Autonomous Rendezvous, Docking, and Berthing in Space

By

Sachit Ravikumar

MSc Robotics Dissertation

Department of Engineering Mathematics

UNIVERSITY OF BRISTOL

&

Department of Engineering Design and Mathematics

UNIVERSITY OF THE WEST OF ENGLAND

A MSc dissertation submitted to the University of Bristol and the University of the West of England in accordance with the requirements of the degree of MASTER OF SCIENCE IN ROBOTICS in the Faculty of Engineering.

Declaration of own work

I declare that the work in this MSc dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

Sachit Ravikumar

05/09/2024

Acknowledgement

I would like to express my sincere gratitude to my Supervisor, Dr. Hamidreza Nemati, for his invaluable support, time, guidance, and mentorship throughout this project. I also extend my thanks to everyone who assisted me, even with the smallest challenges. This project would not have been possible without their collective support.

Abstract

The development of autonomous systems for rendezvous, docking, and berthing in space is essential for developing space mission capabilities, particularly in solving challenges like docking operations and space debris maintenance. This research explores the design, implementation, and evaluation of a completely autonomous docking system that utilizes advanced control algorithms and integrated sensor technologies in a high-fidelity simulation environment. The work uses Gazebo and the Robot Operating System (ROS) to replicate space conditions, with Velodyne LiDAR and depth cameras added to improve spatial awareness. Three control systems are used and evaluated for their effectiveness in moving spacecraft during docking operations: proportional-integral-derivative (PID), linear quadratic regulator (LQR), and non-linear control. Optimizing Hill-Clohessy-Wiltshire (HCW) equations increases trajectory planning accuracy. The system's performance is examined under a variety of conditions, including debris avoidance, docking with orientation changes and docking on different planes, to prove its durability and accuracy. The findings develop autonomous space technology and provide better methods for future space missions.

Number of words in the dissertation: 12684 words.

(Obtained from Overleaf)

Contents

List of Tables

1 Introduction

1.1 Motivation

The motivation for this project is deeply rooted in the urgent need to overcome the current constraints faced by space missions, which are critical for the growth and sustainability of human existence in space and exploring into unknown regions. As we expand the boundaries of space exploration, we face significant obstacles such as the need for removing space debris, improving the efficiency of maintaining satellites in orbit, and dealing with the complexities of assembling structures in space. These challenges are not just technological obstacles; they stand in the way of the long-term sustainability of human presence in space. Addressing these issues involves the development of innovative navigational models and advanced autonomous systems capable of operating consistently in the harsh and unpredictable environments of space. [1]

The existing manual and semi-autonomous docking processes require a lot of resources and can be risky in time-critical scenarios. Operational complexity is constrained and possible errors are introduced when human operators use the system. Developing fully autonomous systems could reduce costs, improve mission flexibility, and enable more complex space activities. This shift toward autonomy is a necessary evolution in space technology that enables ambitious future missions while maintaining a sustainable space environment. [2]

As space becomes more congested with active satellites and debris, accurate autonomous movements become essential to avoiding collisions while remaining safely in orbit. The growing concern about space debris threatens the long-term viability of satellite operations and human spaceflight[3]. Autonomous systems capable of identifying, tracking, and potentially removing debris could play an important role in reducing this risk.

Inspired by the achievements of terrestrial autonomous systems, a viable solution to these problems is the combination of sophisticated control algorithms with innovative sensing technologies[2]. In the space environment, these technologies offer more precise and dependable object identification and relative navigation.

This project aims to push the boundaries of autonomous space operations by developing and evaluating a comprehensive autonomous rendezvous and docking system using Gazebo and ROS(Robot Operating System). Motivated by the latest developments in space robotics, we integrate innovative sensing technologies with advanced control techniques to overcome the existing limitations in space exploration. Future space missions will be safer, more accurate, and more efficient due to the extensive testing and validation of autonomous capabili-

ties that can be done in these simulation settings before they are deployed in the real world.

Our research is driven by the need to address key problems in space exploration, such as **active debris removal, effective on-orbit servicing, and the complexities of in-space assembly.** By focusing on these areas, we hope to aid in the development of spacecraft systems that are more capable, resilient, and flexible, which will ultimately make space more accessible and navigable for future space generations of space explorers.

1.2 Aims

This project aims to develop and evaluate an advanced autonomous system for spacecraft operations. The primary aims are:

1. To develop and implement a precise simulation environment for testing autonomous space operations, using **Gazebo** and **ROS**.
2. To create an integrated sensing system that combines **Velodyne LiDAR**(Light Detection and Ranging)[4] [5] and **depth camera** technologies for precise spatial environment perception and item detection and confirmation of operations.
3. To implement and analyze the effectiveness of **PID**(Proportional–integral–derivative), **LQR**(Linear–quadratic regulator), and **nonlinear control methods** for precise spacecraft maneuvering during rendezvous and docking operations.
4. To improve the application of **Hill-Clohessy-Wiltshire (HCW) equations**[6] for accurate relative orbital motion prediction and trajectory planning.
5. To evaluate the robustness and effectiveness of the developed autonomous system,focusing on metrics such as adaptation to environmental changes, success rate, and maneuver precision across a range of simulated space scenarios, such as **debris avoidance**[7] and docking with difficult targets.
6. To contribute to the advancement of autonomous space technologies by offering better methods.

By combining enhanced sensing, precise control, and intelligent decision-making systems, these goals aim to improve the capabilities, safety, and dependability of next space missions while addressing significant challenges in autonomous space operations.

1.3 Objectives

1.3.1 Develop a High-Fidelity Simulation Environment

The objective is to utilize Gazebo and ROS to set up a realistic simulation environment that replicates space conditions such as orbital dynamics and environmental factors. This includes employing Gazebo for 3D mod-

eling and ROS for integrating robotic components. This objective supports the development and testing of autonomous space operations in a controlled and measured environment, ensuring that the technologies can be evaluated under precise situations that simulate real-world scenarios. Realistic physics models will recreate gravity forces, orbital paths, and space debris.

1.3.2 Integrate Advanced Sensing Technologies

The objective is to offer an integrated sensor suite for the chaser satellite which involves Velodyne LiDAR and depth cameras. The procedure includes selecting and designing models appropriate for space applications, as well as creating sensor fusion algorithms to read data from the sensor. This Objective directly improves the system's spatial environment perception abilities, which are critical for precision docking and operations, in accordance with the project's aims of integrating modern sensing technologies.

1.3.3 Implement and Evaluate Control Strategies

The goal is to implement and evaluate PID, LQR, and nonlinear controllers for spacecraft maneuvering. Control algorithms for each approach will be developed and implemented in the simulation. A series of tests will evaluate the controllers' precision, stability, and response time during docking maneuvers. This objective makes it easier to evaluate multiple control systems and determine the most effective way for spacecraft maneuvering, which directly impacts the project's aim of improving control accuracy and reliability.

1.3.4 Optimize Hill-Clohessy-Wiltshire (HCW) Equations

The objective is to enhance the accuracy of relative motion prediction using HCW equations for trajectory planning. By optimizing these equations, this objective helps achieve the aim of improving trajectory planning and motion prediction, which are critical for accurate maneuvering and position of spacecraft in orbit. The optimized model will be validated by comparing predicted trajectories to simulation results.

1.3.5 Test System Robustness and Efficiency

The goal is to evaluate the autonomous system's performance in situations that include debris avoidance and ineffective target docking, using metrics such as success rate and docking accuracy. This Objective directly examines how effectively the system achieves the aim by testing it's effectiveness and robustness using measured success rates and accuracy in high-stress docking scenarios.

1.3.6 Propose Improved Methodologies for Space Operations

This objective involves developing recommendations that will enhance autonomous rendezvous, docking, and berthing operations. This objective is to convert empirical data into practical changes, contributing to the project's aim of enhancing autonomous space technology by suggesting new methodologies and best practices.

1.4 Scope of the Research

Included in Scope: The Research will focus on creating and evaluating a high-fidelity simulation environment using Gazebo and ROS to assess autonomous docking options. It will combine and test novel sensors including Velodyne LiDAR and depth cameras to improve spatial awareness and identify objects. Various control systems, including as PID, LQR, and nonlinear controllers, will be used to evaluate their usefulness in maneuver accuracy and dependability. HCW equations will be used to optimize trajectory planning and increase orbital motion prediction precision. The research will also assess performance using metrics such as adaptation to environmental changes, success rate, and movement precision, and it will recommend improvements based on the results of these simulation tests.

Excluded in Scope: The scope of the research excludes real hardware installation and testing on actual spacecraft, limiting its application to simulated environments exclusively. It will not include testing in real-space environments or long-term durability testing of devices in space. The research will also not include extensive testing of all possible space environmental variables, such as sun radiation and high temperatures, nor will it look at inter-satellite communications technologies. This difference keeps the study focused on specific characteristics of autonomous system development for space applications in a simulated environment.

2 Literature Review

Autonomous docking systems are at the forefront of modern engineering challenges, especially in the fields of space, maritime, and unmanned aerial vehicles (UAVs). The rising complexity of operational environments, as well as the precision required in such applications, demands innovative approaches. This review examines the most recent research contributions, identifying common trends, limitations, and potential improvements that will influence the future of autonomous docking systems.

2.1 Advanced Control Systems for Spacecraft Docking

The field of spacecraft docking has evolved significantly with the incorporation of advanced control systems designed to enhance the accuracy and reliability of these operations. Each contribution improves prior technologies, collectively pushing the limits of what is possible in space maneuvers.

Abdollahzadeh[8] proposes sophisticated translational and rotational control systems tailored for docking operations. Specifically engineered to navigate the complexities of docking with tumbling targets amid external disturbances, it tackles intricate challenges head-on. Utilizing two sliding mode controls, it demonstrates the potential for achieving high precision and robustness in docking maneuvers through simulation-based validations. This underscores the importance of adaptive and resilient control systems in modern space operations. However, it identifies significant knowledge gaps, including challenges with complexity, computational demand, under real-space conditions and the unaddressed challenges related to maintenance and space debris - Our research addresses this gap by simulation in comprehensive environment that integrates real space dynamics, to evaluate their performance under conditions closer to real space operations.

Similarly, **Li et al.**[9] applied Model predictive control (MPC) to autonomously dock a chaser spacecraft with a tumbling space target. The study underscores MPC's capacity to manage constraints such as control input saturation and collision avoidance while aiming to reduce fuel consumption.Despite its strengths, the study's major reliance on simulations to justify the control technique is a disadvantage, raising questions about the practical application of such algorithms in actual space conditions - Our research seeks to build this gap by combining algorithms with sensor data, enhancing practical applicability of this operation.

Howard et al.[10] developed the Next Generation Advanced Video Guidance Sensor (NGAVGS) at NASA's Marshall Space Flight Center, which improves docking precision through radiation-tolerant components and extended range capabilities. The NGAVGS is essential for missions that include the ISS and future Constellation vehicles. While the sensor represents a significant technological advancement, this documentation does not go into detail on potential scalability issues or the technology's adaptability to various spacecraft designs and mission types, where our research explores on the integration of sensor with control systems.

Hinkel et al.[11] underlined the need to reduce development costs by standardizing sensor suites across multiple missions. To improve situational awareness, they suggested combining optical and LiDAR sensors. Although the idea of a uniform sensor suite is new, it may not be feasible to integrate it across a range of mission profiles without compromising some mission requirements, where this could provide significant challenges. Further clarification of the trade-offs between standardization and mission-specific customization would be helpful. In our research, this balance is assessed ; in particular, it is examined how different mission goals can be met by optimizing standardized sensor inputs using adaptive control algorithms.

An Autonomous Surface Vehicles (ASVs) using symbol recognition is developed by **Kim et al**[12] to enhance the docking accuracy without relying on GNSS data. Field experiments and tank tests were used to verify the system's performance, showing its usefulness. Although the study shows promise for autonomous docking, it also emphasizes the computational burden of real-time image processing and limitations in the accuracy of symbol recognition under various circumstances. Our research evaluates this limitations, by the use of sensor fusion algorithms, that can adapt to its environment.

The study made by **Stensina el al.**[13] tackles CubeSat docking issues utilizing a dual control strategy, which includes Model Predictive Control (MPC) for trajectory management and Sliding Mode Control (SMC) for attitude stabilization. Simulations show how to achieve docking precision requirements with effective control under a variety of situations. the only dependence on simulations shows a lack of knowledge regarding the practicality of these controls. This is expanded by our research, which tests these control strategies in a range of simulated settings that more closely resemble real-space situations in an effort to confirm their effectiveness in a variety of scenarios.

In summary, advancements in spacecraft docking technologies reveal significant knowledge gaps. The frequent reliance on simulations underlines a gap between theoretical models and actual implementation, particularly under real space environments. Problems such as as system complexity, computing demands, and adaptability across varied mission profiles remain challenges. Addressing these gaps through empirical research and real-world testing or setting up space environment, is essential for the realistic execution and reliability of future docking procedures. Reliable sensor integration significantly improves the performance of these complex control systems, as precise control during complicated docking maneuvers requires accurate real-time input from sensors like as LiDAR and deep learning models.

2.2 Sensor Integration and Data Fusion in Docking Operations

Advanced sensor integration and reliable data fusion techniques are essential for the successful execution of spacecraft docking operations. These components are necessary for navigating the challenging environment of space, where mission success is strongly impacted by attitude and position prediction accuracy. To improve these estimations, recent studies have used a variety of techniques, such as deep learning algorithms and

statistical inference models.

Barr el at.[14]investigate the use of Generalized Covariance Intersection (GCI) and Probability Hypothesis Density (PHD) filters in multi-sensor data fusion for the purpose of detecting tiny marine craft in crowded situations. Despite having a maritime focus, their method provides insightful information about managing intermittent observations and non-linear dynamics that are relevant to orbital operations. The potential for space application of PHD filters for managing clutter and false alarms is evident, but validation under space-specific disturbances such as cosmic radiation and microgravity is necessary for adaption to space environments, where our research addresses this gap by validating the performance under space disturbances like microgravity.

Phisannupawong et al.[15] implements deep learning via convolutional neural networks, particularly GoogLeNet, is utilized for precise spacecraft position and orientation estimation during docking, using Unreal Engine 4 for image processing simulation.They provide a cost-effective approach for training models, which could reduce the need for expensive real-world data collecting. However, the model's practical application may be affected by their method's dependence on high-quality simulated data, which could fail to properly represent the unpredictable variables encountered in real space environments, such as varying light conditions and cosmic interference. Our research evaluates this limitation by testing the docking operation under range of scenarios with real space environments - with low light condition and space debris.

The automated aerial docking system developed by **Choi et al.**[16] uses deep learning for real-time pose estimation from camera imagery. Their vision-based approach shows the potential for space docking applications; yet, its adaptation to space requires tackling the significant changes in environment and external disturbances, which raises questions over its scalability in zero-gravity settings. Our research explores how these algorithms are adapted to space settings.

Another significant contribution made by **Liu et al.** [17] is the use of deep learning for precise pose estimation, which is required for autonomous spacecraft docking. Although their method is novel, it underscores the complexity of training models that can adjust to the various space conditions, such as shifting targets and varying lighting. The accuracy and reliability of these models in actual space circumstances remain unresolved. To improve the reliability of position estimate models, our study expands on this by using adaptive learning algorithms that adapt to changing situations in space.

A study by **Jia et al.**[18] developed a LiDAR-based system for autonomous docking and recharging of mobile robots in manufacturing, using deep learning to accurately localize charging stations. The combination of LiDAR and deep learning allows highly accurate object detection and docking alignment. However, the system's reliance on specific environmental settings limits its adaptability and generalizability across numerous operational settings. Our research expand this limitations by testing similar systems in varying environment, with the goal to enhance these systems' ability to adapt to various operating environments.

Volden et al.[19] designed a low-cost stereo-vision positioning system for unmanned surface vehicles (USVs) that optimizes docking precision. They tackled detecting problems in dynamic outside contexts by merging traditional computer vision with modern technologies. Despite its success in field tests, the system's

integration and calibration remain difficult, particularly under changing climatic circumstances that affect sensor accuracy and data fusion. Our research takes similar approach as this study, by focusing on sensor adaption and its intergration in space settings, where conditions are robust.

Christensen's thesis[20] investigates multi-sensor data fusion for spacecraft navigation in non-cooperative circumstances, with a focus on vision-based approaches. It tracks small probes distributed before to a spacecraft's flyby, utilizing optical and radiometric data to increase mass estimation accuracy. While the concept is promising, synchronizing multiple data streams as well as accurate sensor calibration pose significant challenges. Our research tackles this by exploring real-time data processing approaches and synchronization strategies that can improve the accuracy and reliability of sensor fusion in space.

Studies such as those by **Fahey et al.**[21] and **John & Scott** [22] highlight that one advantage of using LiDAR in space is the absence of an atmosphere. Without air molecules to scatter or absorb the LiDAR laser light, the signal could potentially travel much farther than it would on Earth. Our research builds on this principle by using LiDAR technology to take advantage of these longer-range capabilities.

In summary, while the studies reviewed demonstrate progress in sensor integration and data fusion for spacecraft docking, they also highlight important limitations in adapting these technologies to the extreme environment of space. Additional research is required to improve system robustness and accuracy under space-specific disturbances, ensuring reliable operation in real-world conditions. As we look at the difficulties encountered by marine and aerial docking systems, it is evident that each environment has its unique set of challenges.

2.3 Challenges in Maritime and Aerial Docking Systems

Autonomous docking systems for marine and aerial vehicles encounter numerous obstacles due to the highly dynamic and unpredictable situations in which they operate. Recent research initiatives have shown that addressing these difficulties involves the combination of modern technology and strategies.

Park and Kim [23] use reachability analysis to tackle the unpredictability of marine factors such as currents and winds, which impact unmanned surface vehicle (USV) docking. By combining Hamilton-Jacobi partial differential equations, they give a solid foundation that ensures safety and optimality in the face of disruptions. While this strategy efficiently allows for a variety of future states and uncertainties, its computational complexity and reliance on correct environmental data may restrict its practical application. Our research follows similar strategy, where computational complexity is essential, aims to optimise these methods in real-time. **Lee et al.**[24] utilize vision and 2D LiDAR technology to increase docking accuracy for self-driving vehicles, addressing issues such as fluctuating lighting and weather conditions. While this strategy improves sensor dependability in complex situations, it relies significantly on high-quality sensor data, which can be difficult to get in harsh maritime conditions. In addition, adding more sensors complicates the system and requires the

use of sophisticated data fusion algorithms to successfully combine various data streams. Our research extends this by introducing sensor fusion algorithms that may compensate for data quality concerns, increasing overall system robustness. **Kim et al**.[12] developed an autonomous docking system for surface vehicles (ASVs) that uses symbol recognition to guide the process. The system divides docking into two phases: approaching and docking, with symbol recognition allowing GNSS-free navigation. However, relying on visual signals and sensor accuracy presents difficulties in low visibility or adverse weather conditions frequent in maritime or space conditions. Our research tackles these limitations by looking at new sensor technologies that are more resistant to environment.

