision-based guidance, navigation, and control of Hayabusa spacecraft - Lessons learned from real operation. *IFAC Proceedings Volumes*, *43*(15), 259-264.

Hatch, H. G., Pennington, J. E., & Cobb, J. B. (1967). Dynamic simulation of lunar module docking with Apollo command module in lunar orbit (Tech. Rep.). Hampton, Virginia, United States: National Aeronautics and Space Administration. (TN D-3972)

Krüger, H., & Theil, S. (2010). TRON - Hardware-in-the-Loop Test Facility for Lunar Descent and Landing Optical Navigation. *IFAC Proceedings Volumes*, *43*(15), 265-270. (18th IFAC Symposium on Automatic Control in Aerospace) doi: 10.3182/20100906-5-JP-2022.00046

Küppers, M., & Gil Fernandez, J. (2021). Hera Mission Requirement Document Issue 2.10 (Tech. Rep.). Camino bajo del Castillo, Madrid, Spain: The European Space Agency ESAC. (ESA-TECSH-RS-022793)

Mastrodemos, N., Rush, B., Vaughan, A., & Owen, W. (2011). Optical Navigation For The Dawn Mission At Vesta. *Advances in the Astronautical Sciences*, *140*, 1739-1754.

Miller, J. K., Williams, B. G., Bollman, W. E., Davis, R. P., Helfrich, C. E., Scheeres, D. J., ... Yeomans, D. K. (1995). Navigation Analysis for Eros Rendezvous and Orbital Phases. *The Journal of the Astronautical Sciences*, *43*(4), 453-476.

Mooij, E. (2024). *Re-entry Systems*. Springer. (Springer Aerospace Technology; Vol. Part F3992) doi: 10.1007/978-3-031-62174-1

Mooij, E., Root, B. C., & Bourgeaux, A. (2024). Gravity-Field Estimation of Asteroids. In *AIAA SCITECH 2024 Forum.* Orlando, FL: American Institute of Aeronautics and Astronautics. doi: 10.2514/6.2024-2275

Morita, H., Shirakawa, K., Kubota, T., Hashimoto, T., & Kawaguchi, J. (2006). Hayabusa's Real-time Landmark Tracking Navigation for Descents and Touching-Downs. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit* (pp. 1–11). (AIAA 2006-6537) doi: 10.2514/6.2006-6537

Munuera Vilalta, M. (2024). Asteroid Gravity Field Estimation by a Satellite Constelation (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Muñoz, P., Budnik, F., Companys, V., Godard, B., Casas, C. M., Morley, T., & Janarthanan, V. (2015). Rosetta Navigation During Lander Delivery Phase and Reconstruction of Philae Descent Trajectory. In *Proceedings 25th International Symposium on Space Flight Dynamics - 25th ISSFD.* Munich, Germany.

Ogawa, N., Terui, F., Mimasu, Y., Yoshikawa, K., Ono, G., Yasuda, S., ... Tsuda, Y. (2020). Image-based autonomous navigation of Hayabusa2 using artificial landmarks: The design and brief in-flight results of the first landing on asteroid Ryugu. *AIAA Scitech 2020 Forum*. doi: 10.2514/6.2020-0225

Olds, R. D., Miller, C. J., Norman, C. D., Mario, C. E., Berry, K., Palmer, E., ... Seabrook, J. A. (2022). The Use of Digital Terrain Models for Natural Feature Tracking at Asteroid Bennu. *The Planetary Science Journal*, *3*(5), 100. doi: 10.3847/PSJ/ac5184

Pardo de Santayana, R., & Lauer, M. (2015). Optical Measurements for Rosetta Navigation Near the Comet. In *Proceedings 25th international symposium on space flight dynamics - 25th issfd.* Munich, Germany.

Park, T. H., Bosse, J., & D'Amico, S. (2021). Robotic Testbed for Rendezvous and Optical Navigation: Multi-Source Calibration and Machine Learning Use Cases. *2021 AAS/AIAA Astrodynamics Specialist Conference*. (AAS 21-654) doi: 10.48550/arXiv.2108.05529

Park, T. H., Martens, M., Lecuyer, G., Izzo, D., & D'Amico, S. (2021). SPEED+: Next Generation Dataset for Spacecraft Pose Estimation across Domain Gap. *2022 IEEE Aerospace Conference (AERO)*. doi: 10.1109/AERO53065.2022.9843439

Preusker, F., Scholten, F., Matz, K.-D., Roatsch, T., Willner, K., Hviid, S. F., ... Vincent, J.-B. (2015). Shape model, reference system definition, and cartographic mapping standards for comet 67P/Churyumov–Gerasimenko stereo-photogrammetric analysis of Rosetta/OSIRIS image data. *Astronomy & Astrophysics*, *583*. (A33) doi: 10.1051/0004-6361/201526349

Razgus, B. (2017). *Relative navigation in asteroid missions using dual quaternion filtering* (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Reina, G. (2010). Methods for Wheel Slip and Sinkage Estimation in Mobile Robots. In *Robot Localization and Map Building* (chap. 28). Rijeka, Croatia: IntechOpen. doi: 10.5772/9279

Siddiqi, A. (2016). *Beyond earth: A chronicle of deep space exploration, 1958–2016*. Washington, DC, United States: NASA.

