



Lucas Naury

**Optimised Multi-Arm Robot Motion to Reduce Satellite
Disturbances During In-Orbit Assembly**

School of Aerospace, Transport and Manufacturing
MSc in Robotics

MSc Robotics
Academic Year: 2024 - 2025

Supervisor: Leonard Felicetti
Associate Supervisor: Saurabh Upadhyay, Gilbert Tang
August 2025

**© Cranfield University, 2025. All rights reserved. No part of this
publication may be reproduced without the written permission
of the copyright holder.**

Academic integrity declaration

I declare that:

1. the thesis submitted has been written by me alone.
2. the thesis submitted has not been previously submitted to this university or any other.
3. that all content, including primary and/or secondary data, is true to the best of my knowledge.
4. that all quotations and references have been duly acknowledged according to the requirements of academic research.

I understand that to knowingly submit work in violation of the above statement will be considered by examiners as academic misconduct.

Abstract

The increasing demand for large-scale space structures, such as modular telescopes and solar power stations, requires efficient and reliable In-Orbit Assembly (ISA) operations. Robotic systems play a central role in this process, but their locomotion can induce significant disturbances to the hosting spacecraft because of momentum conservation. These need to be actively compensated, at the expense of scarce and valuable fuel resources. This thesis investigates passive disturbance minimisation through optimised path planning for Cranfield's Multi-Arm Robot for In-Orbit Operations (MARIO).

A literature review identified key approaches to reaction-minimising motion planning, including real-time control, trajectory optimisation, and sampling-based planners. Based on this analysis, the Stochastic Trajectory Optimization for Motion Planning (STOMP) algorithm was implemented within ROS2 and MoveIt2, with a custom cost function penalising joint torques as an indicator of disturbance. The methodology was tested through high-fidelity Gazebo simulations of MARIO performing crawling locomotion on modular structures designed for laboratory experiments at Cranfield University's ASTRA-Lab.

Results demonstrated that mass distribution strongly influences disturbance magnitude, and that optimised trajectories can reduce disturbances by nearly 50% compared to linear baselines. However, issues such as inverse kinematics solver variability, limited repeatability, and collision risks limited practical applicability. The most promising results were obtained when combining Cartesian initial trajectories with STOMP optimisation, although the custom disturbance cost function proved less effective than anticipated.

The findings highlight both the potential and challenges of trajectory optimisation for space robotics, offering insights into the coupling between manipulator motion and spacecraft dynamics. Future work should refine disturbance modelling cost functions, integrate hybrid planning approaches such as RRT* with STOMP, and extend validation to physical experiments with MARIO in the ASTRA-Lab.

Overall, this research contributes to safer and more fuel-efficient in-orbit assembly operations, supporting the development of sustainable space infrastructure.

Keywords:

In-Orbit Assembly, space robotics, manipulator, MARIO, path planning, attitude disturbance, minimum reaction

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dr. Leonard Felicetti, Dr. Saurabh Upadhyay, and Dr. Gilbert Tang, for their invaluable guidance, encouragement, and support throughout the course of this project. Their expertise and insightful feedback were instrumental in shaping both the direction and quality of this thesis.

I am also deeply thankful to Praveen Elavazhagan, PhD, for his generous help with the technical aspects of my work. His advice and hands-on support greatly enhanced my understanding and played a crucial role in overcoming many challenges during this research.

Finally, I wish to extend my appreciation to all those who contributed, directly or indirectly, to the successful completion of this thesis.