Miyazaki et al. [25] present an innovative approach to multi-rotor UAV down wash by using a winch mechanism that minimizes unstable rotor airflow, resulting in more stable docking. However, this increases the weight and complexity of the UAV, potentially affecting flight efficiency and maneuverability. Balancing mechanical innovation with flight performance is critical, and further research is needed to improve stability and efficiency in aerial docking. Our Research builds on this, by investigating lightweight stabilizing methods that maintain mobility and be included into spacecraft docking systems.

The papers reviewed show significant gaps in autonomous docking systems, particularly in sensor reliability under extreme conditions and the balance between mechanical innovation and system economy. Future research should focus on strong, adaptive technology that can function successfully in dynamic marine and airborne environments, assuring safe and practical applications. While addressing the issues of docking in dynamic marine and aerial environments requires complex control systems, reinforcement learning presents a viable way to improving the flexibility and decision-making skills of autonomous systems in such unexpected situations.

2.4 Reinforcement Learning

Reinforcement learning (RL) has emerged as a transformational paradigm in autonomous systems, providing novel answers to the dynamic and unpredictable nature of space operations.

Athauda et al. [26] employ the TD3 algorithm for intelligent motion planning in satellite docking, with a focus on collision avoidance in debris-dense settings through a robotic emulation platform. While many RL applications perform well in simulations, transferring them to real-world space operations presents considerable problems because to their complexity and high processing needs. **Anderlini et al.** [27] apply Deep Deterministic Policy Gradient (DDPG) and Deep Q-Network (DQN) to simulate AUV docking, discovering that DDPG has chattering issues whereas DQN docks efficiently with lower computing requirements. Despite successful simulations, applying these RL approaches in real-world applications brings hurdles, such as dealing with changing conditions scaling across multiple AUV models and situations.

Despite the advances pointed out by above studies, significant gaps in knowledge remain, particularly re-

garding the safety of Reinforcement Learning applications in space. The vital importance of space operations illustrates how even small errors may threaten mission safety, raising worries about the dependability and robustness of RL when implemented in such high-risk situations. In our research, we create and implement our own control algorithms that are specifically designed to optimize autonomous spacecraft docking and collision avoidance, addressing computational complexity challenges, ensuring robustness, and effectively closing the gap between simulation and real-world application in high-risk space environments. As reinforcement learning extends the frontiers of autonomous system capabilities, it becomes more important to combine these techniques with collision avoidance algorithms to ensure that these systems can safely navigate complex and hectic environments.

2.5 Autonomous Collision Avoidance

Autonomous collision avoidance systems are essential for ensuring the safety and effectiveness of spacecraft during rendezvous and docking operations. These systems are designed to detect possible collisions and execute actions to avoid them, which is crucial in highly populated orbits and during complex docking operations.

Scharnag et al. [28] built a technology, that utilizes Photonic Mixer Devices (PMDs) to improve real-time trajectory prediction and collision detection during space operations. PMDs' capacity to deliver exact 3D imagery and posture estimation guarantees reliable obstacle identification and avoidance, which is critical for mission safety in a variety of space operation situations. However, the practical deployment of PMD-based systems in space requires additional validation under the varying conditions encountered in different orbits and missions. **Zappulla el al.** [29] introduced Adaptive Artificial Potential Field (AAPF) Method, which improves on existing artificial potential function systems used in spacecraft navigation by introducing adaptability to potential fields based on real-time environmental changes. This adaptability is critical for establishing complicated situations, considerably increasing efficiency and safety during space maneuvers. However, while AAPF improves maneuver efficiency, its complexity could lead to computing issues and integration with modern spacecraft systems. **Wang et al.** [30] explore the use of an upgraded deep reinforcement learning algorithm, the Deep Q-Network with a Faster R-CNN model and a Data Deposit Mechanism (FRDDM-DQN), for UAV autonomy and collision avoidance in unfamiliar situations. Their solution outperforms standard models in simulation environments, but it underlines the issues of reducing training time and retraining when environmental conditions change. This study emphasizes the importance of building more adaptive, efficient, and faster training algorithms to deal with dynamic and complex operational settings.

Significant progress has been made in automated collision avoidance, but important challenges remain. PMD and AAPF-based systems require more testing to ensure durability in a variety of space settings, while FRDDM-DQN models require improved adaptability and training speed. Building a system to overcome these challenges is critical to the successful deployment of effective collision avoidance operations. Our research

presents a unique algorithm for autonomous collision avoidance in space situations. By combining real-time sensor data with advanced decision-making, the system dynamically alters the spacecraft's trajectory, ensuring exact obstacle avoidance with a low computational load. Its primary innovation is quick adaptation to changing circumstances, which allows for safe navigation across complex orbital areas. Given the essential role of collision avoidance in autonomous operations, it is important to thoroughly test and evaluate these systems within simulation settings that closely replicate real-world situations.

2.6 Simulation and Real-World Application

The gap between simulation and real-world application in robot studies poses a substantial obstacle to autonomous system reliability. This gap is caused by differences between controlled simulations and the unpredictability of real-world settings. Bridging this gap is essential for ensuring that theoretical models and algorithms perform as expected in real-world situations.

Simulation environments, such as ROS (Robot Operating System) and Gazebo, are effective platforms for testing and developing autonomous systems. They allow researchers and developers to test algorithms, sensor integrations, and robot behaviors in a safe and controlled environment. For instance, **Chikurtev's** [31] work on mobile robot simulation in ROS and Gazebo demonstrates the importance of these tools in developing and fine-tuning the autonomous navigational abilities of mobile robots, emphasizing the need of accurate simulation models that resemble real-world dynamics as closely as possible. Moreover, **Romero et al** [32] and **Takaya et al.**[33] underline the combination of simulation and real-world testing, describing the process for creating precise 3D maps with ROS and Gazebo. This approach allows code from simulations to be directly implemented on robot hardware without change, bridging the gap between simulated testing and real-world application. The study reveals that simulations can precisely replicate real-world situations, thus validating control systems developed in virtual environments. Our research aims to close these gaps by testing simulation results against real-world data wherever feasible, guaranteeing that control strategies and sensor integration approaches established in simulation can be efficiently applied to real-world applications.

In conclusion, the usage of simulation systems such as ROS and Gazebo plays an important role in connecting theoretical robotics models to real-world applications. Key resources such as the ROS and Gazebo documentation [34], [35], [36], a youtube channel [37], and courses in Udemy and Construct AI [38], [39], as well as research studies by Mengacci et al. [40], Ramón et al. [41], and Bhadani et al. [42], and github repositories [43], provides essential information and methodologies to this work. Continued innovation in these technologies will improve autonomous systems' reliability and adaptability, assuring their usefulness in a wide range of operational situations.

2.7 Conclusion

The development of autonomous docking systems represents an important breakthrough in engineering, particularly in the fields of space exploration, maritime activities, and unmanned aerial vehicles (UAV). These systems, which have improved control mechanisms and integrated sensor technologies, provide significant improvements in operational precision and reliability.

However, a key concern remains: a heavy reliance on simulations for validation. This disparity between theoretical models and practical implementation highlights the difficulty of deploying these technologies in uncertain real-world environments. The transition from controlled environment to operational settings requires thorough real-world setting and a design approach that allows for quick adaptation to unexpected challenges. Furthermore, while standardizing sensor suites across several missions is cost-effective, it may limit adaptation to individual mission needs, which could compromise system performance. Practical implementation requires finding a balance between cost-effectiveness and customization, ensuring that systems may be modified without major repairs.

To summarize, the future of autonomous docking systems is dependent on our abilities to refine testing procedures, improve simulation quality, and create adaptable system designs. These procedures are critical for ensuring that these systems perform consistently in their intended operational contexts, hence expanding the capabilities of autonomous docking in a variety of domains.

3 Research Methodology

3.1 Overview

This section describes the whole process used for developing a high-fidelity simulation for analyzing the autonomous docking capabilities of a chaser satellite with a target satellite. This includes the use of modern control algorithms and advanced sensor technologies. The goal is to evaluate the performance of these systems in simulated space situations, with a focus on their navigation, docking precision, and collision avoidance capabilities.

The simulation framework is based on Gazebo and the Robot Operating System (ROS), which were chosen due to their robust simulation capabilities, support for complex models, and significant libraries for real-time sensor data processing and control system testing. This combination enables accurate modeling of space environments as well as the physical dynamics of satellite interaction, both of which are required for realistic scenario testing.

3.1.1 Simulation Objectives

The primary Simulation Objectives in this project are:

1. **Evaluate Control Algorithms:** Test and validate various control methods, such as Proportional-Integral-Derivative (PID) control, Linear Quadratic Regulator (LQR), and a nonlinear controllers, to determine their effectiveness in precise maneuvering and docking operations.

2. **Examine sensor performance:** Evaluate the ability of onboard sensors - Velodyne LiDAR and depth camera systems—to detect and navigate around obstacles, including space debris, which is critical for autonomous collision avoidance.

3. **Optimize operational safety:** Develop and modify operational protocols to improve the safety and reliability of autonomous docking operations, ensuring that the satellite can operate under a variety of conditions, including unknown environmental changes or system failures.

4. **Optimize System Response:** To increase the satellite's autonomous capabilities, analyze the system's response to dynamic space environments.

3.1.2 Simulation Design Considerations

The Simulation Design considerations include several critical factors:

1. **Environmental Conditions:** Incorporating realistic representations of space, such as microgravity and the vacuum of space, which impact the operation of sensors and control systems.

2. **Designing Dynamic Scenarios:** The ability to develop dynamic testing scenarios that can simulate unexpected space circumstances, such as the sudden appearance of debris or changes to the target satellite's orbit, in order to evaluate the control systems' flexibility and capacity for decision-making.

3. **Physical Modeling:** Detailed modeling of the satellites' physical properties, including mass, size, surface characteristics, and inertia, is required to guarantee accurate simulations of their interactions and behaviors during docking operations.

3.1.3 Implementation

The implementation involves developing the simulation environment in Gazebo, configuring the satellite models, and integrating the ROS system for control and sensor data processing. This system is designed to provide a scalable testing environment in which various sensors are used and control algorithms are changed and tested under consistent conditions.

3.1.4 Performance Metrics

Key performance metrics have been devised to evaluate the simulation's efficiency.

1. **Docking Accuracy:** Measuring docking operation accuracy in terms of distance from the docking location and alignment accuracy during engagement.

2. **Sensor Accuracy:** Evaluating the Velodyne LIDAR sensor ability to accurately map the surroundings and identify obstacles

3. **Control System:** Evaluating the control systems' capacity to keep the satellite secure and on course under varying circumstances, as well as their response to sensor inputs.

4. **Efficiency Metrics:** Recording the time taken for maneuvers during each simulation run in order to measure the efficiency of the integrated systems.

By establishing these components, this method not only ensures a complete examination of the systems, but also sets the way for insightful analysis, that leads to significant improvements in the field of autonomous space systems. These key performance metrics support Objective 1.3.5 by evaluating the system's flexibility and efficiency in docking operations. Robust and efficient autonomous operations in simulated space environments are ensured by evaluating these metrics.

3.2 Setting Up Simulation Environment

The simulation environment is carefully designed to replicate space conditions as exactly as possible, using Gazebo for physical simulation and ROS for sensor data and control system management. This setup provides a flexible framework for simulating space dynamics as well as complex interactions between spacecraft during docking operations.

3.2.1 Tools & Setup

The simulation make use of Gazebo's capabilities for physically simulating satellites and their interactions, as well as ROS (Robot Operating System) for directing satellite operations, handling sensor data, and implementing communication protocols. This combination is essential for producing a high level of realism and dynamic response.

- **Gazebo**, known for its effective physics engine and 3D rendering capabilities, replicates the physical aspects of satellites like as mass, velocity, and collision dynamics. It also simulates environmental elements like microgravity and lack of lighting, which can affect satellite operations and sensor accuracy. It was chosen for its high-fidelity physics simulation and strong interaction with ROS, which provides a realistic environment for testing autonomous space operations.
- **ROS** Acts as the backbone for the simulation's control systems, providing a framework for developing custom control algorithms, integrating sensor data, and handling inter-process communications. ROS's modularity enables independent development and testing of subsystems like navigation and docking algorithms, which are essential to the success of autonomous docking operations. It was chosen because of its adaptability, real-time data processing, and extensive community support, which allow for easy integration of control algorithms and sensor data.

This setup not only replicates physical interactions between satellites, but it also makes it easier to integrate and test the control systems and algorithms required for autonomous space operations. Using Gazebo and ROS to simulate the space environment aligns with Objective 1.3.1, as it replicates orbital dynamics and environmental variables required for testing autonomous space operations. This realistic simulation enables the evaluation of autonomous systems under situations that are similar to those encountered in real-world scenarios.

3.2.2 Satellite Design

Both satellite's are designed as cuboids with solar panel wings, closely mimicking the architecture of the real satellites. Both satellites(Chaser and Target Satellite) are designed to replicate a docking operation in which the chaser's docking mechanism must precisely align and connect with the target's receptacle.

Chaser Satellite

As in Figure 3.1, the Chaser Satellite is designed with cylindrical docking platform, to actively engage and dock with the target's satellite. This satellite's key features also includes:

- **LiDAR:** Positioned to give comprehensive 360-degree spatial data that makes it easier to see obstacles and estimate distances accurately, allowing for accurate target navigation.
- **Depth Camera:** The depth camera provides extensive 3D imaging capabilities, which are essential for guiding the approach and ensuring proper docking alignment.
- **6-Axis Thrusters:** Provide precise maneuvering ability in all directions, which is required to move and align with the target satellite during docking.

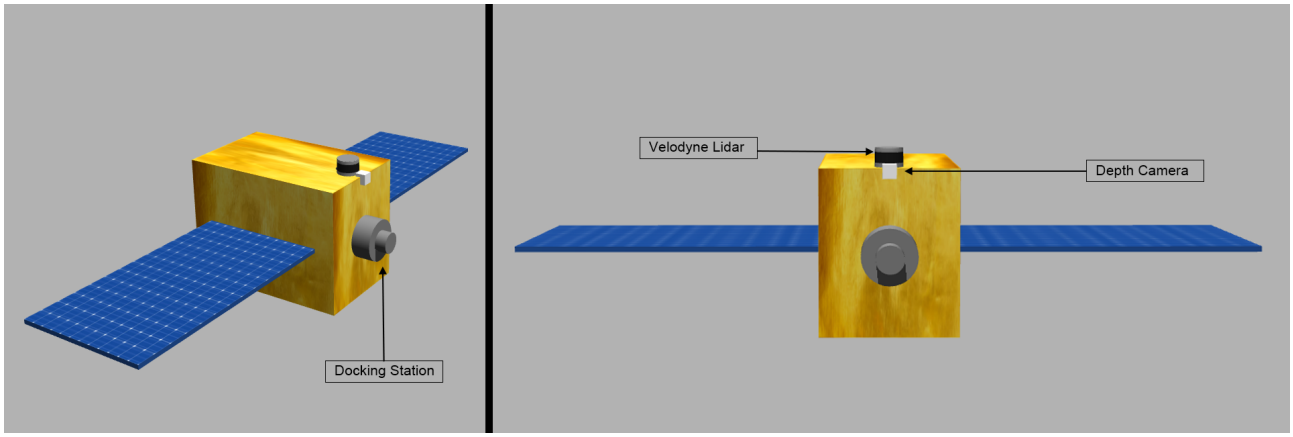

Figure 3.1: Chaser Satellite

Target Satellite

As in Figure 3.2, the target satellite's docking receptacle is a hollow cylinder that is particularly designed to guide and secure the chaser's docking platform. This design ensures a secure fit, minimizing the possibility of misalignment or disconnection during and after docking.

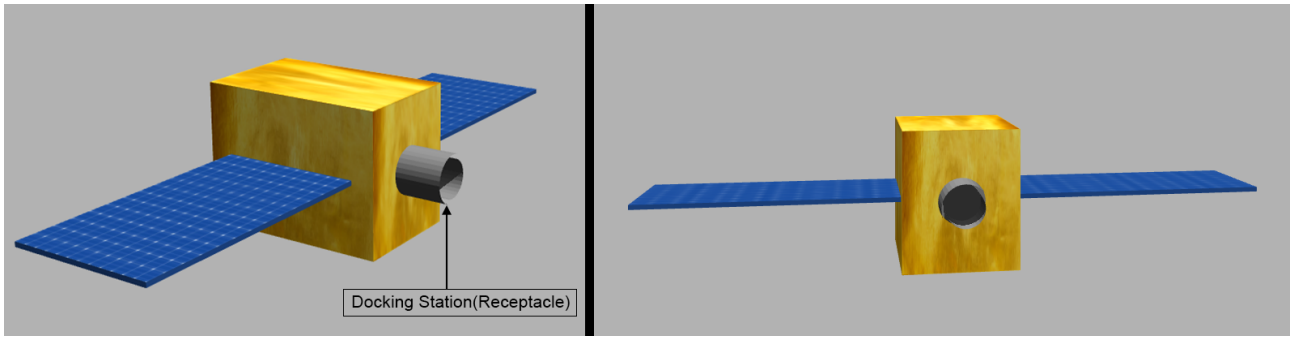

Figure 3.2: Target Satellite

3.2.3 Environment Design

As in Figure 3.3, The simulation environment precisely recreates Earth's orbital conditions in order to offer a realistic backdrop for satellite operations, with a focus on practical docking procedures in space.

- **Orbital Setting:** Both spacecraft operate in low Earth orbit (LEO) at around 400 km altitude, replicating normal operational conditions for many modern satellite missions. A little white dot seen in Figure 3.3 is a satellite.
- **Velocity and Dynamics:** Both satellites orbit Earth at a speed of around 7.88 km/s, which is similar to that of the ISS. The setup explores the control systems' capacity to perform high-speed docking maneuvers, which simulate the high relative velocities and complex interactions encountered in space missions.
- **Earth Dynamics:** The simulation includes an extensive representation of Earth, incorporating gravitational pull and the visual and operational consequences of being in and out of the Earth's shadow.

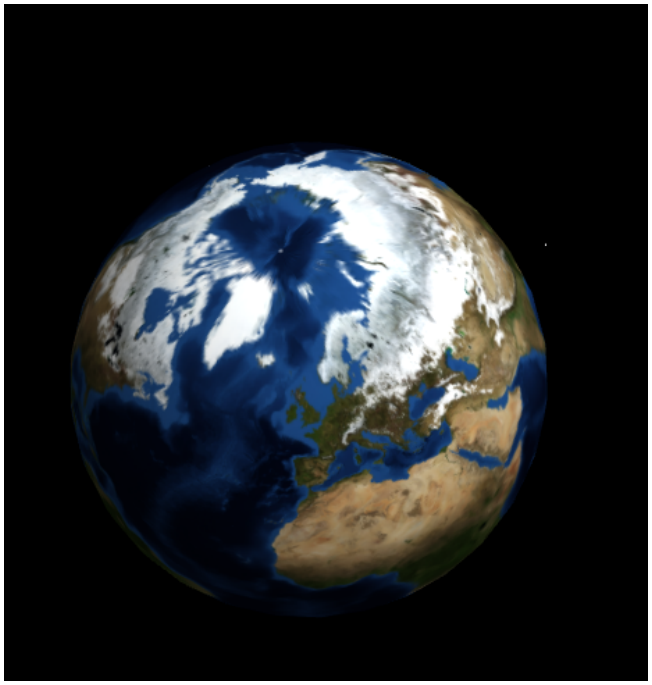

Figure 3.3: Environment Design

This practical simulation environment allows researchers and engineers to deeply test and modify satellite docking technologies and techniques, improving the reliability and safety of autonomous satellite operations in orbit. This environment not only supports the development of new technologies, but it also provides a platform for training and demonstration of operational scenarios that satellite operators may encounter in actual missions.