Sinem Ince, E., Barthelmes, F., Reißland, S., Elger, K., Foerste, C., Flechtner, F., & Schuh, H. (2019). ICGEM – 15 years of successful collection and distribution of global gravitational models, associated services, and future plans. *Earth System Science Data*, *11*, 647-674. doi: 10.5194/essd-11-647-2019

Spee, S. (2022). Asteroid gravity field estimation below the Brillouin sphere (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Sundberg, A. (2024). Very-close Range Satellite Relative Navigation Using Monocular SLAM (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Terwindt, E. (2025). Experimental navigation measurements (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Van der Heijden, L. (2022). Autonomous Navigation around Asteroids using Convolutional Neural Networks (MSc Thesis). Delft University of Technology, Delft, Netherlands.

Volpe, R., Sabatini, M., Palmerini, G., & Mora, D. (2020). Testing and Validation of an Image-Based, Pose and Shape Reconstruction Algorithm for Didymos Mission. *Aerotecnica Missili & Spazio*, *99*. doi: 10.1007/s42496-020-00034-6

Wakker, K. F. (2015). *Fundamentals of Astrodynamics*. Delft, Netherlands: TU Delft Institutional Repository. (UUID 3fc91471-8e47-4215-af43-718740e6694e)

Walsh, K. J., Bierhaus, E. B., Lauretta, D. S., Nolan, M. C., Ballouz, R.-L., Bennett, C. A., … Wolner, C. W. V. (2022). Assessing the Sampleability of Bennu's Surface for the OSIRIS-REx Asteroid Sample Return Mission. *Space Science Reviews*, *218*(4), 20. doi: 10.1007/s11214-022-00887-2

Wie, B. (1998). *Space Vehicle Dynamics and Control*. American Institute of Aeronautics and Astronautics.

Wilde, M., Choon, S. T. K., & Romano, M. (2020). Kinematic and Dynamic Spacecraft Maneuver Simulators for Verification and Validation of Space Robotic Systems. In *Rendezvous, Relative Motion, Proximity Missions, and Formation Flying III.* (AAS 20-424) doi: 10.2514/6.2020-1919

Wilde, M., Clark, C., & Romano, M. (2019). Historical survey of kinematic and dynamic spacecraft simulators for laboratory experimentation of on-orbit proximity maneuvers. *Progress in Aerospace Sciences*, *110*, 100552. doi: 10.1016/j.paerosci.2019.100552

Woicke, S. (2019). *Hazard Relative Navigation: Towards Safe Autonomous Planetary Landings in Unknown Hazardous Terrain* (Doctoral Thesis, Delft University of Technology, Delft, Netherlands). doi: 10.4233/uuid:a638c550-0d30-41df-9d49-4f935890bd2b

A

Appendix A: Command-Line for Robot and Camera Operation

This appendix summarises key ROS terminal commands used to control the RB-Kairos rover base, UR16e robotic arm, and the image acquisition usign MER2 camera.

```
screen -r arm_bringup
```

Retrieves the live session managing the robotic arm's bring-up process. Wait up to 2 minutes for initialisation. The final message must indicate that the robot is ready to receive control commands as shown in Figure A.1.

To exit: `Ctrl + A`, then `D`

```
cd  /catkin_ws
```

Navigates to the root of the Catkin workspace that stores source code and ROS packages.

```
source devel/setup.bash
```

Sources the Catkin workspace. This is mandatory in each new terminal to initialise ROS.

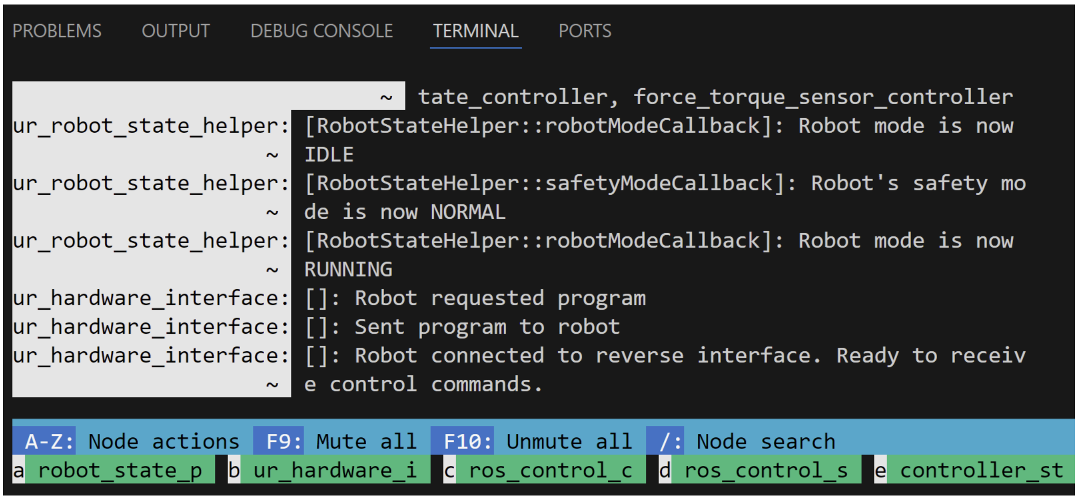
```
rosrun crina move_robot
```

Executes the motion controller that sends trajectory and pose commands to the mobile base and robotic arm based on input CSV data.

```
roslaunch galaxy_camera mer2.launch
```