Table of Contents

Academic integrity declaration	i
Abstract.....	ii
Acknowledgements.....	iv
List of Figures.....	vii
List of Tables	viii
List of Equations	ix
List of Abbreviations.....	x
1 Introduction	1
1.1 Background.....	1
1.2 Aim.....	1
1.3 Objectives	1
1.4 Outline of the thesis	2
2 Literature review.....	4
2.1 In-Space Servicing, Assembly and Manufacturing.....	4
2.2 Space robotic systems for assembly.....	4
2.3 Planning architecture	8
2.4 Reaction minimising algorithms	9
2.4.1 Real-time control	10
2.4.2 Path optimisation.....	10
2.4.3 Path planning	11
2.4.4 Others	11
2.5 Approach trade-off	12
3 Problem definition.....	13
3.1 Problem	13
3.2 Transcribing the problem in a lab demonstration	14
3.3 Lab experimental setup.....	16
3.3.1 Structure design	17
3.3.2 Docking interface	18
3.3.3 Final setup.....	21
3.4 Simulation setup	22
4 Design methodology	24
4.1 Crawling motion	24
4.2 Path planning implementation.....	27
4.2.1 The STOMP algorithm.....	27
4.2.2 Custom cost function.....	28
4.2.3 Implementation as a MoveIt2 planner	29
4.3 Software architecture	30
4.3.1 MARIO CommandServer	30
4.3.2 Gazebo simulation	32
4.3.3 Crawling code	34

5 Results and Discussion	36
5.1 STOMP constrained planning	37
5.1.1 Experimental setup.....	37
5.1.2 Findings.....	38
5.2 STOMP on the free arm.....	39
5.2.1 Experimental setup.....	39
5.2.2 Findings.....	40
5.3 STOMP on the free arm with modified weights	42
5.3.1 Experimental setup.....	42
5.3.2 Findings.....	42
5.4 STOMP on the free arm with joint targets	45
5.4.1 Experimental setup.....	45
5.4.2 Findings.....	46
5.5 STOMP on the free arm with a cartesian initial trajectory	47
5.5.1 Experimental setup.....	47
5.5.2 Findings.....	48
5.6 Summary	48
6 Conclusion & Future work.....	50
References	52
Appendices	57

List of Figures

Figure 1: Classes of space robots for assembly [12].....	5
Figure 2: Multi-Arm Robot (MAR) for on-orbit large telescope [16].....	6
Figure 3: Multi Arm Relocatable Manipulator (MARM) robot performing assembly tasks [18]	6
Figure 4: MOSAR Walking Manipulator Concept [19]	7
Figure 5: MOSAR Demonstrator setup [19].....	7
Figure 6: MARIO in the ASTRA-Lab.....	8
Figure 7: Hybrid planning architecture composed of 2 layers [22]	9
Figure 8: MARIO crawling on a structure [20].....	13
Figure 9: NASA's Neutral Buoyancy Laboratory [35].....	14
Figure 10: Multi-Arm Robot hanging from a cable [36]	15
Figure 11: ASTRA-Lab floor slope measurements	16
Figure 12: MARIO's air bearing platform	16
Figure 13: Crawling structure unit.....	17
Figure 14: 3 Units in a row.....	18
Figure 15: 3 Units in a L shape.....	18
Figure 16: Docking interface.....	19
Figure 17: Tool changer (left) and UFactory 850 arm attach points (right)	20
Figure 18: Docking tool wiring diagram	20
Figure 19: Built structure	21
Figure 20: Docking interface (docking tool, docking point, full assembly)	21
Figure 21: Gazebo simulation.....	22
Figure 22: Docking in simulation	23
Figure 23: MARIO's crawling cycle.....	25
Figure 24: MARIO crawling on the structure	26
Figure 25: The STOMP algorithm process [39]	28
Figure 26: Complete software architecture in simulation	30
Figure 27: PoseCommand flow diagram	32
Figure 28: Robot links and DetachableJoint.....	33
Figure 29: TaskScheduler sequence diagram	35
Figure 30: Constrained planning for the left arm when docked	38
Figure 31: Compliant docking system [41]	39
Figure 32: Free arm motion (yellow circles).....	40
Figure 33: STOMP on free arm - Structure yaw over time.....	41
Figure 34: Robot and structure position trajectories in space.....	41
Figure 35: Weight distribution (Red: free arm moving, Blue: static elements) ..	42
Figure 36: Robot and structure position trajectories in space.....	43
Figure 37: Right arm shoulder joint values over time	45
Figure 38: Left arm joint values over time	45
Figure 39: Left arm joint values over time when using STOMP with joint targets	47
Figure 40: Structure force diagram.....	57

List of Tables

Table 1: STOMP Free arm - Structure results 40

Table 2: STOMP on the heavy free arm - Structure results..... 43

Table 3: Linear trajectory in cartesian space VS joint space – Structure results
..... 44