3.3 Control Systems

This section covers the development of control systems which control the docking operation between the chaser and target satellites. Three types of controllers are used: proportional-integral-derivative (PID), linear quadratic

regulator (LQR), and nonlinear controllers. Each of these systems uses the Hill-Clohessy-Wiltshire (HCW) equations to express relative motion in orbit, which is critical for precise and effective control in the microgravity environment of space.

3.3.1 HCW (Hill-Clohessy-Wiltshire) Equation

The **HCW equations** are essential for understanding relative motion in orbit, especially during satellite rendezvous and docking process. These equations explain the relative velocity of two satellites orbiting in close proximity, presuming circular orbits with negligible disturbances.

For the Derivation, Consider two satellites in a circular orbit: a target (which is stationary) and a chaser (which moves relative to the target). Let the satellite be in a circular orbit around the earth with radius R_0 and angular velocity w_0, and

x: Radial Direction(Away from earth)

y: Along Track Direction(Direction of satellite motion)

z: Cross- Track Direction(Normal to orbit plane)

As of Newton's Second Law [44] F = ma(Force = Mass x Acceleration)

Let (x,y,z) be the position co-ordinates of the chaser satellite relative to the target satellite, which is in circular orbit motion with radius R_0 and angular velocity w_0

The acceleration in the inertial frame can be transformed to the rotating frame using equation(3.1)[45]:

$$a_{rot} = a_{inertial} - 2w.V_{rel} - w(w.r_{rel}) - \dot{w}.r_{rel} \tag{3.1}$$

where,

$$w = w_0\hat{z} \; and \; r_{rel} = x\hat{x} + y\hat{y} + z\hat{z} \tag{3.2}$$

The Centripetal Acceleration of the target satellite is $w_0^2 R_0$. The relative position vector is small, so the equations around the reference orbit is linearized in equations(3.3), (3.4), (3.5) by derivation in [46]:

$$\ddot{x} - 2w_0\dot{y} = -3w_0^2 x \tag{3.3}$$

$$\ddot{y} + 2w_0\dot{x} = 0 \tag{3.4}$$

$$\ddot{z} + w_0^2 z = 0 \tag{3.5}$$

now, the equations(3.3), (3.4), (3.5) will be,

$$\ddot{x} - 2w_0\dot{y} - 3w_0^2 = U_x \tag{3.6}$$

$$\ddot{y} + 2w_0\dot{x} = U_y \tag{3.7}$$

$$\ddot{z} + w_0^2 z = U_z \tag{3.8}$$

where, U_x, U_y, U_z from equations(3.6), (3.7), (3.8) are the control inputs along each axis, representing the forces needed to be exerted by the chaser's thrusters. These equation describes the relative motion of a chaser satellite with respect to a target satellite in a circular orbit, under the assumptions of small relative distances and linear dynamics.

Optimizing the HCW equations for relative motion prediction helps achieve Objective 1.3.4 by increasing the accuracy of control algorithms, used for calculating the chaser satellite's trajectory. The improvement is essential for accurate docking and overall trajectory planning.

3.3.2 Proportional-Integral-Derivative (PID) Controller

The **PID controller** is a feedback mechanism that is often used in control systems for its simplicity and effectiveness under a variety of scenarios. It modifies the control outputs depending on the difference between desired and actual positions, with the aim to minimize this error over time. A PID controller's control law includes of three constant parameters: proportional, integral, and derivative gains(denoted as K_p. K_i and K_d) [47]

$$\text{output} = K_p \cdot \text{e(t)} + K_i \cdot \int \text{e(t)}\,dt + K_d \cdot \frac{d(\text{e(t)})}{dt} \tag{3.9}$$

where in equation (3.9),

- e(t) is the error
- $\int \text{e(t)}\,dt$ is the integral of the error over time, which helps to reduce residual steady-state error.
- $\cdot\frac{d(\text{e(t)})}{dt}$ is the derivative of the error, which indicates its future direction.

PID controller is used for its:

- **Simplicity and Robustness:** The PID controller provides a simple yet effective method for ensuring stability and precision in a variety of docking settings.
- **Ease of Tuning:** Its settings are easily customizable, making it adaptable to various operational situations and allowing for reliable docking actions.

Algorithm 1 describes a satellite docking process utilizing PID controllers and Hill-Clohessy-Wiltshire (HCW) equations. This simulation aims to direct the course of a chaser satellite toward a docking operation with a target satellite by accurately modifying its trajectory based on real-time positional errors.

The process begins with the setup of three PID controllers, one for each spatial dimension (x, y, and z). These controllers are crucial for accurately modifying the satellite's position relative to the target. In order to control the satellite's relative motion, it also creates the orbital mean motion and initial alignment flags.

The algorithm's core operates in a continuous loop for as long as the ROS system is running, providing real-time responsiveness. At each iteration, the program initially checks to see if the chaser and target satellites' present positions are still available. If available, it calculates the time since the last update and uses it, together with the distances and velocities, to calculate the necessary modifications using HCW equations.

Orientation adjustment is an important initial step in docking, in which the required orientation is determined using quaternion mathematics to precisely align the chaser with the target. The method then proceeds with sequential alignments, beginning with the z-axis and verifying if the positional difference is within a predefined tolerance before setting the alignment flag. If not aligned, the PID controller determines the necessary correction. After aligning the z-axis, it repeats the process for the y-axis before moving on to adjust the x-axis.

The alignment in each axis is dynamically managed: if the satellite is already aligned in an axis, it uses direct HCW accelerations to make changes; otherwise, it uses the PID controller to minimize positional errors. After aligning all axes and computing the necessary positional and orientation modifications, the revised trajectory with the new coordinates and orientation is updated.

This system not only guarantees a logical and controlled approach to satellite docking by controlling modifications in stages, but it also combines dynamic management with strong mathematical modeling to handle real-time changes in satellite locations and velocities. This systematic methodology is critical for performing accurate, safe, and practical docking operations in the challenging conditions of space.

Algorithm 1 Satellite Docking with PID Controller

Initialize PID controllers for x, y, and z axes, and mean motion n
Initialization:
    kp, ki, kd, max_output
    integral $\leftarrow 0$
    last_error $\leftarrow 0$
function COMPUTE(error, dt)
    integral $\leftarrow$ integral + error $\times$ dt
    derivative $\leftarrow$ (error - last_error) / dt
    last_error $\leftarrow$ error
    output $\leftarrow$ kp $\times$ error + ki $\times$ integral + kd $\times$ derivative
    output $\leftarrow$ Clamp(output, max_output)
    return output
end function
Set $z\_aligned \leftarrow$ False, $y\_aligned \leftarrow$ False
while ROS is running **do**
    if chaser_pose and target_pose are available **then**
        Compute Δt and HCW accelerations a_{x_hcw}, a_{y_hcw}, a_{z_hcw}
        # **Orientation Adjustment**
        Compute $desired\_orientation \leftarrow \text{ComputeQuaternion}(target\_pose - chaser\_pose)$
        # **Z-Axis Alignment**
        Compute $z\_diff \leftarrow target\_pose.z - chaser\_pose.z$
        if $|z\_diff| <$ tolerance **then**
            $z\_aligned \leftarrow$ True
        end if
        Update $new\_z \leftarrow chaser\_pose.z + (PID.compute(z\_diff, \Delta t)$ **if not** $z\_aligned$ **else** $a_{z\_hcw} \cdot \Delta t)$
        # **Y-Axis Alignment**
        if $z\_aligned$ **then**
            Compute $y\_diff \leftarrow target\_pose.y - chaser\_pose.y$
            if $|y\_diff| <$ tolerance **then**
                $y\_aligned \leftarrow$ True
            end if
            Update $new\_y \leftarrow chaser\_pose.y + (PID.compute(y\_diff, \Delta t)$ **if not** $y\_aligned$ **else** $a_{y\_hcw} \cdot \Delta t)$
        end if
        # **X-Axis Alignment**
        if $z\_aligned$ **and** $y\_aligned$ **then**
            Compute $x\_diff \leftarrow target\_pose.x - chaser\_pose.x$
            Update $new\_x \leftarrow chaser\_pose.x + PID.compute(x\_diff, \Delta t) + a_{x\_hcw} \cdot \Delta t$
            Publish trajectory with new_x, new_y, new_z, and $desired\_orientation$
        end if
    end if
    Sleep for loop rate
end while

3.3.3 Linear Quadratic Regulator(LQR) Controller

The **LQR controller** aims to identify the control inputs that will drive the state of a system to a particular set point or trajectory while minimizing a given cost function. This cost function is usually a quadratic form that balances the state deviation from the objective with the effort (energy or size) of the control inputs. [48]

The system is represented in state-space form, which is standard method to represent linear dynamic systems. The standard form is in equation(3.10):

$$\dot{x} = Ax + Bu \tag{3.10}$$

where in equation(3.10),

- x is the state vector of the system,
- u is the control input vector,
- A is the state matrix, representing the system dynamics,
- B is the input matrix, describing how control inputs affect the system.

The LQR aims to minimize the cost function J over an infinite time horizon.

$$J = \int_0^\infty (x^T Qx + u^T Ru)\, dt \tag{3.11}$$

where in equation(3.11),

- Q is a symmetric positive semi-definite matrix that weighs the state errors.
- R is a symmetric positive definite matrix that weighs the control effort.

The goal is to generate a control matrix (K) that minimizes the cost function. The optimal K is determined by solving the **Algebraic Riccati Equation (ARE)** in equation(3.12) [49]:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{3.12}$$

where P is a symmetric positive definite matrix. Once P is calculated, the control law is defined as in equation(3.13):

$$K = R^{-1}B^T P \tag{3.13}$$

The optimal control input for minimizing the cost function is given by in equation(3.14):

$$u = -Kx \tag{3.14}$$

LQR is chosen for its:

- **Optimal Control:** LQR optimizes control by minimizing a cost function that balances precision and control effort, making it excellent for accurate spacecraft maneuvering.
- **Predictive Capability:** The state-space method enables accurate predictions and smooth functioning in multi-input, multi-output systems.

Algorithm 2 Satellite Docking with LQR Controller

```
Initialize LQR controller
function INITIALIZE((A, B, Q, R))
    Compute K using Riccati Equation
    function COMPUTE(state_error)
        return −K · state_error
    end function
    Set z_aligned ← False, y_aligned ← False
    while ROS is running do
        if chaser_pose and target_pose are available then
            Compute HCW accelerations a_x_hcw, a_y_hcw, a_z_hcw
            Calculate target direction vector
            Compute desired_orientation quaternion
            Calculate time difference Δt
            Compute state error (position and velocity differences)
            Calculate LQR control input
            # Same as before
            # Orientation Alignment with LQR
            # Z-Axis Alignment with LQR
            # Y-Axis Alignment With LQR
            # X-Axis Alignment with LQR
        end if
        Sleep for loop rate
    end while
```

Algorithm 2 designed to control the autonomous docking procedure between a chaser satellite and a target satellite . The script uses a Linear-Quadratic Regulator (LQR) control method, which is appropriate for applications requiring precision control, such as satellite docking.

The algorithm starts solving the continuous-time Algebraic Riccati Equation (ARE) utilizing system dynamics matrices A and B, as well as weighting matrices Q and R. The ARE solution, matrix P, is used to generate the optimal gain matrix K. This defines the control action to minimize a cost function that balances state error and control effort.

Then it establishes a ROS node and configures publishes and subscribers to handle communication with the simulation environment. The system dynamics matrices A and B are generated to characterize the satellite's key kinematic relations and directly transform control inputs into accelerations.

This controller listens for position updates from both the chaser and target satellites via ROS topics. These updates initiate computations that determine the control operations required to align and dock with the target. The program calculates relative motion dynamics using a simplified version of the Hill-Clohessy-Wiltshire (HCW) equations, which have been modified to add control inputs which influence the satellite's trajectory.

During each control cycle, the script calculates the current state error between the chaser and the target, which is then utilized by the LQR controller to calculate the appropriate control input to reduce the error. This controller, like the previous one, first corrects orientation, then aligns along the z-axis, followed by the y-axis and x-axis using the LQR controller, continuously adjusting the satellite's position and orientation until docking, and applying advanced control theory in simulation for real-world space mission preparation.

3.3.4 Non-Linear Control System

This non-linear control system uses the dynamic inversion approach to build a control input u that drives the chaser satellite's acceleration $\ddot{x}$ to match the intended trajectory $\ddot{x_r}$, simply canceling out the inherent nonlinearities of the system. The system dynamics are given by in equation(3.15):

$$\ddot{x} = F(x,\dot{x}) + H(x,\dot{x})u \tag{3.15}$$

where,

- $\ddot{x}$ is the chaser satellite's acceleration in relation to the target
- $F(x,\dot{x})$ describes the nonlinear dynamics of the system, reflecting how the position and velocity impact its motion without any control input.
- $H(x,\dot{x})$ is the control effectiveness matrix, shows how control inputs impact the system's acceleration.
- u is the control input, which represents the thrusts or forces applied by the satellite's propulsion system.

The objective is to make the chaser's acceleration$\ddot{x}$ align with the desired acceleration$\ddot{x_r}$, which is defined by the docking trajectory requirements. To achieve equation(3.16):

$$\ddot{x} = \ddot{x_r} \tag{3.16}$$

This gives the control equation(3.17):

$$\ddot{x_r} = F(x,\dot{x}) + H(x,\dot{x})u \tag{3.17}$$

To accomplish the desired acceleration, the control input u is computed using dynamic inversion in equation(3.18):

$$u = H(x,\dot{x})^{-1}(\ddot{x_r} - F(x,\dot{x})) \tag{3.18}$$

here in equation(3.18):

- $H(x,\dot{x})^{-1}$ is the inverse of the control effectiveness matrix, giving it is non-singular.
- $(\ddot{x_r} - F(x,\dot{x}))$ Calculates the difference between the desired acceleration and the system's natural dynamics, isolating the component of acceleration required by the control inputs.

Algorithm 3 Satellite Docking with Non-Linear Control System

```
Initialize Non-Linear Control System
function INITIALIZE((mean_motion)
    Set mean_motion
    function COMPUTE_NONLINEARITIES(x, ẋ)
        F_x = x^2 + sin(x) + abs(x)
        F_y = exp(x) + x * x_dot
        F_z = sin(x) + x_dot^2
        H = Identitymatrix(3x3)
        return F = [F_x, F_y, F_z], H
    end function
    function COMPUTE_CONTROL_INPUT(ẍ_r, x, ẋ_r)
        F, H = compute_nonlinearities(x, ẋ))
        H_inv = Inverse_of_H
        u = H_inv * (ẍ_r − F)
        return u
    end function
    Set z_aligned ← False, y_aligned ← False
    while ROS is running do
        if chaser_pose and target_pose are available then
            Set mean_motion
            Initialize NonlinearController with mean_motion
            Compute HCW accelerations a_x_hcw, a_y_hcw, a_z_hcw
            Non-Linear Control Inputs
            # Same as before
            # Orientation Alignment with Non-Linear Control Inputs
            # Z-Axis Alignment with Non-Linear Control Inputs
            # Y-Axis Alignment with Non-Linear Control Inputs
            # X-Axis Alignment with Non-Linear Control Inputs
        end if
        Sleep for loop rate
    end while
```

This Non-Linear System is used for its:

- **Handling Non-Linear Dynamics:** This controller directly handles nonlinearities in spacecraft dynamics, which are essential to complex maneuvers.
- **Dynamic Inversion:** Allows precise control even under uncertain settings, leading to dependable docking in non-linear environments.

Algorithm 3 offers a sophisticated nonlinear control method for satellite docking, which uses dynamic inversion to successfully manage the complex nature of orbital dynamics. The basis of this technique is to calculate the nonlinear dynamics and control matrix based on the satellite's current position and velocity. These computations allow the system to provide the appropriate control inputs to directly counteract these dynamics, allowing the chaser satellite to follow the selected trajectory towards the target satellite.

This strategy depends on dynamic inversion, which reverses the effects of the satellite's inherent dynamics (expressed by the function $F(x,\dot{x})$ and uses the control matrix $H(x,\dot{x})$ to apply exact forces. The control inputs u coordinate the satellite's acceleration with docking procedures, altering its trajectory to correspond with with the planned course. This method consists of solving for u to correct for inherent system behaviors and move the satellite towards the target with the necessary acceleration $\ddot{x_r}$

This setup guarantees that control commands are dynamically updated to reflect changes in the satellite's environment or its position with respect to the target. This control method is crucial for precise satellite docking, as adapting to dynamic situations is essential. Its success is dependent on precise modeling of satellite dynamics, which necessitates thorough simulation testing before real-world use.

The PID, LQR, and non-linear control systems play an important role in satisfying Objective 1.3.3, which examines control methods for accurate satellite maneuvering.

3.4 Sensor Integration

In the harsh conditions of space, sensor integration plays an important role in autonomous satellite operations, particularly docking procedures. Integrating and processing data from a Velodyne LiDAR with a depth camera guarantees accuracy, safety, and dependability in these challenging tasks. This section describes the sensors utilized, data processing algorithms, and direct applications in satellite docking operations. The Sensor Capabilities are:

- **Velodyne Lidar:** This sensor is key for producing detailed 3D maps of the environment, since it provides high-resolution point clouds and the accurate distance measurements required for navigation and obstacle

identification. The Velodyne Points are viewed in Rviz[50], which provides the data about its surrouding as in Figure 3.4.