Launches the Galaxy camera driver. This must be run in a separate terminal to enable image publishing.

```
rosrun crina camera
```



```
                              ~ tate_controller, force_torque_sensor_controller
ur_robot_state_helper: [RobotStateHelper::robotModeCallback]: Robot mode is now
                     ~ IDLE
ur_robot_state_helper: [RobotStateHelper::safetyModeCallback]: Robot's safety mo
                     ~ de is now NORMAL
ur_robot_state_helper: [RobotStateHelper::robotModeCallback]: Robot mode is now
                     ~ RUNNING
ur_hardware_interface: []: Robot requested program
ur_hardware_interface: []: Sent program to robot
ur_hardware_interface: []: Robot connected to reverse interface. Ready to receiv
                     ~ e control commands.

 A-Z: Node actions  F9: Mute all  F10: Unmute all  /: Node search
a robot_state_p  b ur_hardware_i  c ros_control_c  d ros_control_s  e controller_st
```

Figure A.1: Correct connection of the UR16e robotic arm

Starts the camera node responsible for image capture, triggered by a flag during motion sequences.

```
rostopic echo /galaxy_camera/galaxy_camera/image_raw
```

Displays the raw image stream from the Galaxy camera. Useful for verifying camera connectivity and data flow.

```
catkin_make -DCATKIN_WHITELIST_PACKAGES="crina"
```

Compiles only the `crina` package, avoiding unnecessary rebuilding of other packages in the workspace.

It has been observed that the robotic arm frequently enters a protective stop upon startup. In standard cases, this can be resolved remotely via SSH using the ROS service call:

```
rosservice call /ur_hardware_interface/dashboard/unlock_protective_stop
```

B

Appendix B: Installing the GNC Lab Gazebo Simulation

This appendix summarises the essential terminal commands used to install and configure the RB-Kairos simulation environment, provided by Robotnik Automation. The default simulation includes a Gazebo model of the RB-Kairos mobile base equipped with a UR5 manipulator. The configuration and launch files have been adapted to work for an UR16e

The simulation runs on **Ubuntu 20.04** with **Gazebo 9.19**, and supports ROS Noetic. While Robotnik officially documents ROS Melodic, that distribution is intended for Ubuntu 18.04. To ensure compatibility with newer systems, this guide assumes a Noetic-based setup.

```
sudo apt-get install -y python3-vcstool
```

Installs the version control tool used to clone multiple repositories via a ‘.repos‘ file.

Option 1: Simulation Environment from Robotnik Automation

```
mkdir -p  /catkin_ws/src
cd  /catkin_ws
vcs import –input https://raw.githubusercontent.com/RobotnikAutomation/rbkairos_sim/noetic-devel/repos/rbkairos_sim.repos
rosdep install –from-paths src –ignore-src -y -r
```

Clones the simulation repositories and installs all dependencies using `rosdep` directly from the manufacturer.

Option 2: Simulation Environment GNC Laboratory

```
git clone https://github.com/fcmihalache/Msc_Thesis_Crina_Mihalache.git
mkdir -p  /catkin_ws/src
cd  /catkin_ws
cp -r Msc_Thesis_Crina_Mihalache/src/*  /catkin_ws/src/
rosdep install –from-paths src –ignore-src -y -r
```

Clones the simulation repositories and installs all dependencies using `rosdep` directly with the adjustments made to match the GNC laboratory. Access may have to be requested.

```
catkin build
source devel/setup.bash
```

Builds the Catkin workspace and sources the environment for the current session.

```
roslaunch rbkairos_sim_bringup rbkairos_complete.launch
```

Launches the default simulation environment as provided by Robotnik. This serves as a baseline test to verify correct installation of all required packages and components.

For GNC Lab-specific simulations, modified launch files and configuration packages are maintained in a dedicated repository. After cloning the appropriate package:

```
roslaunch rbkairos_sim_bringup gnc_lab.launch
```

Launches the customised simulation environment tailored for experiments in the GNC Lab.

```
cd /catkin_ws/src/inspection
rosrun inspection move_robot
```

Move the robotic system using input files.