Table 4: STOMP on the heavy arm with joint targets – Structure results 46

Table 5: Linear trajectory in cartesian space VS joint space – Structure results
..... 47

Table 6: STOMP on the free arm with initial trajectory - Structure results 48

List of Equations

(4-1)..... 28

(4-2)..... 28

(4-3)..... 28

(4-4)..... 28

(4-5)..... 28

(4-6)..... 29

(6-1)..... 58

(6-2)..... 58

(6-3)..... 58

(6-4)..... 58

(6-5)..... 58

(6-6)..... 58

List of Abbreviations

MARIO	Multi-Arm Robot for In-Orbit Operations
ASTRA-Lab	Advanced Space Technology for Robotics and Astronautics Laboratory
ESA	European Space Agency
NASA	National Aeronautics and Space Administration
ROS2	Robot Operating System 2
ISAM	In-Space Servicing, Assembly and Manufacturing
ISA	In-Space Assembly
ORU	Orbital Replaceable Units
PID	Proportional Integral Derivative
MPC	Model Predictive Control
STOMP	Stochastic Trajectory Optimization for Motion Planning
CHOMP	Covariant Hamiltonian Optimization for Motion Planning
OMPL	Open Motion Planning Library
IK	Inverse Kinematics
KDL	Kinematics and Dynamics Library

1 Introduction

1.1 Background

Large-scale space structures are becoming increasingly important, with projects that were once thought impossible now becoming a reality thanks to advances in space technology. Although the payload capacity of modern launchers has increased, it remains insufficient for deploying large structures in a single launch, thus requiring them to be sent in multiple parts and assembled in orbit. This process often requires the use of robotics, as it is a dangerous and complex task to perform for astronauts in a low-gravity environment. Mobile robots, such as walking manipulators or multi-arm robots, are thus essential for this approach as their flexibility enables a wide range of applications. However, the robot's movement along the structure can cause the platform to undergo rotational deviation due to the conservation of angular momentum in orbit. This can lead to major problems, such as deviation from the intended trajectory or loss of communication with Earth, which necessitate active counterbalance measures that consume valuable fuel.

1.2 Aim

This project aims to implement a path planning algorithm for a multi-arm robotic system that passively minimises the disturbance induced on the structure it is moving on. By optimising the robot's motion, the project seeks to enhance the reliability, safety and efficiency of In-Orbit Assembly operations, reducing the risk of unintended rotational deviations and the associated need for fuel-consuming counterbalance measures.

1.3 Objectives

First, a comprehensive literature review will be performed, to understand the locomotion challenges for In-Orbit Assembly. Algorithms proposed for the motion

planning of multi-arm robots in orbit will also be studied, to evaluate how they address these challenges.

Then, a locomotion scenario will be defined and the best algorithm for it will be implemented on the MARIO platform. Simulation will be performed using ROS2 and Gazebo, to validate the motion planning algorithm. In the future, a real-life demonstration with the MARIO robotic platform at the ASTRA-Lab (Advanced Space Technology for Robotics and Astronautics Laboratory) will be done to demonstrate the effectiveness of the algorithm

1.4 Outline of the thesis

First, Section 2 provides a comprehensive literature review of the topic, covering In-Orbit Assembly projects, the various types of robots employed for such tasks, and the planning algorithms used to compute manipulator trajectories that minimise disturbances.

Section 3 defines the problem and the scope of this work, first describing the problem in the context of space conditions, then detailing the experimental setup that replicates microgravity on Earth, and finally presenting the simulation that models this setup. Additionally, the design of both the structure and the docking interface developed for this experiment is explained in detail.

The Design methodology section outlines the design work carried out in this study, with the primary goal of minimizing disturbances. It first describes MARIO's locomotion to characterize the types of movements the robot must perform. It then presents the rationale for the chosen algorithm, followed by a detailed explanation of the algorithm itself and the custom cost function developed for this work. Finally, it explains how these components integrate into the overall software architecture of the robot.

The Results and Discussion section describes the iterative experiments that were performed and details their findings, supported by quantitative results, graphical

representations and comprehensive explanations. It provides a comprehensive overview of all experiments and their outcomes.