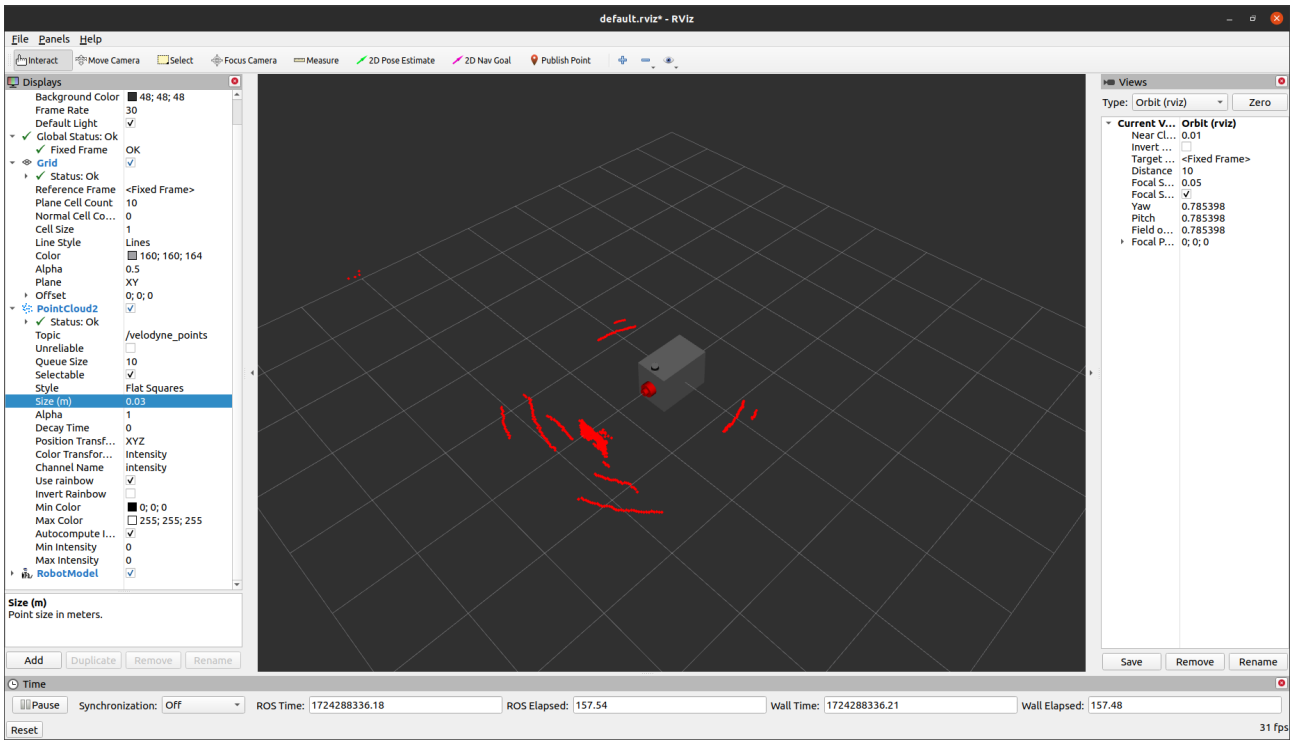

Figure 3.4: Velodyne Points

- **Depth Camera:** Complements LiDAR by providing depth information through visual imaging, improving object recognition capabilities, and adding texture and color features that LiDAR does not capture, as well as being used for further confirmation in docking operations. This integration is displayed in RViz, which provides a complete picture of the environment as in Figure 3.5.

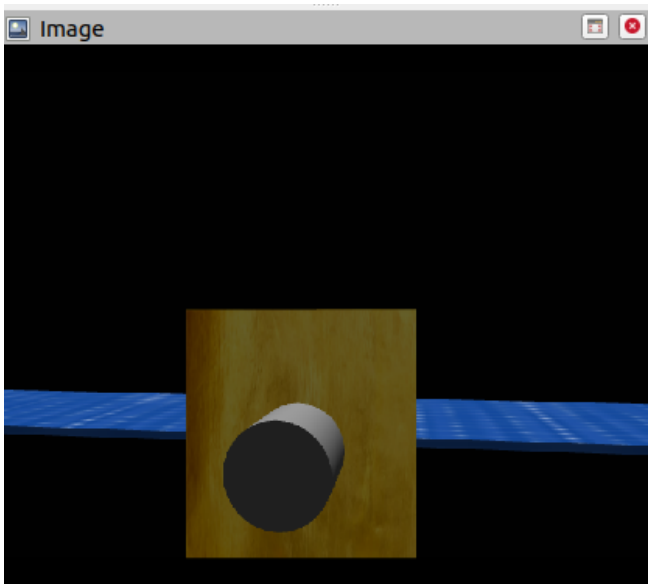

Figure 3.5: Depth Camera

Algorithm 4 Sensor Data Processing - Point Cloud

```
Subscribe to the '/velodyne_points' topic to receive PointCloud2 data
function LIDAR_CALLBACK(data)
    point_cloud ← convert_ros_to_pcl(data)
    filtered_cloud ← voxel_grid_filter(point_cloud)
    clusters ← euclidean_clustering(filtered_cloud)
    detect_satellite_and_debris(clusters)
end function
function CONVERT_ROS_TO_PCL(ros_cloud)
    Convert the ROS PointCloud2 message into PCL PointCloud format
    return pcl_data
end function
function VOXEL_GRID_FILTER(cloud)
    Apply voxel grid filtering to downsample the point cloud
    return filtered_cloud
end function
function EUCLIDEAN_CLUSTERING(cloud)
    Perform Euclidean clustering to segment the point cloud into clusters
    return clusters
end function=0
```

As in Algorithm 4, Firstly, it subscribes to the *"/velodyne_points"* topic, which is where the LiDAR sensor sends its point cloud data. This subscription calls the *lidar_callback* function whenever new data comes. This function does the point cloud processing task, namely:

- **Conversion to PCL:** The *'convert_ros_to_pcl'* function transforms ROS PointCloud2 messages into PCL (Point Cloud Library) format, enabling proficient point cloud operations. PCL is a widely used library for 2D/3D image and point cloud processing.
- **Voxel Grid Filtering:** This function uses a voxel grid filter to downsample point cloud data processed previously, making the analysis quicker and less resource-intensive. The amount of downsampling can be controlled by setting the voxel grid size.
- **Euclidean Clustering:** The function splits the point cloud into clusters using Euclidean clustering, which groups points by their spatial proximity. This is essential to separating particular items from the surroundings, such as space debris or docking station.

These divided clusters are sent to *'detect_satellite_and_debris'* function to detect whether the derived data is satellite or space junk to avoid.

Algorithm 5 Sensor Data Processing - Object Detection and Classification

```
..... Continuation
Subscribe to '/target_pose ' topic to check if target is known
Initialize debris_positions ← []
function DETECT_SATELLITE_AND_DEBRIS(clusters)
    for each cluster in clusters do
        points ← cluster.to_array()
        volume ← calculate_volume(points)
        centroid ← calculate_centroid(points)
        if is_satellite(volume) and target_pose = None then
            target_pose ← centroid
            Log "Satellite detected and target assigned."
        else if is_debris(volume) then
            Append centroid to debris_positions
            Log "Space debris detected at position: centroid"
        end if
    end for
end function
function CALCULATE_VOLUME(points)
    Calculate the bounding box volume of the point cloud cluster
    return volume
end function
function CALCULATE_CENTROID(points)
    Calculate the centroid of the point cloud cluster
    return centroid
end function
function IS_SATELLITE(volume)
    Determine if a cluster might be a satellite based on volume
    return True or False
end function
function IS_DEBRIS(volume)
    Determine if a cluster might be space debris based on volume
    return True or False
end function
while ROS is running do
    Continuously process incoming LiDAR data
end while
```

As in Algorithm 5, it initally subscribes to *'target_pose'* topic check if any target satellite position is already found. If not, this would assign newly found target to the *'target_pose'* to proceed it with docking procedure.

After Clustering, this code evaluates each cluster to determine if it is the target spacecraft or space debris. This includes calculating the volume and centroid of each cluster. Volume Calculation is a straightforward but

accurate method to calculate the volume of an object in space by using the spread (peak-to-peak) of points within each cluster. Centroid Calculation calculates the cluster's geometric center, which aids in determining an object's position in relation to the satellite.

Based on the volume, clusters are classified as either satellite or debris, using pre-defined volume ranges specific to each category. If a target satellite is found, it proceeds to docking procedure or if its space debris, it proceeds to collision avoidance process. By detecting and classifying objects surrounding the satellite, can help in safe docking operations, preventing collisions and assisting during navigation.

The current implementation of this method enhances environmental awareness by successfully integrating sensor inputs; yet, it fails to account for drifts, measurement noise, and sensor biases. These factors can have major effects on the accuracy and reliability of sensor data, particularly in harsh conditions of space.

- **Measurement Noise:** Random variations in sensor data that may cause the surroundings to appear distorted. Although filtering techniques can effectively reduce noise, our approach does not specifically address this.
- **Sensor Biases:** Systematic errors lead sensors to constantly provide incorrect data. Biases may arise from a variety of causes, including sensor misalignment or calibration issues.
- **Drifts:** Sensor readings alter gradually over time, potentially leading to errors in long-term missions. Drifts are especially important for systems that must function continuously over long periods of time.

In order to minimize these impacts and enhance system performance generally, future study could include implementing strategies like sensor calibration, error modeling, and adaptive filtering into practice.

Integrating Velodyne LiDAR and depth cameras, along with data fusion methods, achieves Objective 1.3.2. This method improves the chaser satellite's spatial awareness, which is essential to accurate docking and autonomous operations, resulting in safe and accurate motions.

3.5 Collision Avoidance

Collision avoidance is an important aspect of autonomous satellite operations, particularly during complex maneuvers like docking or navigating through debris-filled orbits. This section outlines the techniques and technology used to assure satellites' safe passage by avoiding collisions with unexpected risks.

Algorithm 6 Collision Avoidance for Satellite

Initialize ROS node for collision avoidance
Initialize $current\_satellite\_position$ and $debris\_positions$ //(Data From Object Detection)
Set $safe\_distance \leftarrow$ 1.0 meters
Initialize *avoidance_pub* to publish trajectory on '/satellite/trajectory' topic
function CALCULATE_DISTANCE(point1, point2)
    Compute Euclidean distance between $point1$ and $point2$
    return distance
end function
function AVOID_DEBRIS
    Initialize $safe\_position \leftarrow target\_pose$
    for each $debris$ in $debris\_positions$ **do**
        $distance \leftarrow$ calculate_distance$(safe_position, debris)$
        if $distance < safe\_distance$ **then**
            Compute $direction \leftarrow safe\_position - debris$
            Normalize $direction$
            Calculate $avoidance\_move \leftarrow direction \times (safe\_distance - distance + 0.1)$
            Update $safe\_position \leftarrow safe\_position + avoidance\_move$
        end if
    end for
    publish_trajectory(safe_position)
end function
function PUBLISH_TRAJECTORY(safe_position)
    Create a trajectory message with $safe\_position$
    Publish the trajectory to avoid debris
end function

The inital part of the algorithm 6, starts with initialising positions of of current satellite position and known debris position. Then, it sets *'safe_distance'* parameter, which defines the minimum distance the satellite has to maintain from the debris to avoid collision.

This code uses the *'calculate_distance'* function to compute the Euclidean distance between the satellite's present position and each piece of debris. This is an essential step in determining how close the satellite is to potential hazards. If the satellite is within an unsafe range of debris, this system creates an avoidance route. It computes and normalizes the direction vector away from the debris. The satellite then changes its position along this vector, just enough to re-establish a safe buffer zone with an additional degree of safety.

After calculating the safe position, the *'publish_trajectory'* method changes the satellite's trajectory. The trajectory is then published to the *'/satellite/trajectory'* topic, providing that the navigation system receives and executes the updated trajectory in real time. The system runs in a continuous loop, monitoring and correcting the satellite's trajectory on a high-frequency cycle to ensure that the satellite adapts quickly to dynamically

changing conditions in its immediate environment.

By constantly monitoring the surroundings and altering the satellite's path as needed, the system helps guarantee that the spacecraft remains on track and avoids collisions with debris or other orbital risks. This method not only protects the satellite's integrity, but also makes certain that its mission objectives are accomplished successfully and securely.

3.6 System Design

The System Design outlines the architectural and functional design of the simulation environment used to evaluate a satellite's autonomous docking capabilities. This design employs control algorithms, sensor technologies, and other systems that support real-world space operations in an efficient and safe manner. The design effectively integrates real-time data processing with efficient decision-making processes, shown in the System Flow Diagram (Figure 3.6).

The System design has been divided into several key components, each of which plays an important part in the overall operation of the satellite system:

1. **Sensors(LiDAR, Camera, IMUs):** These Sensor collects environmental data, such as LiDAR gathers velodyne points based on its surroundings, camera gathers visual picture, and IMUs[51] gathers specific force, angular rate and orientation of the satellite. These Sensor data are transmitted to Data Processing Unit, where they are analysed and integrated.

2. **Data Processing Unit:** This component processes raw sensor data to provide an organized and useful model of the satellite's environment. It analyses the data and transmits relevant information to the Decision Making System to make gained decisions.

3. **Decision Making System:** This system evaluates processed data to make significant decisions about docking operations, collision avoidance, and system responses to identified obstacles. It also provides data to monitor system operations and for communications.

    - **Docking Operation:** If the target is known, it commands the Control System to execute accurate docking operations.
    - **Collision Avoidance:** If any space debris is found, it executes the collision avoidance system, which uses satellite thrusters to avoid collision.

4. **Control System:** This System uses any one of the controllers(PID, LQR, Non-linear) and executes the docking protocols to precisely dock with the target. It sends commands to satellite thrusters to adjust the satellite's position, speed and orientation.

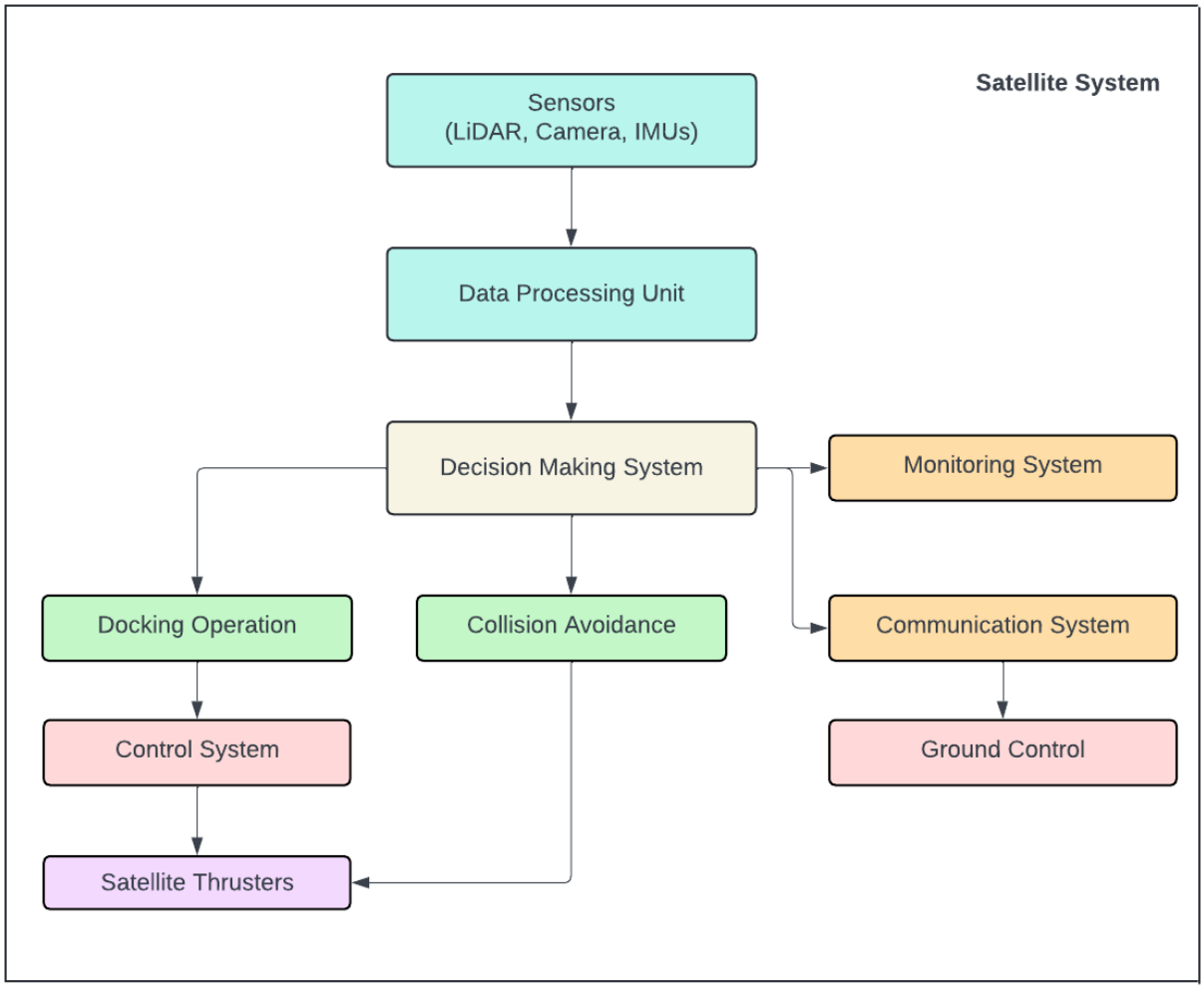

Figure 3.6: System Flow Diagram

5. **Satellite Thrusters:** This executes movement commands from the Control System and changes the satellite's trajectory or maintains a specific orbit.

6. **Monitoring System:** It gets information from the Decision Making System, which provides valuable input for adaptive responses and system improvements, and monitors the health and condition of all satellite activities, providing optimal system performance and alerting when defects are found.

7. **Communication Systems:** Manages all data transfers between the satellite and Ground Control, involving intra-satellite communications and command and data relay to ensure the mission's objectives are met, and provides remote intervention as necessary.

8. **Ground Control:** Overarching mission control and support, including additional computing resources and human oversight. It receives and sends commands and data to the satellite via the Communication System, which is vital to strategic decision-making and emergency response.

This system design combines innovative technology with robust control mechanisms to provide fast and

secure autonomous docking operations. By clearly outlining each component's tasks and interactions, the design guarantees a high level of reliability and performance, which is important for the different situations encountered in space missions.

3.7 Experiment Design

The experiments are designed to address solution-driven problems, with a particular focus on understanding the system's performance under various docking circumstances and collision avoidance strategies. The methodology is aimed at helping examine and gather data on the satellite system's capabilities in four key operational scenarios:

1. **Two-Plane Docking(Y, X):** It tests the system's ability to align and dock along the X and Y axes.
2. **Three-Plane Docking(Z, Y, X:** It examines the system's accuracy and efficiency in performing complex moves in three axes.
3. **Docking with Orientation Change:** It evaluates the system's response to changes in orientation before the docking, simulating the satellite's orbital adjustment to bring it into line with the docking station.
4. **Avoiding Debris:** It assesses how well the system detects obstacles and can avoid them to avoid collisions.

The experiment methodology for testing the autonomous satellite docking system uses the ROS and Gazebo simulation environment for performing a series of organized tests across these four scenarios. Each scenario is carefully created to replicate realistic space conditions, allowing the system's sensors and control algorithms to be evaluated in dynamic environments. These simulations provide a controlled, reproducible environment for evaluating the system's response to docking maneuvers and obstacle detection, providing thorough testing of the satellite's navigation and safety functions. Multiple trials are carried out for each situation to ensure that the system's performance is reliable and robust.

Systematic testing of docking scenarios helps fulfill Objective 1.3.6 by providing concrete information needed to develop methods for autonomous rendezvous, docking, and collision avoidance, thereby improving the reliability of future space missions.

4 Results

This chapter offers an in-depth examination and presentation of the findings from the tests done to assess the satellite's autonomous docking capabilities. The findings are organized based on the situations specified in the experiment design, comparing both the baseline and improved systems. Each part includes both quantitative and qualitative data, along with relevant observations and insights.

4.1 Two-Plane Docking

This section presents the results of the two-plane docking tests, in which the chaser satellite aligns along one axis (Y) before adjusting along the other axis (X) to achieve successful docking. The following figures show the procedure of this docking maneuver.

Figure 4.1 shows the initial setting of the chaser satellite preparing for the two-plane docking procedure. The chaser starts by aligning along the Y-axis, as shown in Figure 4.2. The Velodyne LiDAR points provide vital spatial information to control the satellite's motions. Once the Y-axis is fixed, then the chaser satellite starts to align along the X-axis, as seen in Figure 4.3. At last, Figure 4.4 shows that the docking process was successfully completed.