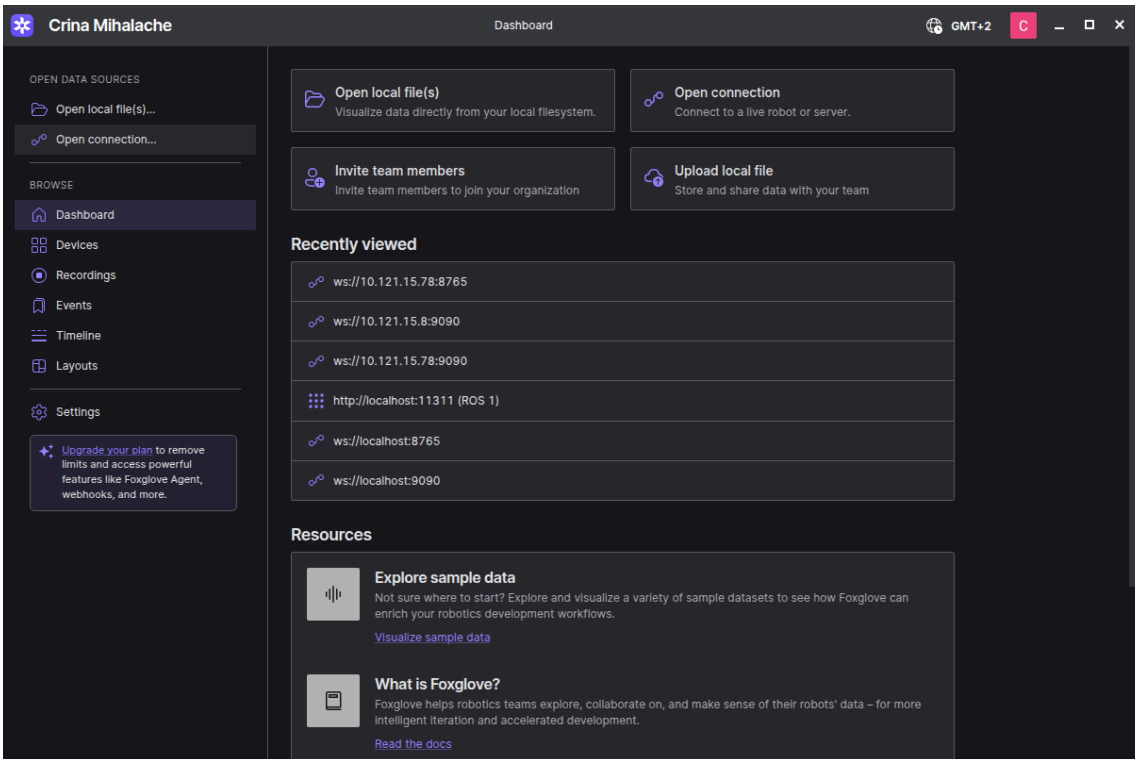
C

Appendix C: Human Robot Interface

The Foxglove studio window is showcased in Figure C.1. In order to ensure a connection between the simulated robot and Foxglove, before continuing any further, the steps presented in Appendix B to launch the simulation are required. Once the Gazebo simulation has initialized, it is possible to establish a connection via Open connection (Connect to a live robot or server). The connection parameters are shown in Figure C.2. A wide range of panels and commands can be established to customise the HRI. The selection for the HRI of the GNC lab is shown in Figure C.3

Terminal command to launch Foxglove studio.

```
foxglove-studio
```



Upgrade your plan to remove limits and access powerful features like Foxglove Agent, webhooks, and more.

Visualize data directly from your local filesystem.

Connect to a live robot or server.

Invite team members to join your organization

Store and share data with your team

ws://10.121.15.78:8765

ws://10.121.15.8:9090

ws://10.121.15.78:9090

http://localhost:11311 (ROS 1)

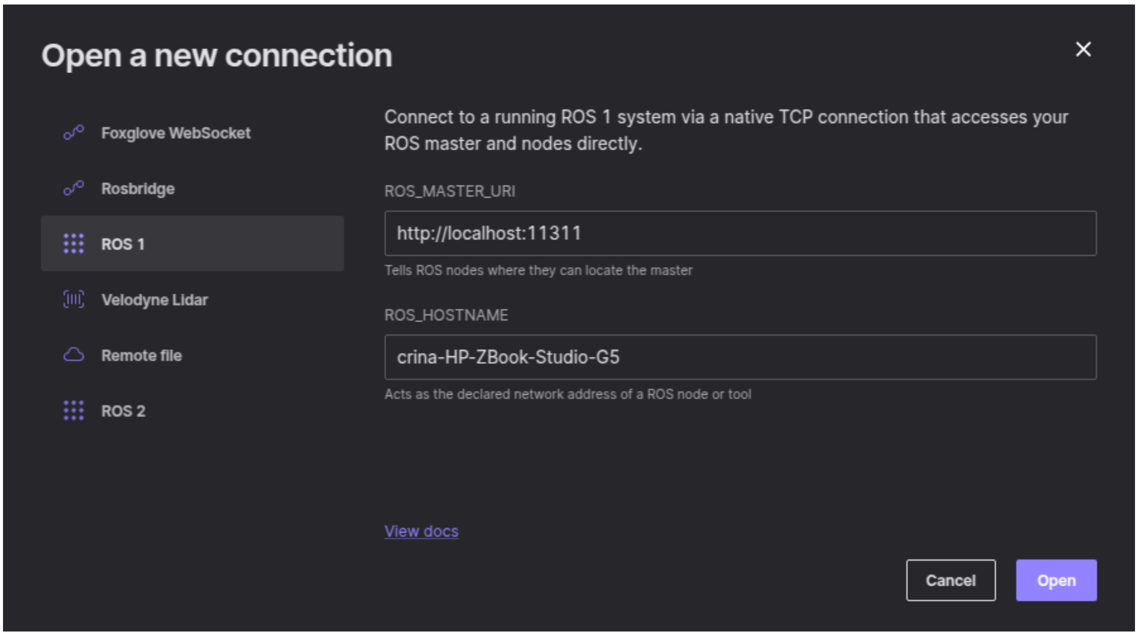
ws://localhost:8765

ws://localhost:9090

Not sure where to start? Explore and visualize a variety of sample datasets to see how Foxglove can enrich your robotics development workflows.

Foxglove helps robotics teams explore, collaborate on, and make sense of their robots' data – for more intelligent iteration and accelerated development.

Figure C.1: Launch Foxglove Studio



Open a new connection

Connect to a running ROS 1 system via a native TCP connection that accesses your ROS master and nodes directly.