Finally, the Conclusion & Future work section emphasizes the contributions and key findings of this study, demonstrating how they address the objectives established at the beginning of the work. It also outlines potential avenues for future work, including possible enhancements to the proposed methodologies and extensions of the experimental framework.

2 Literature review

2.1 In-Space Servicing, Assembly and Manufacturing

In-Space Servicing, Assembly and Manufacturing [1] (ISAM) is gathering a lot of interest, by allowing flexible and sustainable operation while enabling further exploration and research possibilities.

ISAM encompasses 3 different aspects:

- **Servicing:** consumable refuelling, component repair or de-orbiting
- **Assembly:** assembling large structures in space, as they can't be sent in a single piece [2]
- **Manufacturing:** manufacture elements for use in space (to reduce reliance on Earth supply) or for Earth by taking advantage of micro-gravity to manufacture products that can't be produced there [3] such as bone tissue [4] or exotic optic fibre [5]

This report focuses on In-Space Assembly (ISA) and In-Orbit Assembly of large structures, with existing projects such as building large telescopes from modular mirror tiles [6–9] or Space-based solar power (SBSP) [10,11]. These tasks are often designed to be performed by robotic systems, as they are too complex and dangerous for astronauts to perform them. Additionally, space manipulators are much more efficient than astronauts for this kind of repetitive and structured tasks.

2.2 Space robotic systems for assembly

Three different types of robotic systems are used for ISA tasks (Figure 1, [12]):

- **On-board robots:** attached to a satellite or module to perform tasks in its direct neighbourhood
- **Free-flying robots:** that use propellant to navigate in micro-gravity and perform assembly tasks
- **Persistent assembly platforms:** persistent structure in space with a robotic arm that allows to attach any kind of module to the structure

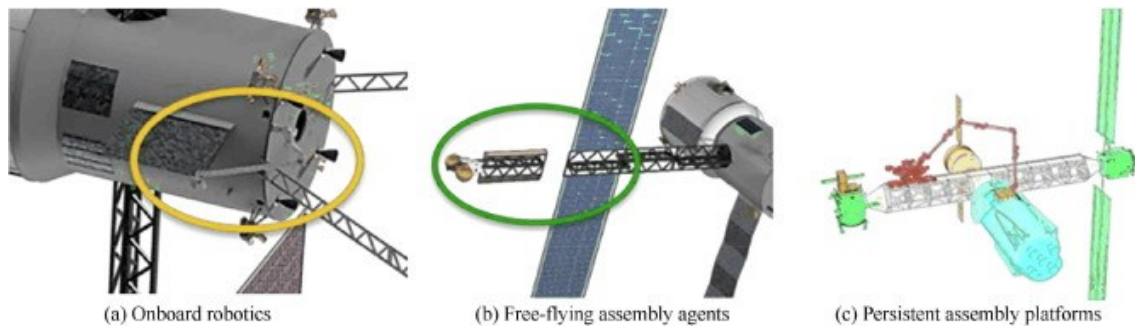


Figure 1: Classes of space robots for assembly [12]

While onboard robots share similarities with terrestrial systems, thus making them more straightforward to implement, their fixed base severely limits their workspace and flexibility. In contrast, free-flying assembly robots offer much greater manoeuvrability and can perform complex tasks thanks to their mobility. However, they present significant control challenges and rely on propellant for movement, which either limits their operational lifespan or introduces the added complexity and cost of in-orbit refuelling.

This project will focus on walking robots, moving on persistent assembly platforms or similar structures. These robots are not fixed to a base but are rather able to crawl on the structure using standard interconnects [13], such as HOTDOCK [14] or SIROM [15], that provide a mechanical docking mechanism as well as power and data. This allows these robots to move to any part of the structure to perform assembly tasks, which makes them very flexible. Additionally, they only require electricity to function, which can be easily supplied by solar panels on the structure, thus making it a reliable and sustainable solution. Often, those robots contain three arms, of which two are used for locomotion, while the third one carries the modular tiles and/or any tool required for assembling them.