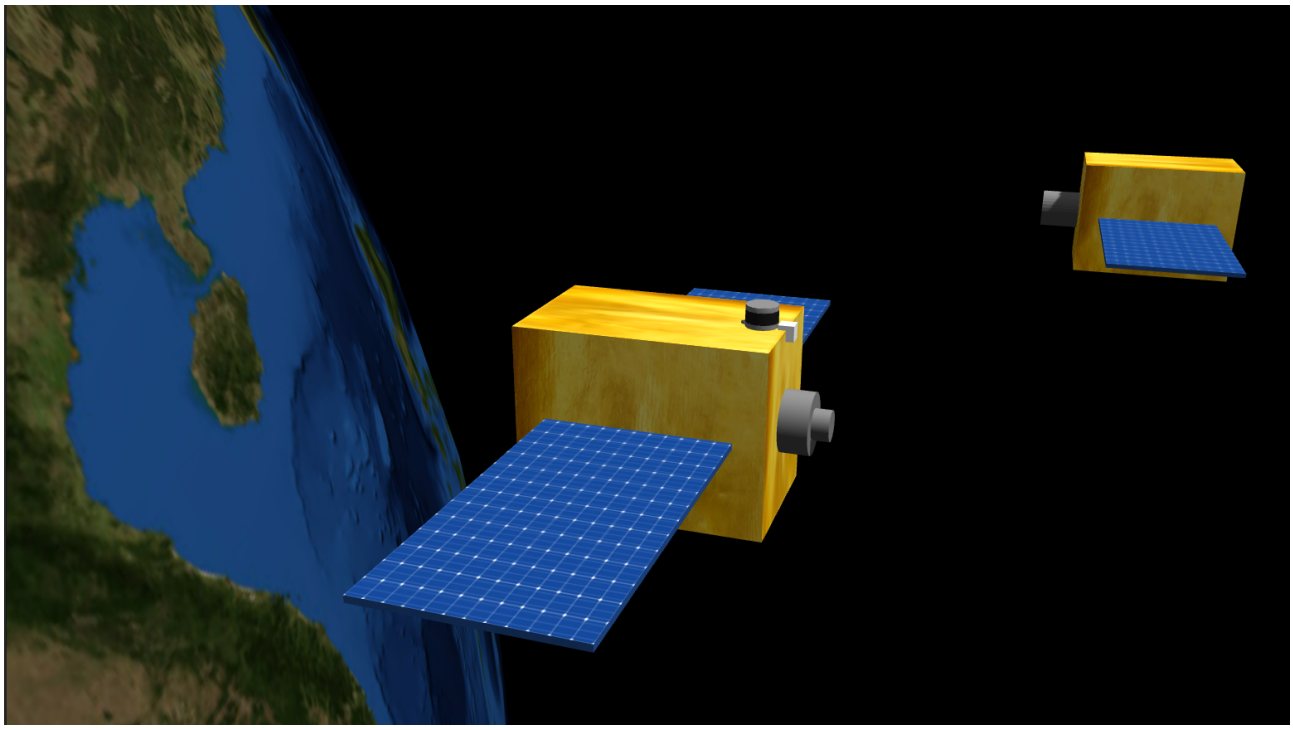

Figure 4.1: Two-Plane Scenario

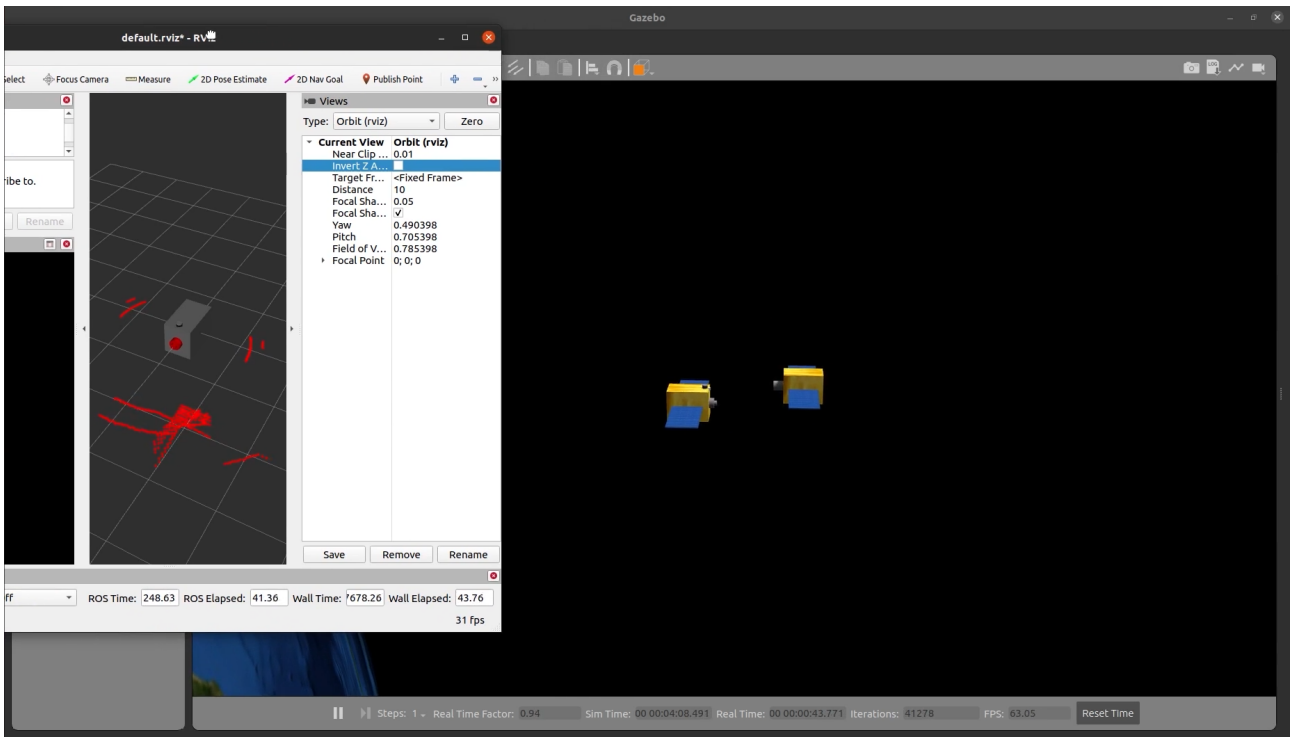

Figure 4.2: Correcting Y-axis in Two-Plane

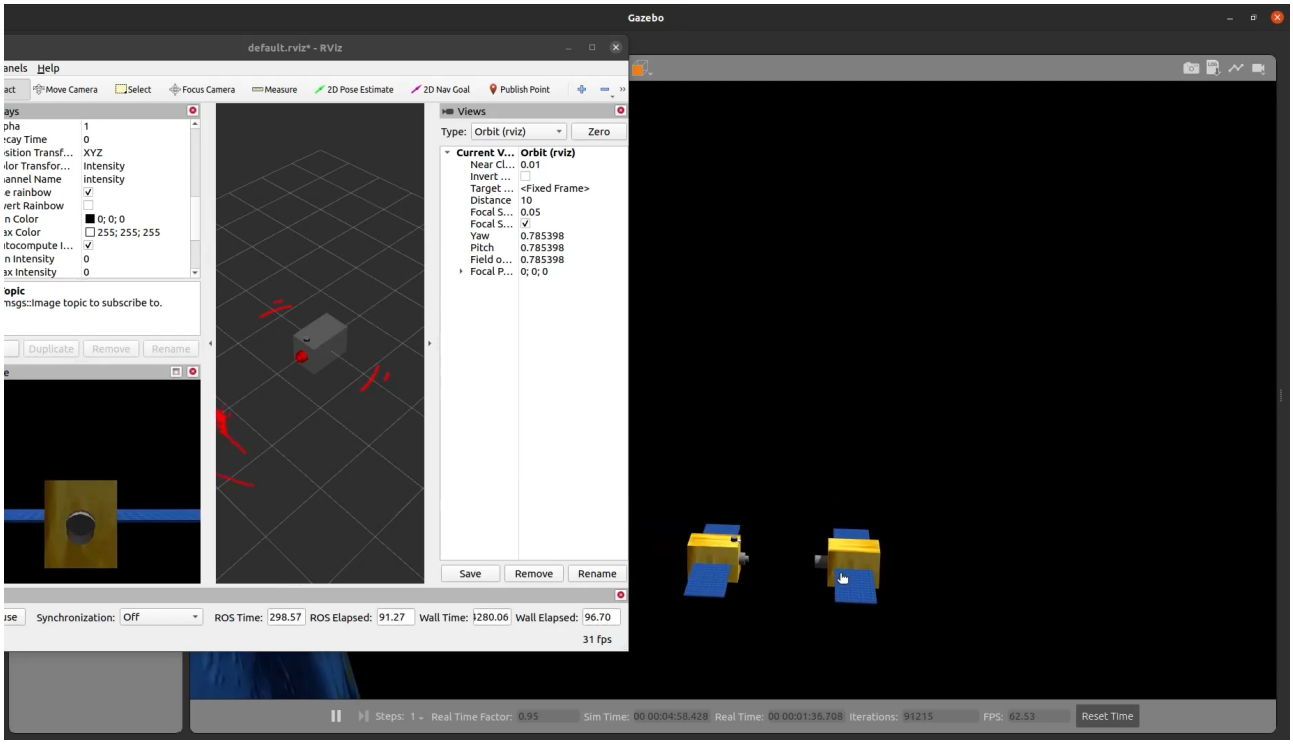

Figure 4.3: Now Docking X

Following a successful docking maneuver, a comparison study of the three control systems (PID, LQR, and Non-linear) is performed to establish their efficiency in this scenario based on the performance metrics, such as docking accuracy error and docking time. To confirm the authenticity of the results, each controller conducted multiple testing.

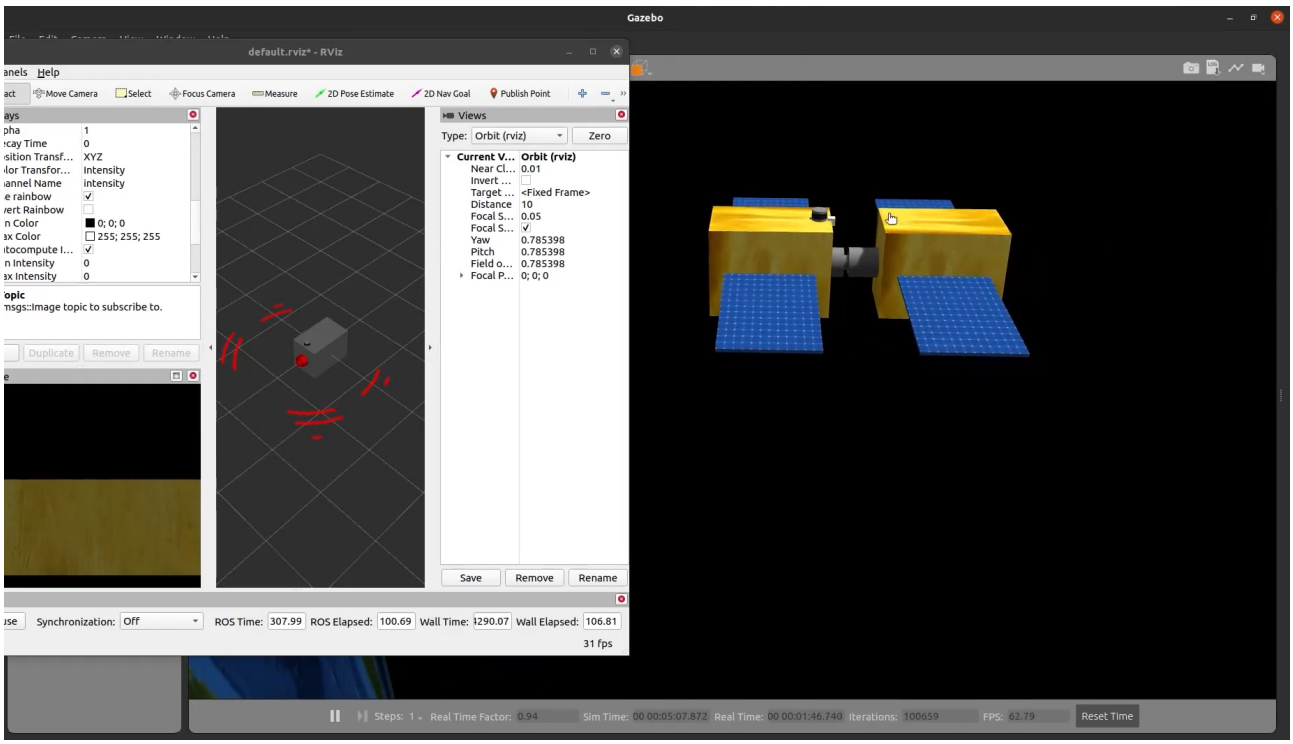

Figure 4.4: Docking Successful

Control System	Docking Accuracy Error (Mean ± SD,cm)	Docking Time (Mean ± SD,seconds)
PID	1.9 ± 0.4	21.2 ± 1.4
LQR	1.05 ± 0.21	26.3 ± 1.1
Non-linear	0.61 ± 0.17	30.8 ± 2.7

Table 4.1: Comparative Analysis of Control Systems in Two-Plane Docking

The table 4.1 compares the performance of the PID, LQR, and Non-linear controllers in the two-plane docking scenario in terms of mean docking accuracy error and mean docking time. In this context, the mean docking accuracy error refers to the average offset from the center of the docking platform on the chaser satellite to the target docking position. The PID controller, with a mean docking accuracy error of 1.9 cm and a mean docking time of 21.2 seconds, provides the fastest response time but exhibits the highest variability, resulting in the least precision. The LQR controller, with a mean docking accuracy error of 1.05 cm in 26.3 seconds, offers a balanced trade-off between speed and precision, making it suitable for missions that require both timely execution and moderate accuracy. The Non-linear controller, designed to minimize mean docking accuracy error, achieves the highest precision with an error of just 0.61 cm, though it requires 30.8 seconds of mean docking time to complete the maneuver. Despite its long docking time, the Non-linear controller works well for situations demanding high accuracy. This analysis highlights each controller's strengths and limitations, guiding the decision based on the unique mission requirements.

4.2 Three-Plane Docking

This section shows the results from the three-plane docking tests, in which the chaser satellite aligns sequentially along the Z, Y, and X axes to accomplish successful docking. The method and results of this docking maneuver are shown in the following figures.

Figure 4.5 shows the initial scenario of the three-plane docking process. Unlike the two-plane docking, the chaser satellite begins by aligning along the Z-axis. After establishing Z-axis alignment, the satellite adjusts its course to correct Y-axis alignment, as shown in Figure 4.6. After aligning the Z and Y axes, the chaser aligns along the X-axis, completing the docking procedure as shown in Figure 4.7.

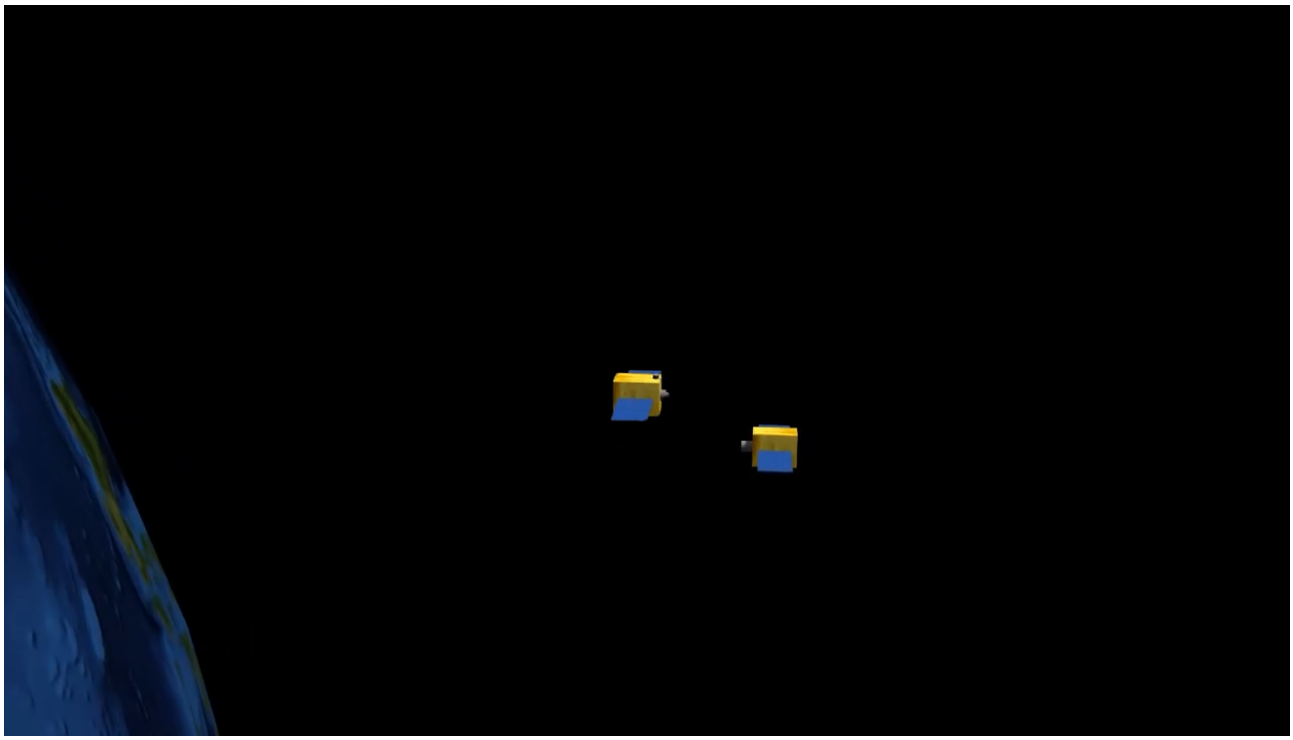

Figure 4.5: Three-Plane Scenario

After a successful three-plane docking maneuver, the three control systems are compared for efficiency, same as previous scenario. The table 4.2 compares the mean docking accuracy error and mean docking time

Control System	**Docking Accuracy Error (Mean ± SD,cm)**	**Time to Dock (Mean ± SD,seconds)**
PID	1.6 ± 0.4	23.9 ± 1.4
LQR	0.91 ± 0.3	28.2 ± 1.1
Non-linear	0.52 ± 0.2	36.4 ± 2.9

Table 4.2: Comparative Analysis of Control Systems in Three-Plane Docking

required to dock for each control system. The PID controller reached a mean docking accuracy error of 1.6