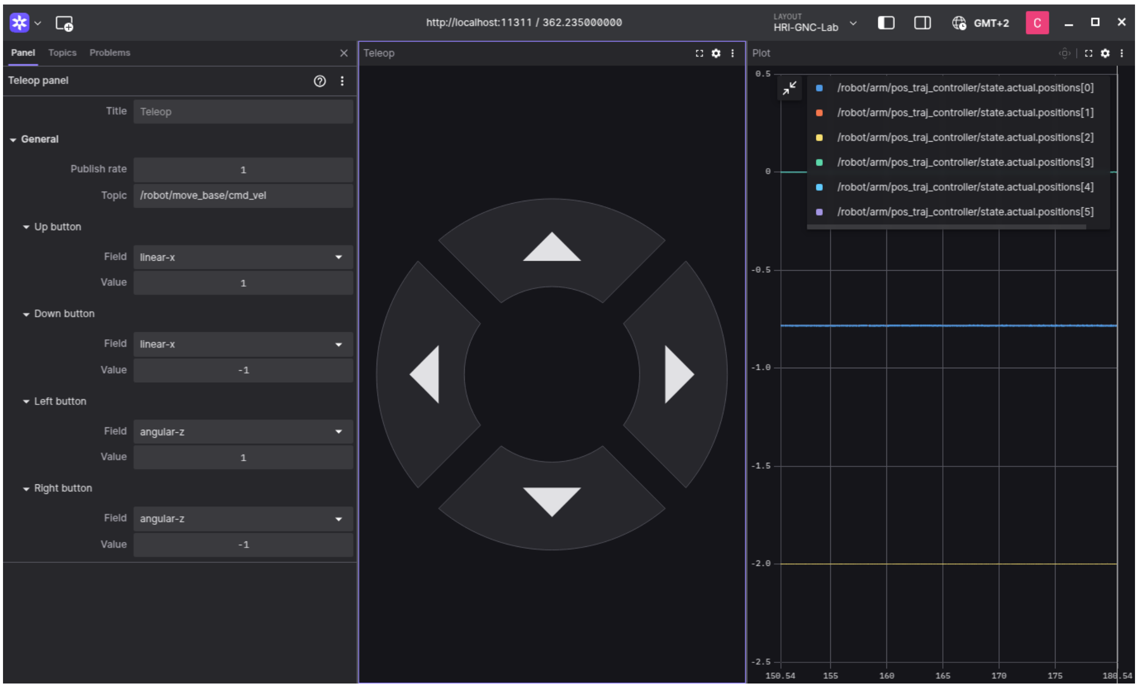
ROS_MASTER_URI: http://localhost:11311

Tells ROS nodes where they can locate the master

ROS_HOSTNAME: crina-HP-ZBook-Studio-G5

Acts as the declared network address of a ROS node or tool

Figure C.2: Establishing a foxglove ROS1 connection with the Gazebo simulation of the robotic system of the GNC Lab



http://localhost:11311 / 362.235000000

LAYOUT: HRI-GNC-Lab

GMT+2

Panel
Topics
Problems
Teleop
Plot

Teleop panel

Title: Teleop

General

Publish rate: 1

Topic: /robot/move_base/cmd_vel

Up button

Field: linear-x

Value: 1

Down button

Field: linear-x

Value: -1

Left button

Field: angular-z

Value: 1

Right button

Field: angular-z

Value: -1

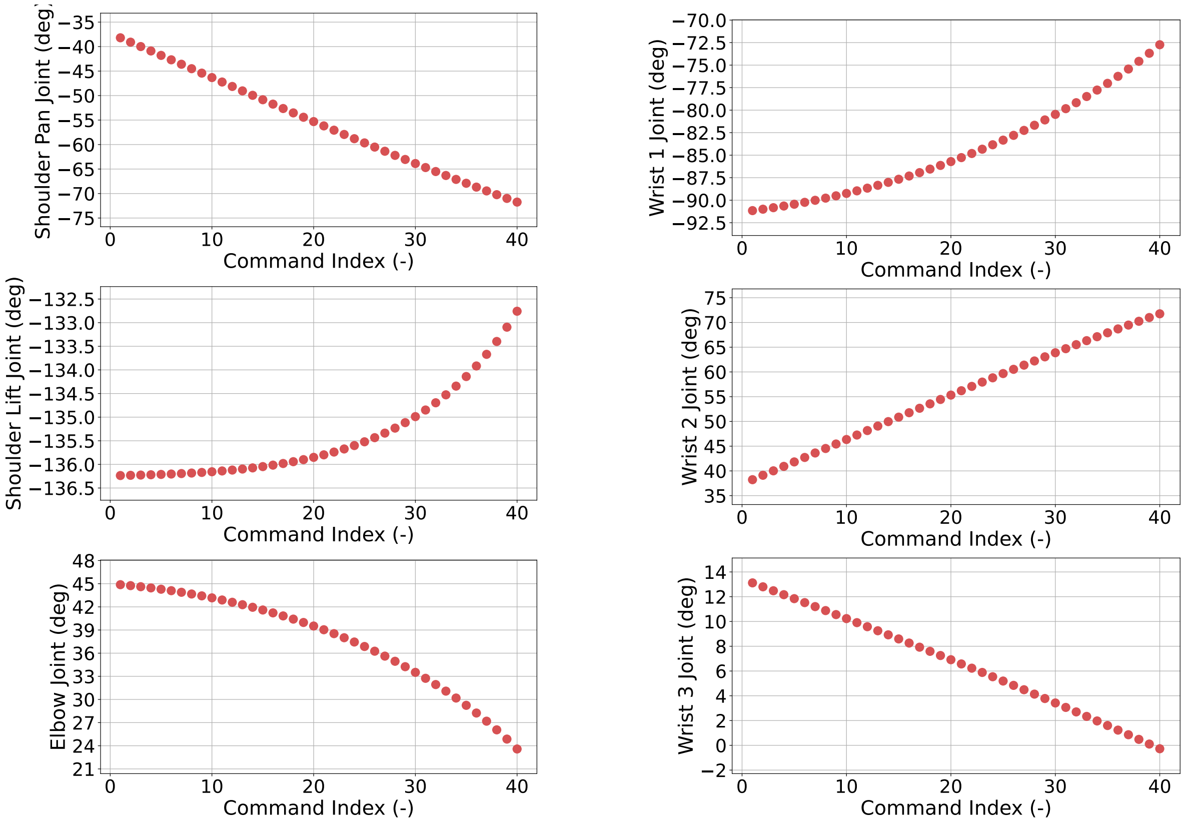
/robot/arm/pos_traj_controller/state.actual.positions[0]
/robot/arm/pos_traj_controller/state.actual.positions[1]
/robot/arm/pos_traj_controller/state.actual.positions[2]
/robot/arm/pos_traj_controller/state.actual.positions[3]
/robot/arm/pos_traj_controller/state.actual.positions[4]
/robot/arm/pos_traj_controller/state.actual.positions[5]
0.5
0
-0.5
-1.0
-1.5
-2.0
-2.5
150.54
155
160
165
170
175
180.54