Several projects already exist with such robots. First, the Multi-arm Installation Robot for Readyng ORUS and Reflectors (MIRROR), aims to build a space telescope by assembling mirror tiles using walking robot manipulators. Two different robots were developed as part of this project: the Multi-Arm Robot

(Figure 2, [16,17]) as well as the Multi Arm Relocatable Manipulator (Figure 3, [18]).



Figure 2: Multi-Arm Robot (MAR) for on-orbit large telescope [16]

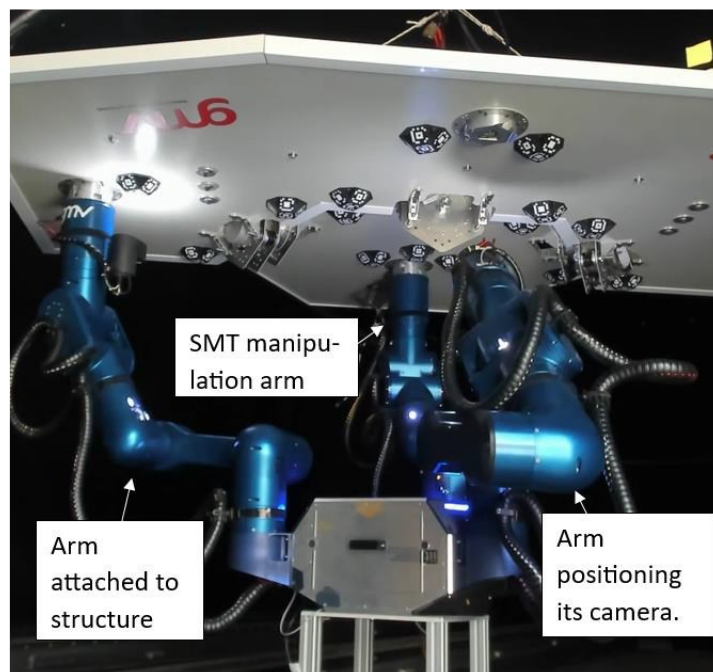


Figure 3: Multi Arm Relocatable Manipulator (MARM) robot performing assembly tasks [18]

Another walking manipulator project for space assembly is the MODular Spacecraft Assembly and Reconfiguration (MOSAR) robot arm [19]. This robotic

arm (Figure 4) is able to perform a wide variety of tasks, such as reconfiguring the spacecraft (Figure 5), performing docking and rendezvous operations.

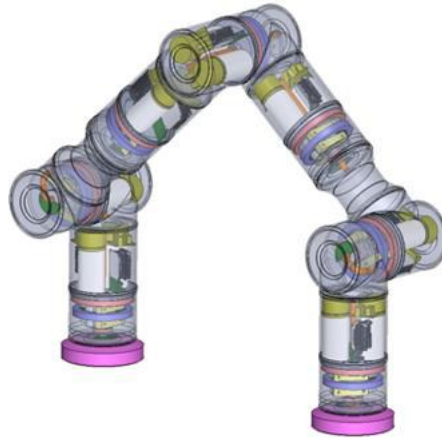


Figure 4: MOSAR Walking Manipulator Concept [19]



Figure 5: MOSAR Demonstrator setup [19]

Cranfield University also developed its own space walking manipulator (Figure 6), the Multi-Arm Robot for In-orbit Operations (MARIO) as part of the ASTRA-Lab [20]. MARIO is mounted on an air bearing platform, allowing it to move freely in the horizontal plane without any friction, to simulate a micro-gravity environment.