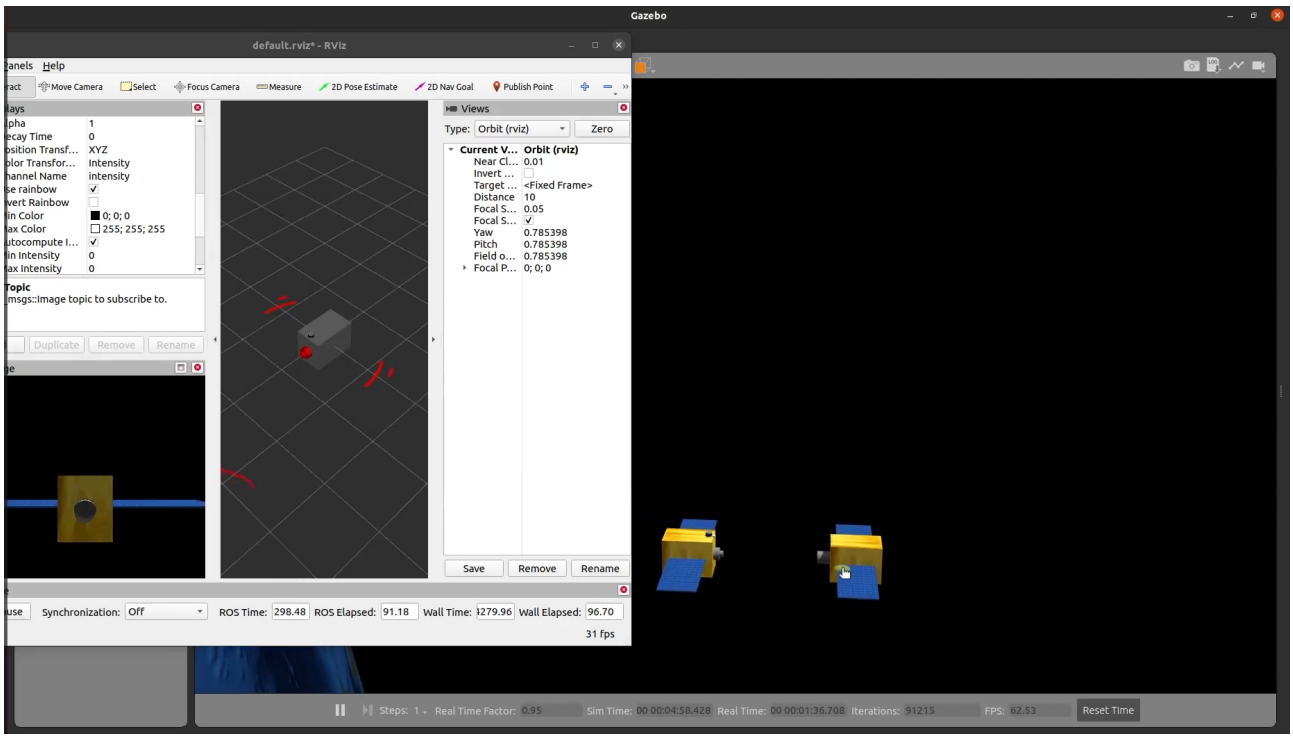

Figure 4.6: Docking Process

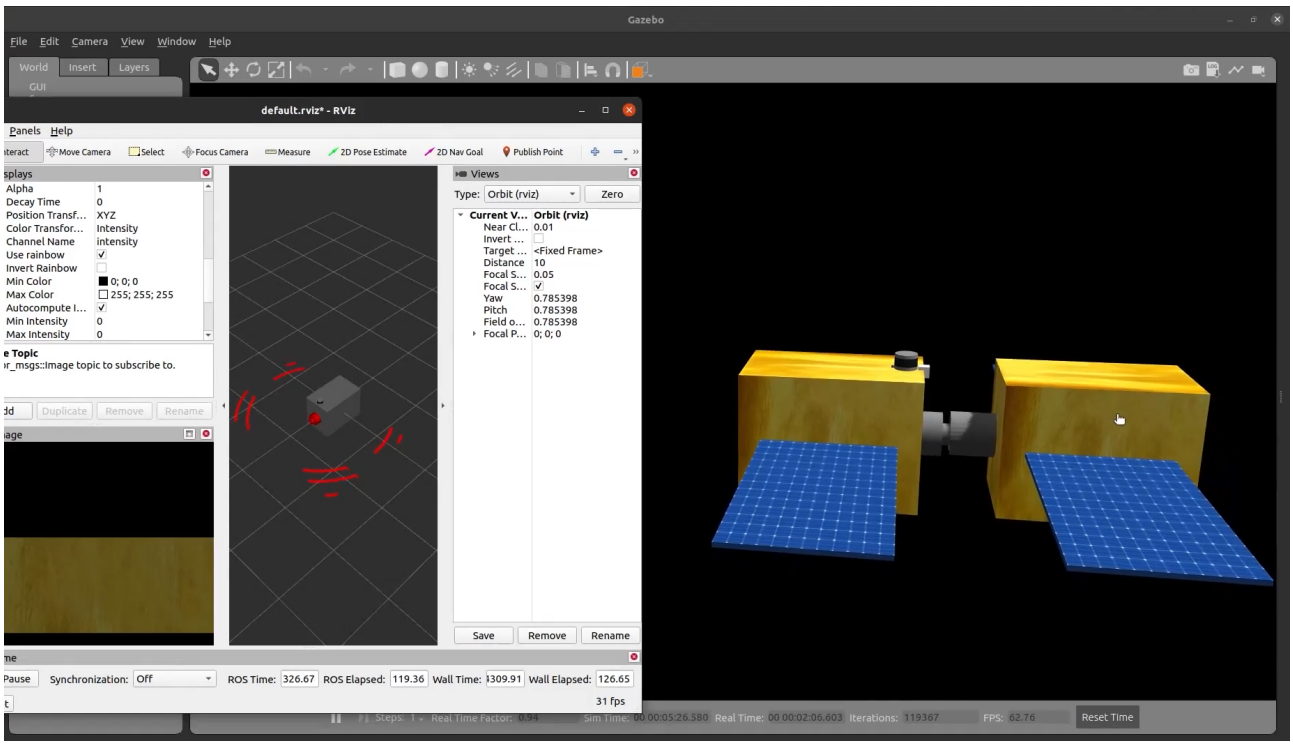

Figure 4.7: Docking Successful

cm in 23.9 seconds, making it the quickest but also the most variable in accuracy. The LQR controller, with an estimated mean docking accuracy error of 0.91 cm and mean docking duration of 28.2 seconds, strikes a reasonable compromise between speed and precision. The Non-linear controller, though slower with a mean docking time of 36.4 seconds, achieves the mean docking accuracy error of 0.52 cm. This comparison highlights

the trade-offs between speed and accuracy, with the Non-linear controller being the best option for precision-critical missions and the PID controller possibly preferred in time-sensitive situations. This investigation gives useful information about each control system's strengths and limitations in dealing with the additional problems given by three-plane docking. A suitable controller will be chosen based on the mission's unique needs, with a focus on balancing speed and precision.

4.3 Docking with Orientation Change

In this section, we provide the results of docking experiments in which the chaser satellite changes orientation, before performing the docking operation. This scenario is slightly harder than the previous docking scenarios since the satellite must first adjust its orientation, if needed, before aligning along the Z, Y, and X axes to complete the docking operation.

Figure 4.8 shows the initial situation for docking with orientation adjustment. The chaser satellite begins by changing its orientation to coincide with the target docking port. Unlike the two- and three-plane docking scenarios, this approach prioritizes orientation correction before axis alignment. Figure 4.9 shows the rviz, before the orientation change.After correcting the orientation,the satellite begins docking by aligning the Z-axis first, then the Y-axis, and lastly the X-axis. Figure 4.10 shows successful docking after the orientation change.

After a successful docking maneuver with orientation change, the three control systems are compared for efficiency same as previous scenario.

Control System	**Docking Accuracy Error (Mean ± SD,cm)**	**Time to Dock (Mean ± SD,seconds)**
PID	2.0 ± 0.38	22.1 ± 1.9
LQR	1.1 ± 0.21	26.5 ± 2.3
Non-linear	0.624 ± 0.18	32.1 ± 2.8

Table 4.3: Comparative Analysis of Control Systems in Docking with Orientation Change

The table 4.3 compares the mean docking accuracy error and mean docking time for each control system during the orientation change scenario. The PID controller achieved a mean docking accuracy error of 2 cm in 22.1 seconds, making it the quickest but also the most variable in precision. The LQR controller, with an estimated mean docking accuracy error of 1.1 cm and a mean docking duration of 26.5 seconds, provides a good combine of speed and precision. The Non-linear controller, though slower with a mean docking time of 32.1 seconds, achieves the mean docking accuracy error of 0.624 cm. This comparison highlights the trade-offs between speed and accuracy, with the Non-linear controller being the best choice for precision-essential missions involving complicated orientation changes, and the PID controller possibly better in cases where time is

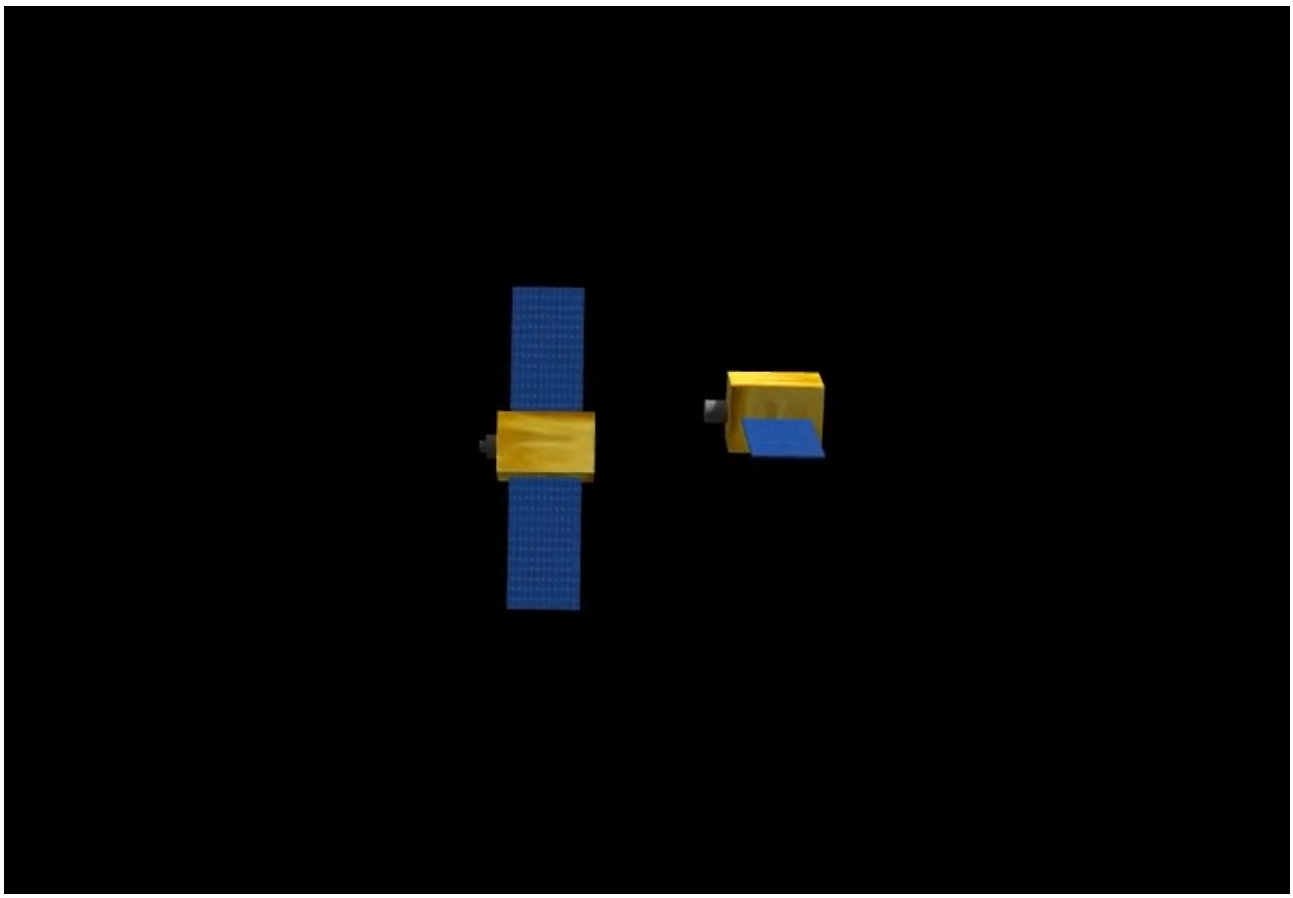

Figure 4.8: Orientation Change Scenario

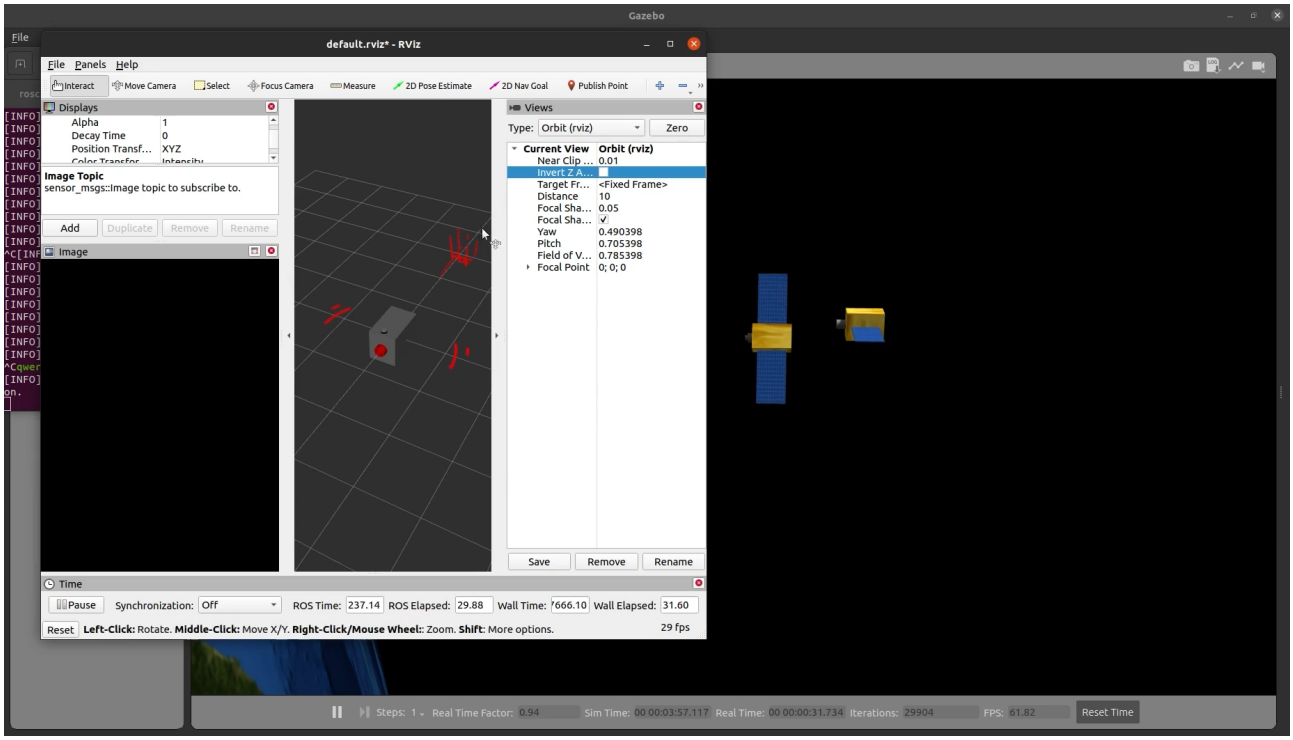

Figure 4.9: Docking Process

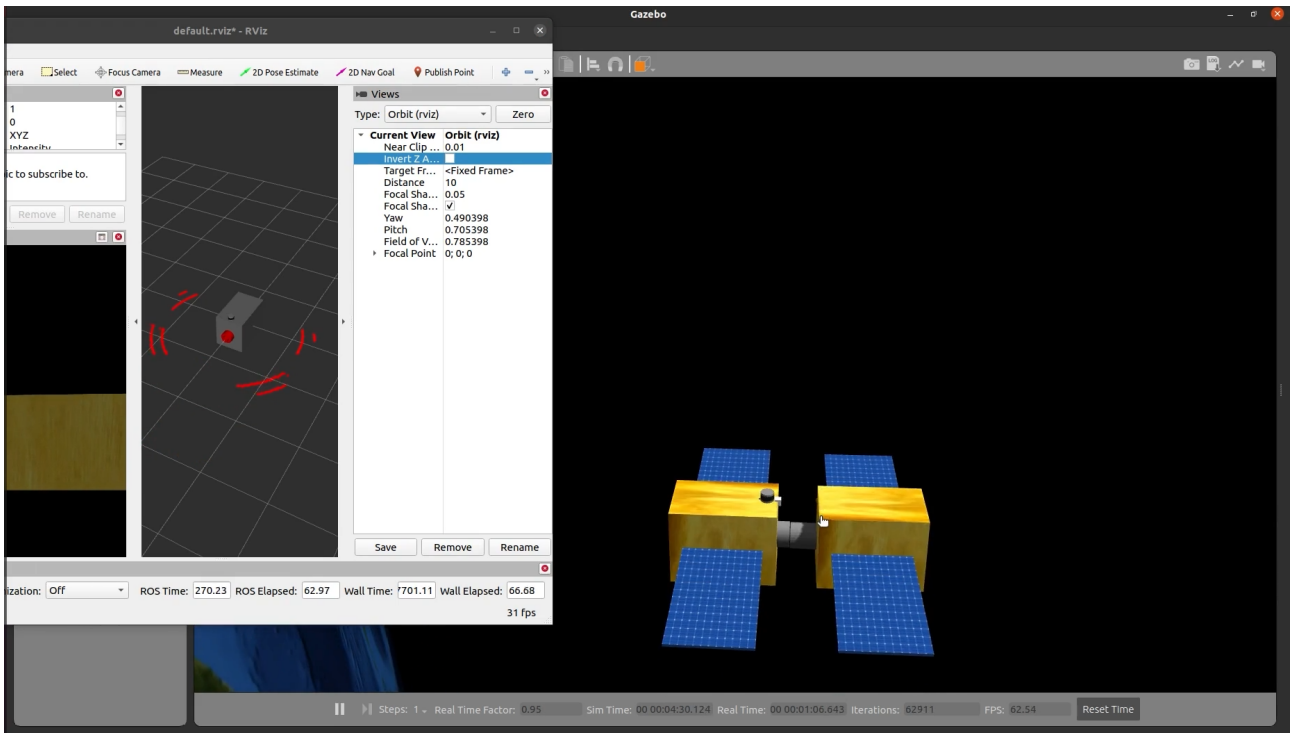

Figure 4.10: Docking Successful

an essential factor.

This examination gives important insights into each control system's capability in managing docking circumstances that necessitate orientation modifications. The control system used will be determined by the mission's unique objectives, which must balance the need for quick reaction with accurate orientation control.

4.4 Collision Avoidance

This section covers the results from the collision avoidance tests, in which the chaser spacecraft has recognize and prevent space debris during the docking process. This scenario examines the satellite's ability to adapt easily to unexpected challenges while remaining on mission with minimal disruption.

Figure 4.11 shows the initial scenario, where space debris is near the chaser satellite, posing a possible hazard to docking operations. The identification of debris is important to the successful completion of the collision avoidance action. In Figure 4.12, the Velodyne LiDAR data presented in RViz shows the detection of space debris. Once the debris has been recognized, the collision avoidance programs is activated, which calculates the appropriate trajectory adjustments to avoid the collision. Figure 4.13 shows the satellite's movement as it ascends slightly to avoid collision with space debris. The satellite's ability to execute this action fast and correctly is vital to the mission's integrity. After successfully avoiding the collision, the satellite returns to its docking operation, as seen in Figure 4.14.