Figure C.3: Foxglove Panels for the HRI of the GNC Lab

D

Appendix D: Joint Commands Visualisation

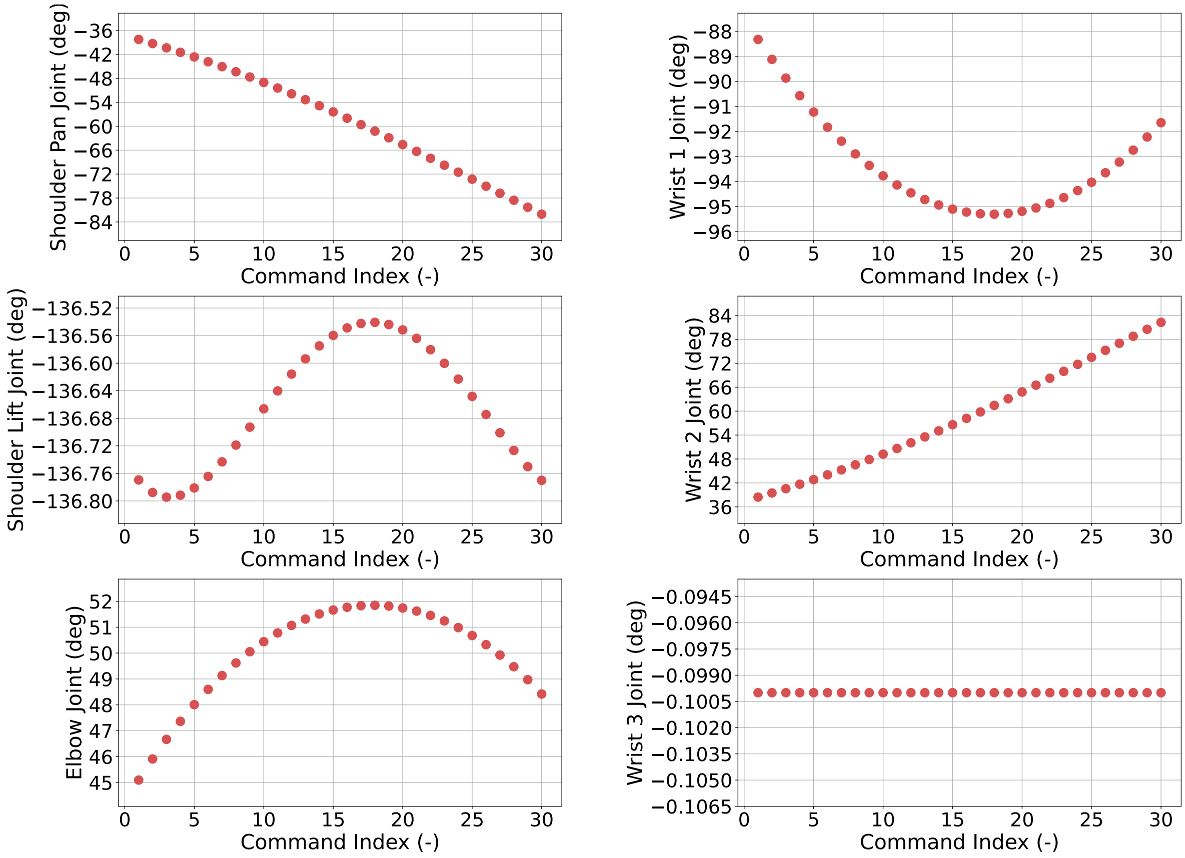
This appendix presents joint angle trajectories computed via inverse kinematics for predefined end-effector paths. These were executed using the Static Base, Mobile Arm configuration (Section 5.4), where the base remains fixed and the arm tracks waypoints. Joint commands are shown for each test case, split into two parts for clarity. Tests involve linear and curved motions in different planes, labelled by path direction. The profiles (Fig. D.1–D.5) illustrate the smoothness, continuity, and feasibility of arm motion in this setup.



(a) Part 1

(b) Part 2

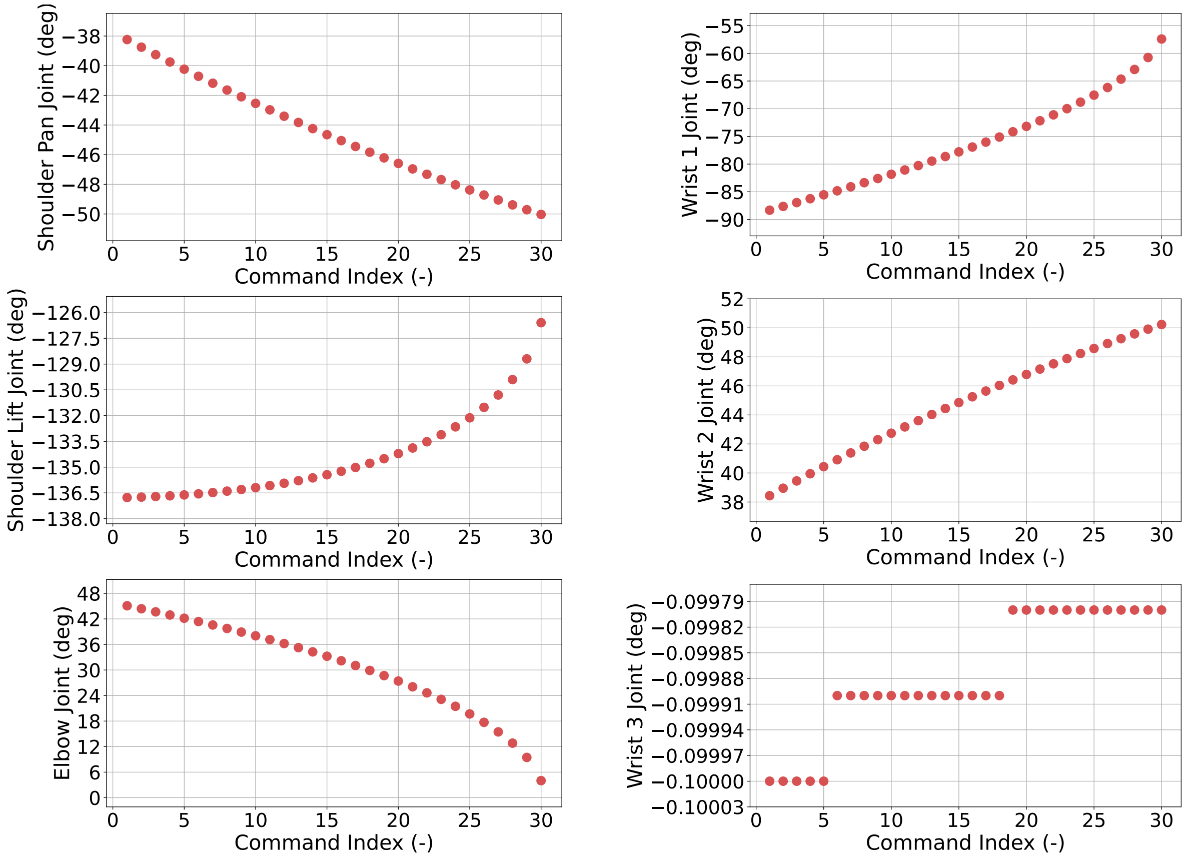
Figure D.1: Joint commands for test C2_C1_track_target_line