After a successful collision avoidance, the performance of this operation is examined by number of tests.

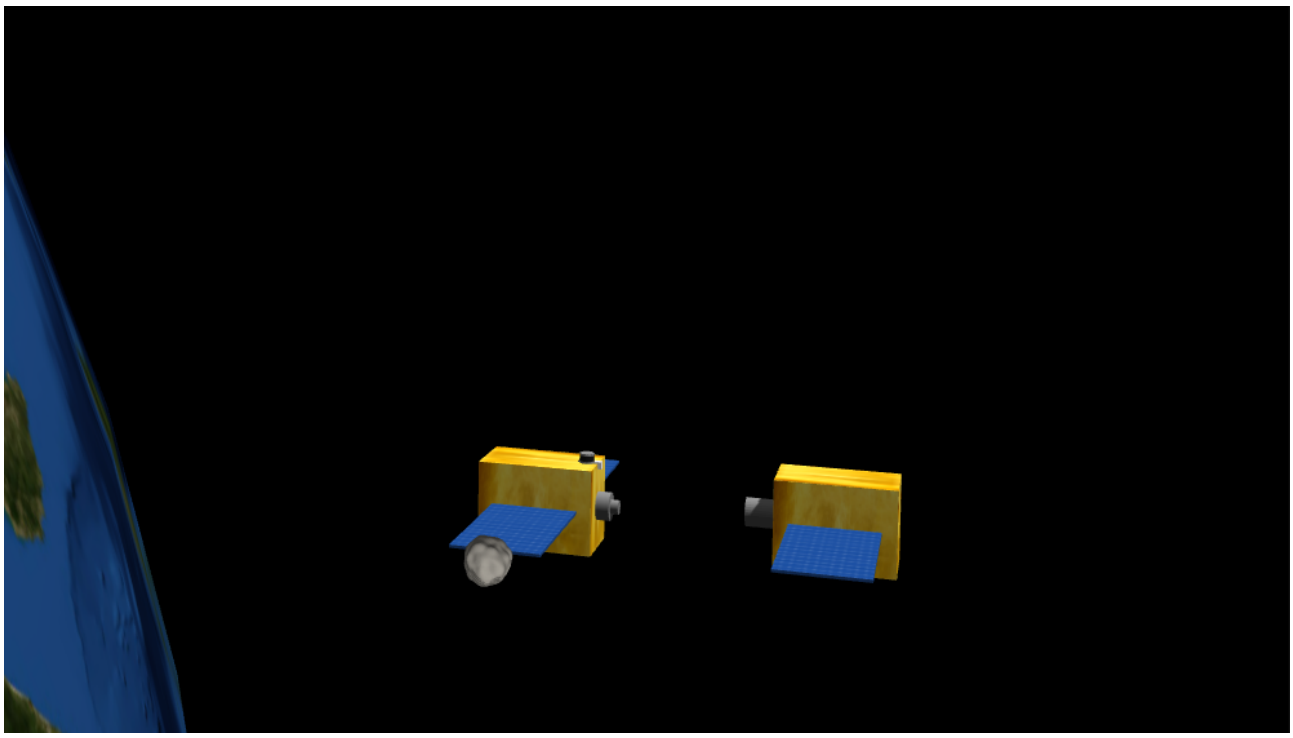

Figure 4.11: Space Debris near satellite

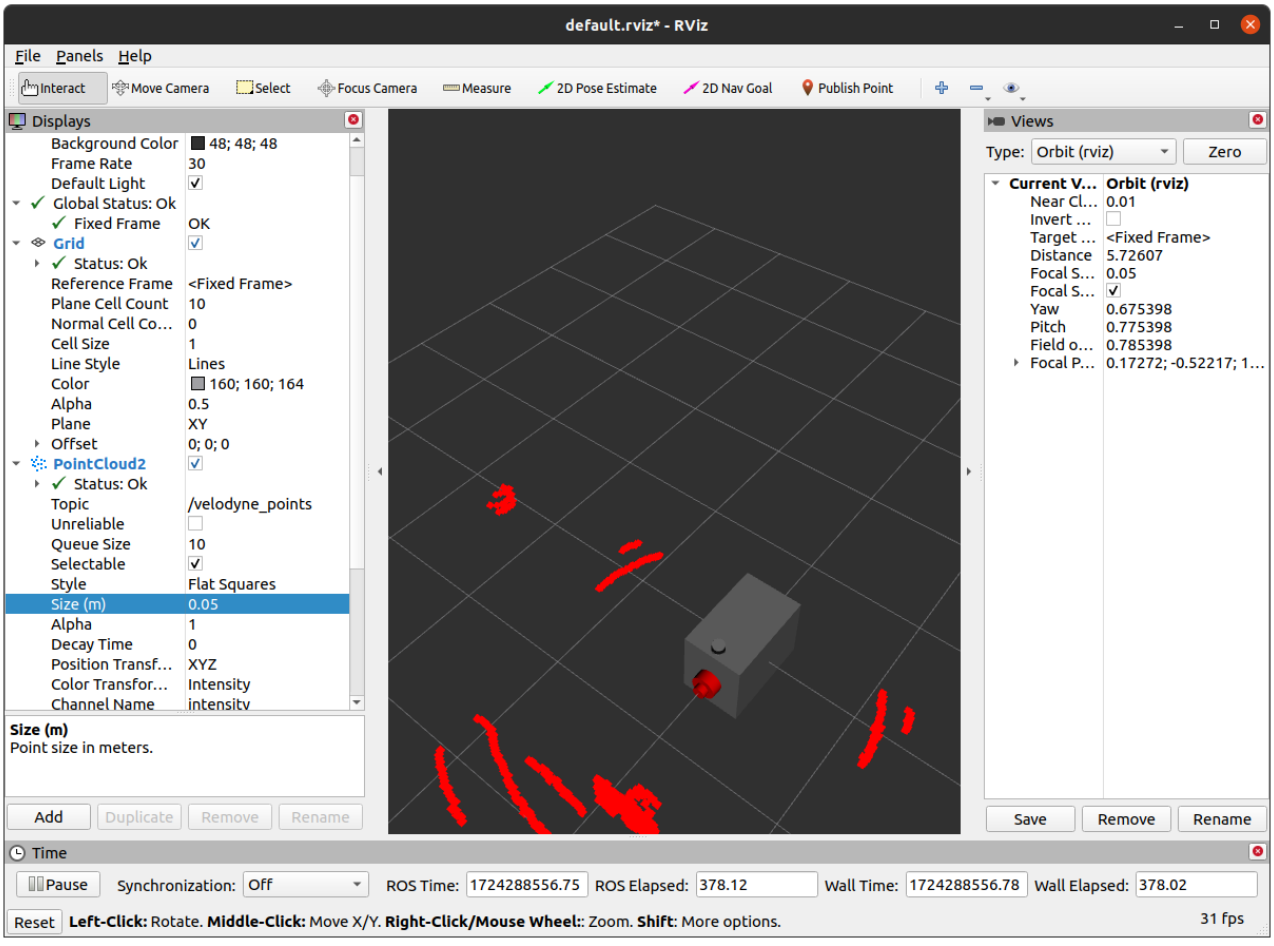

Figure 4.12: Object Detection

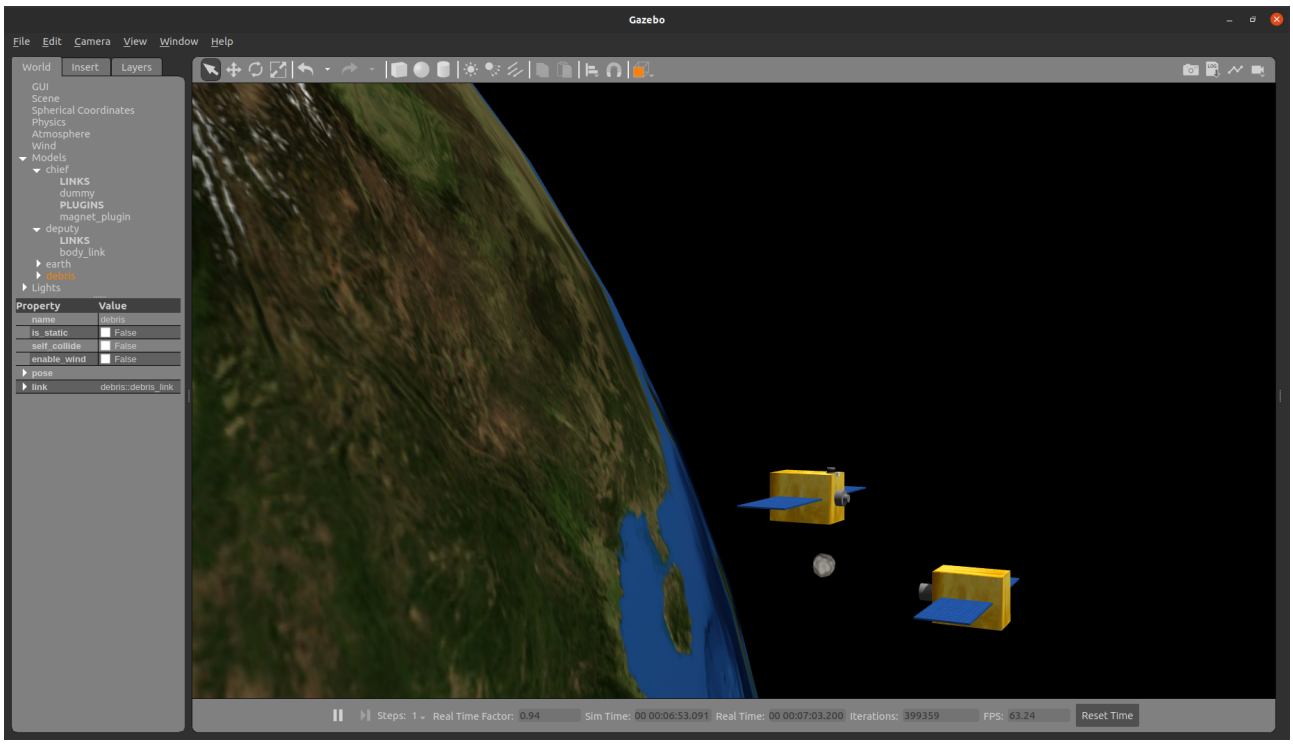

Figure 4.13: Collision Avoidance

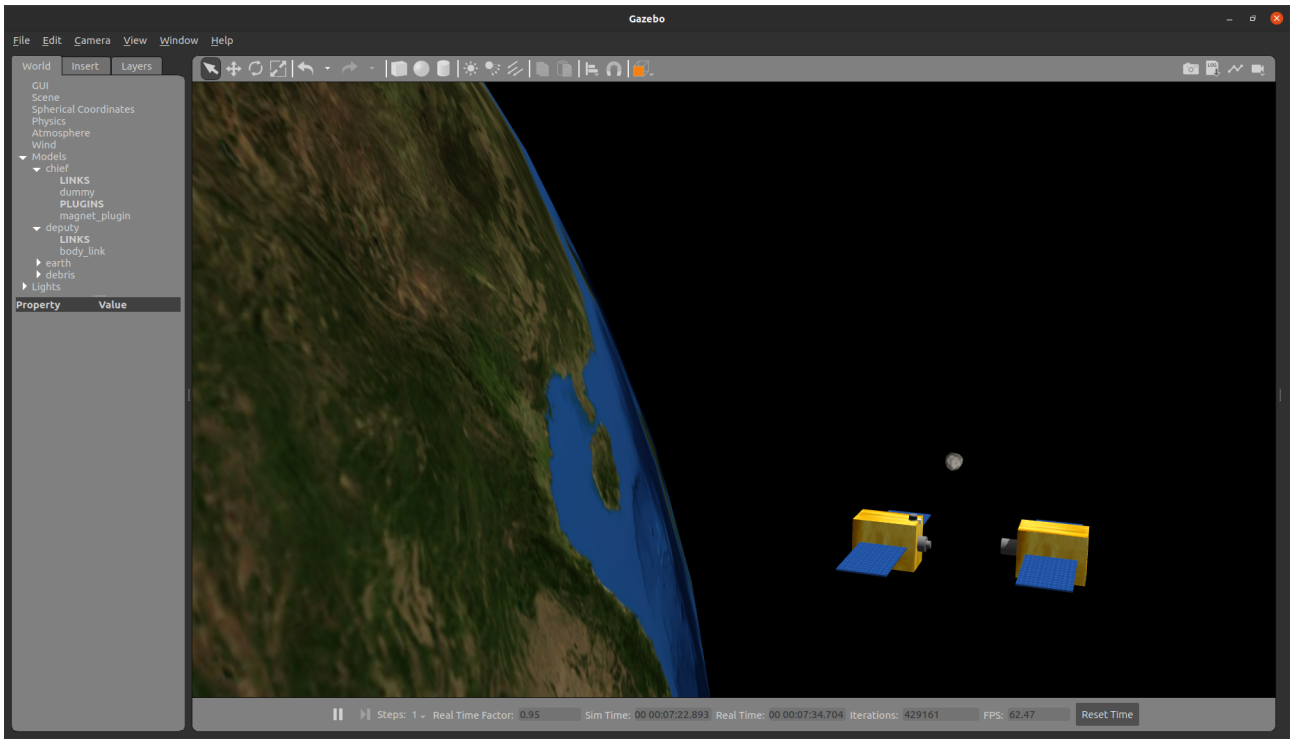

Figure 4.14: Return to Docking

From the Table 4.4, the collision detection accuracy was average of 96%, with minimal false positives, indicating that the system consistently recognized potential hazards. It took around of 10.3 seconds to identify and respond to the debris, including the time between initial detection and thruster activation. Finally, the satellite returned to the docking procedure in around 12.2 seconds following the collision avoidance operation,

Performance Metric	**Value**
Collision Detection Accuracy (%)	96 ± 2
Time to Detect and Respond (seconds)	10.3 ± 1.1
Return to Docking Time (seconds)	12.2 ± 1.4

Table 4.4: Collision Avoidance System Performance

validating the system's ability to resuming operations immediately.

This investigation shows that the thruster-based collision avoidance technology is successful in protecting the satellite during docking. Its great accuracy, quick response, and quick return to docking demonstrate its dependability and efficiency, making it important to the satellite's autonomous operations.

The mission requirements should be taken into consideration while selecting a control system for autonomous docking operations. Despite its long docking time, the Non-linear controller is the best solution for jobs requiring high accuracy. The LQR controller is ideal for missions requiring a balance of speed and accuracy. The PID controller, though the quickest, should be utilized in situations when speed is critical and accuracy is less important.

The collision avoidance technology, which uses thruster movements, proved to be both dependable and efficient, assuring mission safety without significant disturbance to the docking operation.

These findings help to advance the development of robust and dependable autonomous docking systems, which might have uses in future space missions where accuracy and safety are important.

5 Discussion and Conclusion

5.1 Overview

This Research aimed to create an enhanced autonomous docking system for spacecraft by using innovative simulation tools such as Gazebo and ROS (Robot Operating System) and incorporating strong sensors such as Velodyne LiDAR and depth cameras. The study also looked at additional control systems, such as proportional-integral-derivative (PID) controllers, linear-quadratic regulators (LQR), and non-linear controllers, to improve the accuracy and reliability of docking operations in space. The aim of this study was to provide more accurate predictions of relative orbital motion by optimizing the Hill-Clohessy-Wiltshire (HCW) equations.

The simulation environment developed for this research permitted a series of controlled experiments that closely replicated real-world space conditions. These simulations were useful in determining the performance of the proposed system under various operating situations, such as two-plane docking, three-plane docking, docking with orientation change, and collision avoidance. The results showed that combining modern sensors with strong control algorithms improves the accuracy and safety of autonomous docking operations in space.

5.2 Key Findings

The results of this study highlight many important findings:

1. **Control System Performance:** The comparison of PID, LQR, and non-linear controllers showed that each has unique strengths and limitations based on the mission's needs. The PID controller, while fastest, has the most fluctuation in docking accuracy. The LQR controller offered a fair trade-off between speed and accuracy, making it ideal for missions that required both. The non-linear controller, despite being the slowest, had the best accuracy, making it optimal for applications requiring accuracy.

2. **Sensor Integration:** The combination of Velodyne LiDAR and depth cameras greatly increased the system's spatial awareness, which is vital to accurate docking procedures. The sensor fusion algorithms created in this work effectively processed and interpreted data in real time, allowing the system to adapt dynamically to changing circumstances. This integration was especially useful in collision avoidance situations, where accurate and quick obstacle recognition is critical.

3. **Simulation and Real-World Application:** While the simulation environment provided a highly controlled environment for testing the system, it also pointed out the challenges of transitioning from simulation to real-world application. The difference between simulated and real-world scenarios, such as the

uncertain characteristics of space conditions and the possibility of sensor drift, poses substantial issues that must be addressed in future study.

4. **Collision Avoidance:** The thruster-based collision avoidance method proved to be both dependable and efficient, assuring the satellite's safety without significant disruption to docking procedures. The system's capacity to recognize and respond to possible collisions in an average of 10.3 seconds illustrates its usefulness in high-risk situations.

5.3 Contribution to Research Field

This research contributes to our growing knowledge of autonomous spacecraft operations by:

1. **Advancing Sensor Fusion Techniques:** This work improves autonomous systems' ability to see and navigate their environment by integrating Velodyne LiDAR and depth cameras. The sensor fusion methods proposed here might be used to a variety of space applications, increasing the safety and dependability of autonomous operations.

2. **Optimizing Control Strategies:** The comparative examination of various control systems reveals important information about their applicability in various mission conditions. This research provides a framework for determining the best control approach based on mission-specific requirements while balancing speed, accuracy, and computing complexity.

3. **Providing Insights for Real-World Applications:** This research provides important insights into the challenges and complications involved in implementing autonomous docking systems to actual satellites under real-world conditions. The insights and approaches presented here can serve as an outline for future research that bridges the gap between simulation and actual space missions, improving the practicality and dependability of autonomous systems in space operations.

5.4 Limitation of the System

Despite the positive results, this study has numerous limitations, which must be acknowledged:

1. **Reliance on Simulation:** The findings may not fully apply to real-life situations due to their reliance on simulation systems like as ROS and Gazebo. Sensor noise, ambient disturbances, and hardware limits can all have an influence on the system's performance in space. Therefore, while the simulation results are intriguing, more validation in real-world conditions is required.

2. **Control Strategy Adaptability:** The control methods examined in this study were effective in the specific conditions simulated, but their applicability to a larger variety of scenarios is uncertain. For Instance, Despite its excellent accuracy, the non-linear controller may not perform as well in missions requiring quick response times.

3. **Environmental Factors Affecting LiDAR:** As the space is not perfect vacuum, LiDAR can be affected by factors like solar radiation and beam divergence. These issues may reduce the accuracy of distance measurements and object recognition. However, these can be solved with employing adaptive optics, optimize wavelength selection, use advanced modulation techniques, and implement error correction algorithms. While these answers were acknowledged, they were not extensively investigated in this study, suggesting a need for further research and development.

5.5 Future Work

Based on the findings of this research, multiple areas for further investigation have been suggested:

1. **Real-World Testing:** The findings of the research should logically be verified in actual space missions as the next logical step. This might include deploying the system on small-scale missions, such as CubeSats, to collect empirical data on its performance in real-world space settings.

2. **Adaptive Control Systems:** Future study should look at the development of adaptive control systems that can react to changing mission parameters in real time. This would include combining machine learning techniques with standard control methods to develop systems that can learn from their surroundings and improve their performance over time.

3. **Improved Sensor Calibration Techniques:** Autonomous systems in space will require addressing sensor drift and calibration in order to be successful in the long term. Research into advanced calibration approaches, such as machine learning or real-time modification algorithms, might help to reduce the risks associated with sensor errors.

4. **Enhanced Simulation Environment:** Expanding the simulation environment to accommodate more complicated situations, such as multi-satellite operations or connections with non-cooperative objects, would give further information about the system's capabilities and limitations. Incorporating more realistic environmental variables, such as fluctuating illumination conditions and cosmic radiation, may further increase simulation accuracy.

5.6 Conclusion

In conclusion, this research successfully demonstrated the potential of advanced autonomous systems for spacecraft docking operations. This research provides the groundwork for safer and more efficient space missions by combining sensor technology with effective control methods. However, the transition from simulation to real-world application remains a significant issue, and further study is required to guarantee these systems can work successfully in the unpredictable environment of space. The study's findings provide important insight

to the field of autonomous space operations and establish a foundation for future developments that have the potential to transform how we explore and operate in space.

6 Contributions

My work has been initially accepted as a book chapter to publish in a book titled **"Algorithms for Machine Vision in Navigation and Control"** to be published by **Springer**, and the acceptance process is **"peer-reviewed"**.

A Appendix

A.1 Gantt Chart

The project timeline is represented in the Gantt chart shown in the Figure A.1.

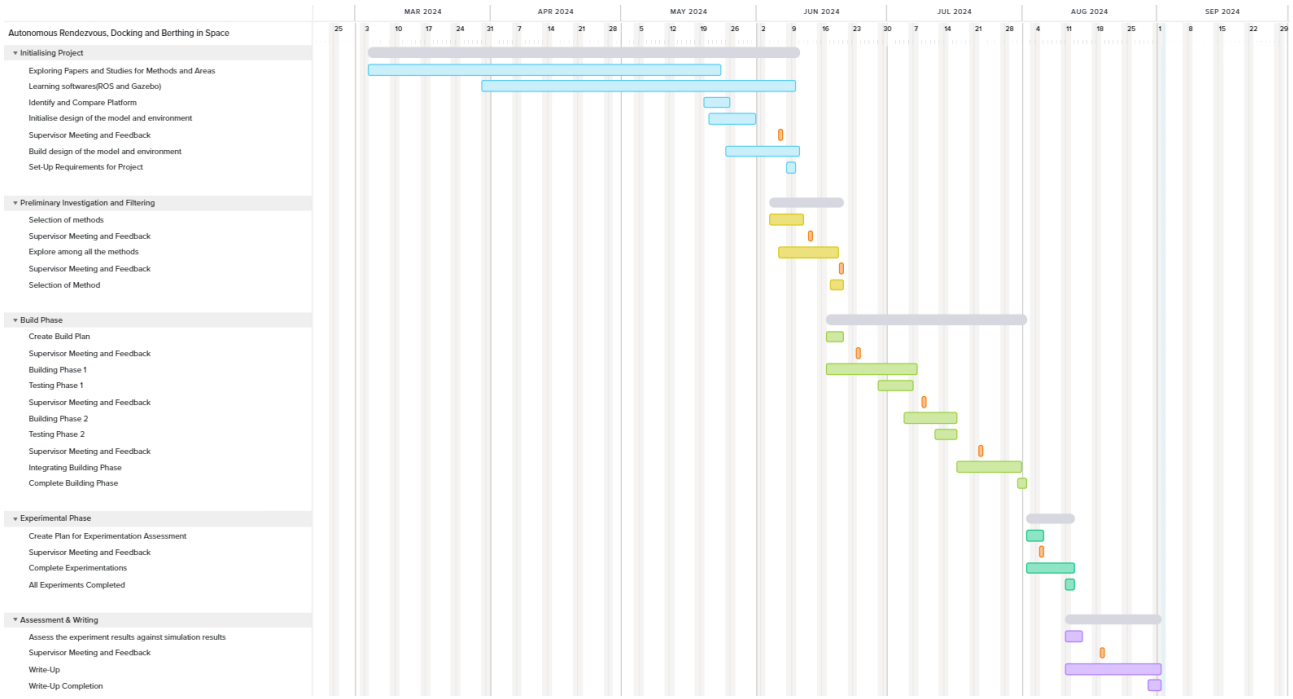

Figure A.1: Gantt Chart

This Gantt chart illustrates the completed timeline for this project. It systematically details the completion of tasks across various phases, including Project Initialization, Preliminary Investigation and Filtering, Build Phase, Experimental Phase, and Assessment & Writing. Each phase is clearly marked with specific start and end dates, with tasks visually distinguished to indicate their completion, demonstrating adherence to the scheduled timeline. Supervisor meetings and feedback throughout these phases are noteworthy, since they have played an important role in guiding the project's iterative development and improvement.

A.2 Risk Assessment

The table below details and ranks the project risks. Severity varies from 1 to 5, where 5 is very likely and Impact varies from 1 to 5, where 5 is major impact.

Hazards Identified *(state the potential harm)*	Existing Control Measures	S Severity	L Likelihood	Risk Level SxL	Additional Control Measures	S	L	Risk Level SxL
Unplanned Additional work to the project.	Regular weekly reviews and updates of the plan.	3	2	6	None.	3	2	6
Unresolved project conflicts not resolved in a timely manner.	Held regular project meetings and looked out for conflicts. Reviewed the project plan or potential areas of conflict.	3	3	9	When aware, immediately escalated to the Project supervisor and gained assistance from the supervisor to resolve the conflict.	2	2	4
Errors that conflicted with the project plan.	Always had some extra time for error solving if it occurs.	3	3	9	Tried to get advice from a project supervisor or from a person who is expert in that field.	2	2	2
The project's scope expanded beyond the initial plan that leads to delays.	Scheduled periodic review meetings to assess the project's progress against the initial plan.	3	3	9	Established a clear scope baseline early in the project lifecycle. Any proposed changes to the scope is compared against this baseline to determine their impact on the project timeline and resources.	3	1	3
Resource limitations.	Regularly analysed resource allocation against the expanded scope to identify any gaps or constraints.	2	3	6	Prioritised project activities based on resource availability and criticality.	2	2	4
Data availability and quality at risk	Established continuous monitoring and auditing processes for project data to detect and address any irregularities or inconsistencies.	2	2	4	Developed contingency plans to address unexpected data issues. Outlined specific actions to be taken in case of data unavailability or poor quality.	2	1	2
Got Sick during Project Time	Having a buffer and extra time solves this problem a lot	3	1	3	None.	3	1	3

References

[1] NASA. "Challenges of human space exploration." (2020), available from: https://www.nasa.gov/humans-in-space/why-go-to-space/.

[2] I. A. Nesnas, L. M. Fesq, and R. A. Volpe, Autonomy for space robots: Past, present, and future, *Current Robotics Reports* [online], vol. 2, no. 3 Sep. 2021, pp. 251–263, Sep. 2021, ISSN: 2662-4087. DOI: 10.1007/s43154-021-00057-2. available from: https://doi.org/10.1007/s43154-021-00057-2.

[3] NASA. "Nasa orbital debris program office." (2024), available from: https://orbitaldebris.jsc.nasa.gov/faq/.

[4] Wikipedia. "Velodyne lidar." (2024), available from: https://en.wikipedia.org/wiki/Velodyne_Lidar.

[5] Ouster. "Velodyne lidar vlp-16." (2024), available from: https://ouster.com/products/hardware/vlp-16.

[6] Wikipedia. "Hill-clohessy-wiltshire equation." (2023), available from: https://en.wikipedia.org/wiki/Clohessy%E2%80%93Wiltshire_equations.

[7] Wikipedia. "Collision avoidance (spacecraft)." (2024), available from: https://en.wikipedia.org/wiki/Collision_avoidance_%28spacecraft%29.

[8] S. M. E. Pegah Abdollahzadeh, "Automatic orbital docking with tumbling target using sliding mode control," in *Advances in Space Research Volume 67, Issue 5,*, 2021, pp. 1506–1525.

[9] Q. Li, J. Yuan, B. Zhang, and C. Gao, "Model predictive control for autonomous rendezvous and docking with a tumbling target," in *Aerospace Science and Technology Volume 69*, 2017.

[10] R. T. Howard, T. C. Bryan, L. L. Brewster, and J. E. Lee, "Proximity operations and docking sensor development," in *2009 IEEE Aerospace conference*, 2009, pp. 1–10. DOI: 10.1109/AERO.2009.4839574.

[11] H. Hinkel, J. J. Zipay, M. Strube, and S. Cryan, "Technology development of automated rendezvous and docking/capture sensors and docking mechanism for the asteroid redirect crewed mission," in *2016 IEEE Aerospace Conference*, 2016, pp. 1–8. DOI: 10.1109/AERO.2016.7500637.

[12] S.-R. Kim, H.-J. Jo, J.-H. Kim, and J.-Y. Park, Development of an autonomous docking system for autonomous surface vehicles based on symbol recognition, *Ocean Engineering* [online], vol. 283 2023, p. 114 753, 2023, ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2023.114753. available from: https://www.sciencedirect.com/science/article/pii/S002980182301137X.

[13] F. Stesina, S. Corpino, C. Novara, and S. Russo, Docking manoeuvre control for cubesats, *The Journal of the Astronautical Sciences* [online], vol. 69, no. 2 Apr. 2022, pp. 312–334, Apr. 2022, ISSN: 2195-0571. DOI: 10.1007/s40295-022-00307-1. available from: https://doi.org/10.1007/s40295-022-00307-1.

[14] J. Barr, M. Uney, D. Clark, *et al.*, "A multi-sensor inference and data fusion method for tracking small, manoeuvrable maritime craft in cluttered regions," English, in *Proceedings of the 3rd IMA on Mathematics in Defence*, Oct. 2013.

[15] T. Phisannupawong, P. Kamsing, P. Tortceka, and S. Yooyen, "Vision-based attitude estimation for spacecraft docking operation through deep learning algorithm," in *2020 22nd International Conference on Advanced Communication Technology (ICACT)*, 2020.

[16] A. J. Choi, J. Park, and J.-H. Han, Automated aerial docking system using onboard vision-based deep learning, *Journal of Aerospace Information Systems* [online], vol. 19, no. 6 2022, pp. 421–436, 2022. DOI: 10.2514/1.I011053. available from: https://doi.org/10.2514/1.I011053.

[17] Y. L. Wenwen Liu and R. Bucknall, Filtering based multi-sensor data fusion algorithm for a reliable unmanned surface vehicle navigation, *Journal of Marine Engineering & Technology* [online], vol. 22, no. 2 2023, pp. 67–83, 2023. DOI: 10.1080/20464177.2022.2031558. available from: https://doi.org/10.1080/20464177.2022.2031558.

[18] F. Jia, M. Afaq, B. Ripka, Q. Huda, and R. Ahmad, *Vision- and lidar-based autonomous docking and recharging of a mobile robot for machine tending in autonomous manufacturing environments*, 2023. DOI: 10.3390/app131910675. available from: https://doi.org/10.3390/app131910675.

[19] Ø. Volden, A. Stahl, and T. I. Fossen, Vision-based positioning system for auto-docking of unmanned surface vehicles (usvs), *International Journal of Intelligent Robotics and Applications* [online], vol. 6, no. 1 Mar. 2022, pp. 86–103, Mar. 2022, ISSN: 2366-598X. DOI: 10.1007/s41315-021-00193-0. available from: https://doi.org/10.1007/s41315-021-00193-0.

[20] L. A. M. Christensen, *Multi-sensor Data Fusion for Spacecraft Navigation*. Kgs. Lyngby: Technical University of Denmark, 2020, Ph.D. thesis.

[21] T. Fahey, M. Islam, A. Gardi, and R. Sabatini, Laser beam atmospheric propagation modelling for aerospace lidar applications, *Atmosphere* [online], vol. 12, no. 7 2021, 2021, ISSN: 2073-4433. DOI: 10.3390/atmos12070918. available from: https://www.mdpi.com/2073-4433/12/7/918.

[22] J. A. Christian and S. Cryan, "A survey of lidar technology and its use in spacecraft relative navigation," in *AIAA 2013-4641*, 2013. DOI: https://doi.org/10.2514/6.2013-4641.

[23] J. Park and J. Kim, "Autonomous docking of an unmanned surface vehicle based on reachability analysis," in *20th International Conference on Control, Automation and Systems*, 2020.

[24] J. W. Jooho Lee and N. Kim, "Vision and 2d lidar based autonomous surface vehicle docking for identify symbols and dock task in 2016 maritime robotx challenge," in *2017 IEEE Underwater Technology*, 2017.

[25] R. Miyazaki, R. Jiang, H. Paul, K. Ono, and K. Shimonomura, "Airborne docking for multi-rotor aerial manipulations," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4708–4714. DOI: 10.1109/IROS.2018.8594513.

[26] D. Athauda, A. Banerjee, S. Satpute, A. Agha-Mohammadi, and G. Nikolakopoulos, Intelligent motion planning for collision free autonomous docking of satellite emulation platform using reinforcement learning, *IFAC-PapersOnLine* [online], vol. 56, no. 2 2023, pp. 3354–3359, 2023, 22nd IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2023.10.1481. available from: https://www.sciencedirect.com/science/article/pii/S240589632301889X.

[27] E. Anderlini, G. G. Parker, and G. Thomas, *Docking control of an autonomous underwater vehicle using reinforcement learning*, 2019. DOI: 10.3390/app9173456. available from: https://doi.org/10.3390/app9173456.

[28] J. Scharnagl, L. Srinivasan, K. Ravandoor, and K. Schilling, Autonomous collision avoidance for rendezvous and docking in space using photonic mixer devices, *IFAC-PapersOnLine* [online], vol. 48, no. 9 2015, pp. 239–244, 2015, 1st IFAC Workshop on Advanced Control and Navigation for Autonomous Aerospace Vehicles ACNAAV'15, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.08.090. available from: https://www.sciencedirect.com/science/article/pii/S240589631500957X.

[29] R. Zappulla II, H. Park, J. Virgili-Llop, and M. Romano, "Experiments on autonomous spacecraft rendezvous and docking using an adaptive artificial potential field approach," in *26th AAS/AIAA Space Flight Mechanics Meeting At: Napa, CA, Volume: AAS 16-459*, 2016.

[30] F. WANG, X. ZHU, Z. ZHOU, and Y. TANG, Deep-reinforcement-learning-based uav autonomous navigation and collision avoidance in unknown environments, *Chinese Journal of Aeronautics* [online], vol. 37, no. 3 2024, pp. 237–257, 2024, ISSN: 1000-9361. DOI: https://doi.org/10.1016/j.cja.2023.09.033. available from: https://www.sciencedirect.com/science/article/pii/S1000936123003448.

[31] D. Chikurtev, "Mobile robot simulation and navigation in ros and gazebo," in *International Conference Automatics and Informatics (ICAI)*, 2020.

[32] J. Millan-Romera, J. Acevedo, Á. Rodríguez Castaño, H. León, C. Capitán, and A. Ollero, "A utm simulator based on ros and gazebo," in *Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*, 2019.

[33] K. Takaya, T. Asai, V. Kroumov, and F. Smarandache, "Simulation environment for mobile robots testing using ros and gazebo," in *20th International Conference on System Theory, Control and Computing (ICSTCC)*, 2016.

[34] ROS. "Setting up robot simulation." (2024), available from: https://docs.ros.org/en/humble/Tutorials/Advanced/Simulators/Gazebo/Gazebo.html.

[35] ROS. "Simulator gazebo." (2018), available from: https://wiki.ros.org/simulator_gazebo/Tutorials.

[36] Gazebo. "Gazebo tutorials." (2024), available from: https://classic.gazebosim.org/tutorials.

[37] YouTube. "Ros and gazebo tutorials." (2023), available from: https://www.youtube.com/@ArticulatedRobotics/featured.

[38] Udemy. "Ros basics: Program robots!" (2018), available from: https://www.udemy.com/course/ros-basics-program-robots/.

[39] C. AI. "Ros basics in 5 day." (2022), available from: https://app.theconstruct.ai/courses/55.

[40] R. Mengacci, G. Zambella, G. Grioli, D. Caporale, M. G. Catalano, and A. Bicchi, An Open-Source ROS-Gazebo toolbox for simulating robots with compliant actuators, en, *Front Robot AI*, vol. 8 Aug. 2021, p. 713 083, Aug. 2021.

[41] J. L. Ramón, J. Pomares, and L. Felicetti, Task space control for on-orbit space robotics using a new ros-based framework, *Simulation Modelling Practice and Theory* [online], vol. 127 2023, p. 102 790, 2023, ISSN: 1569-190X. DOI: https://doi.org/10.1016/j.simpat.2023.102790. available from: https://www.sciencedirect.com/science/article/pii/S1569190X23000679.

[42] S. Bhadani, S. R. Dillikar, O. N. Pradhan, *et al.*, A ros-based simulation and control framework for in-orbit multi-arm robot assembly operations, *ASTRA 2023: 17th Symposium on Advanced Space Technologies in Robotics and Automation* [online] 2023, 2023. available from: https://dspace.lib.cranfield.ac.uk/items/57024ec8-14f5-4a3f-94cb-3ae282d6f7f4.

[43] Github. "Velodyne lidar simulator." (2021), available from: https://github.com/lmark1/velodyne_simulator.

[44] Wikipedia. "Newton laws of motion." (2024), available from: https://en.wikipedia.org/wiki/Newton%27s_laws_of_motion.

[45] P. LibreText. "Rotating reference frames." (2024), available from: https://phys.libretexts.org/Bookshelves/University_Physics/Mechanics_and_Relativity_(Idema)/07%3A_General_Rotational_Motion/7.02%3A_Rotating_Reference_Frames#:~:text=Rotating%20reference%20frames%20are%20not,application%20of%20a%20net%20force..

[46] ensatellite. "Hill clohessy wiltshire equations." (2024), available from: https://ensatellite.com/hills-equations/.

[47] Wikipedia. “Proportional–integral–derivative controller.” (2024), available from: https://en.wikipedia.org/wiki/Proportional%E2%80%93integral%E2%80%93derivative_controller.

[48] Wikipedia. “Linear–quadratic regulator.” (2024), available from: https://en.wikipedia.org/wiki/Linear%E2%80%93quadratic_regulator.

[49] Wikipedia. “Algebraic riccati equation.” (2023), available from: https://en.wikipedia.org/wiki/Algebraic_Riccati_equation.

[50] ROS. “Rviz.” (2018), available from: https://wiki.ros.org/rviz.

[51] Wikipedia. “Inertial measurement unit.” (2024), available from: https://en.wikipedia.org/wiki/Inertial_measurement_unit.