(a) Part 1

(b) Part 2

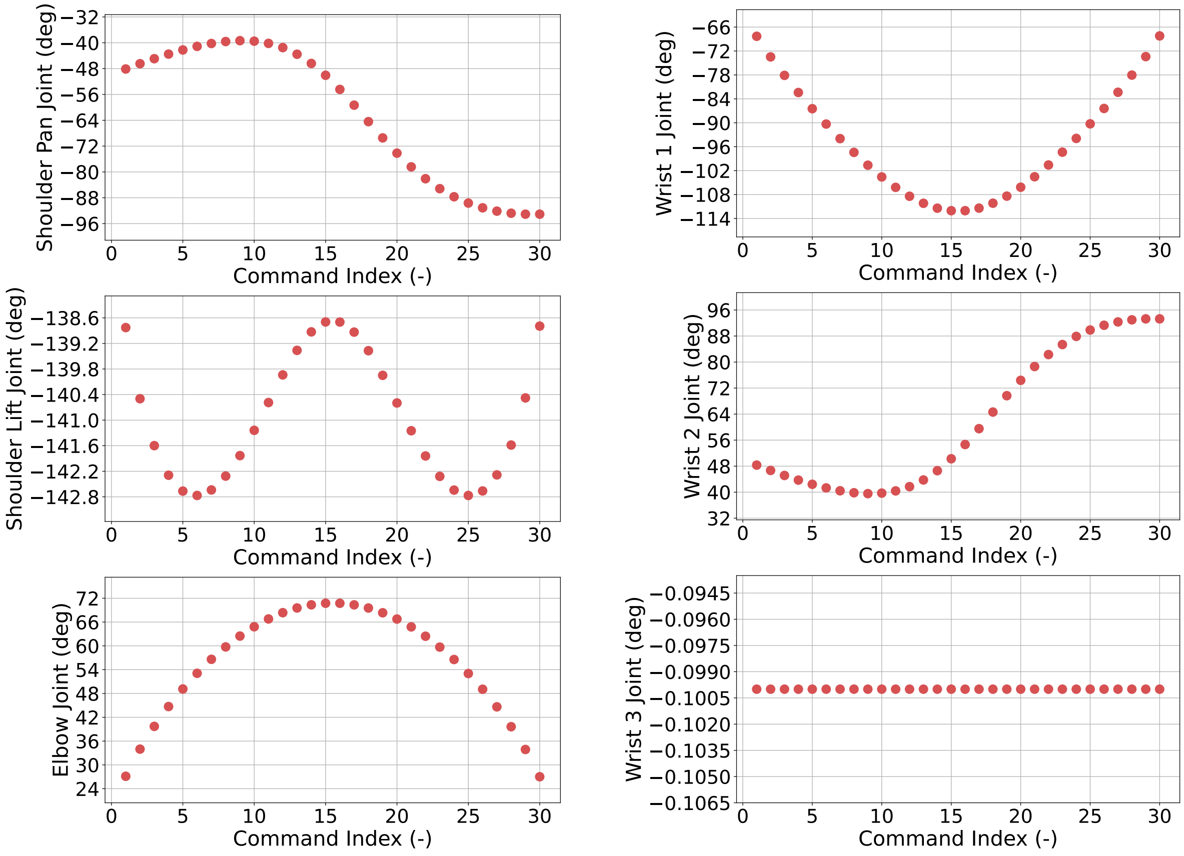
Figure D.2: Joint commands for test C2_A1_line_x



(a) Part 1

(b) Part 2

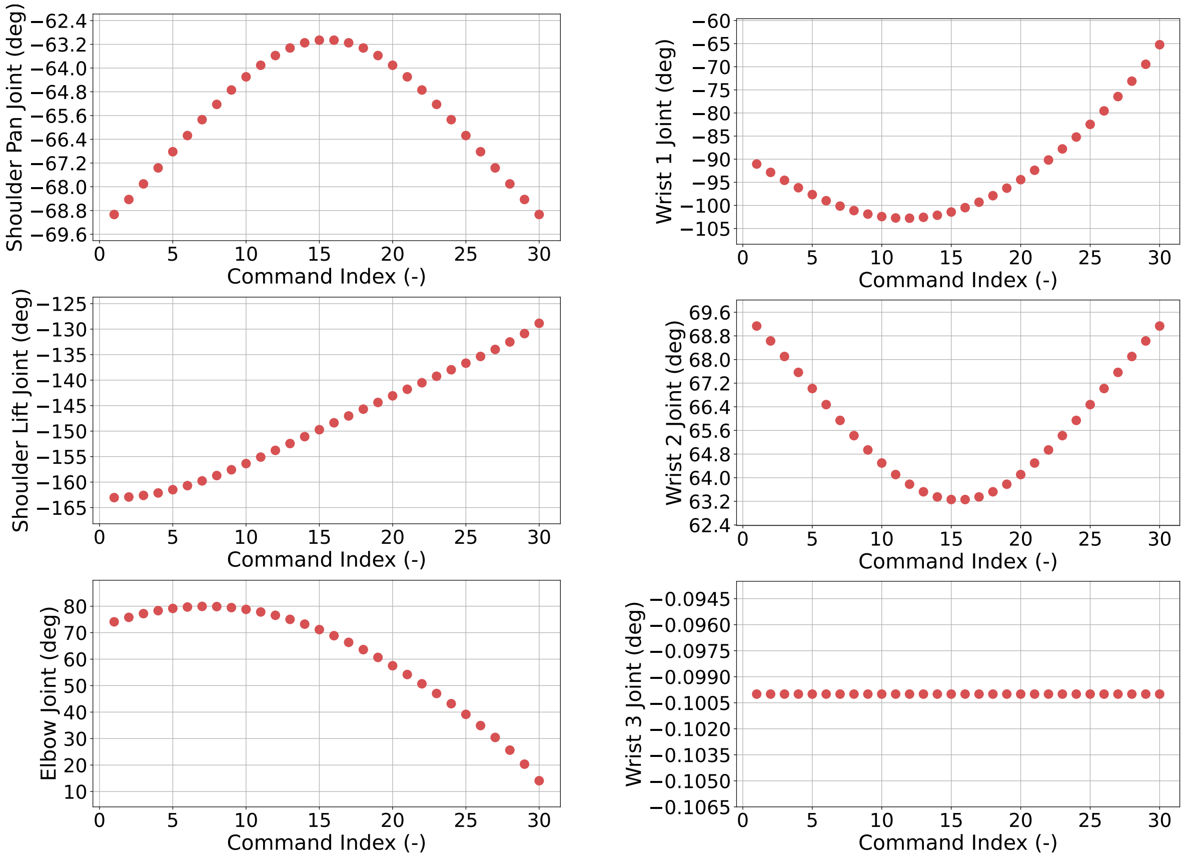
Figure D.3: Joint commands for test C2_A2_line_y



(a) Part 1

(b) Part 2

Figure D.4: Joint commands for test C2_B3_half_circle_xy



(a) Part 1

(b) Part 2

Figure D.5: Joint commands for test C2_B4_half_circle_yz