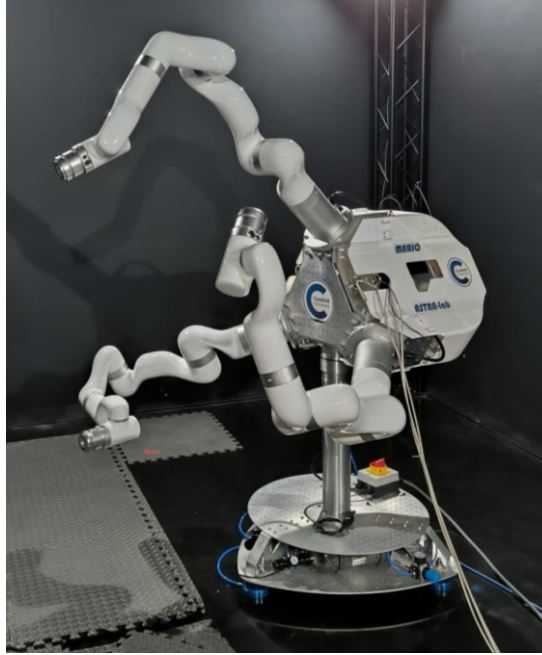


Figure 6: MARIO in the ASTRA-Lab

However, as these robots move on the structure, they can influence the platform's attitude and trajectory because of momentum conservation. This effect is particularly significant when the mass of the structure is comparable to or less than that of the robot, but it remains relevant even for large structures such as the International Space Station (ISS) [21]. Therefore, it is essential to use path planning algorithms that minimise those disturbances.

2.3 Planning architecture

Rodríguez et al. highlight a hybrid approach to path planning for performing assembly tasks with robotic manipulators, composed of 2 layers (Figure 7) [22]. The logic layer is responsible for optimising the overall assembly, by choosing the order of assembly that minimises the total disturbance. The physical layer on the other hand is responsible for optimising the movement in a point-to-point trajectory [22].

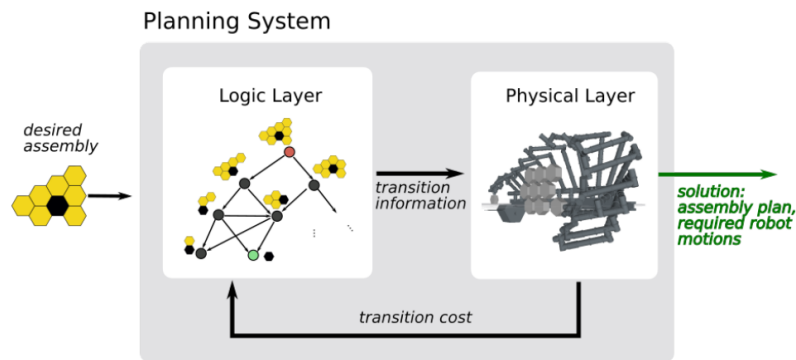


Figure 7: Hybrid planning architecture composed of 2 layers [22]

This hybrid approach is particularly efficient in minimising disturbances, as it allows a reduction of the attitude disturbance of the platform between 50% and 80% while the physical layer alone only achieves up to 45% [22].

However, the rest of this work will be focusing on the physical layer, to remain scenario agnostic, as the logic layer will depend on the type of structure that is assembled. Thus, this project will only try to minimise the force and torques applied to the structure by the moving manipulator in a point-to-point trajectory.

2.4 Reaction minimising algorithms

The literature identifies three complementary layers for controlling a space manipulator to minimize reaction during point-to-point motion:

1. **Real-time control:** The lowest layer, handling immediate adjustments to counteract disturbances as the arm moves.
2. **Path optimisation:** The middle layer, shaping the trajectory in advance to reduce reactions before execution.
3. **Path planning:** The highest layer, determining feasible paths and waypoints that respect kinematic and environmental constraints.

Together, these layers work hierarchically: planning feasible paths, optimising them to minimise disturbances, and actively compensating for residual reactions during execution.

2.4.1 Real-time control

The first approach is to use a **real-time control strategy** on the motor joint torques. The trajectory is performed in joint space, by interpolating between the current and target joint configurations, in a way that minimises the attitude disturbance.

Few examples of algorithms used are Reaction Null Control Strategy (RNCS) and a combination of RNCS and Jacobian Transpose Control (RNS-JCT) [23]. Although these control approaches are computationally inexpensive, the downsides are that they don't consider obstacles that might be in the way, and that some algorithms drastically reduce the number of reachable points.

Another approach is to use Model Predictive Control (MPC) to compute control strategies over a certain time horizon, which minimise a given cost function [21] that would factor in attitude disturbance. However, this approach only allows to minimise the disturbance locally, in the given time horizon, but not of the overall trajectory. Similarly, the Quadratic Programming (QP) formulation is an optimiser which is often used for walking robots and space applications [21].

2.4.2 Path optimisation

The second level is the **optimisation** of a pre-computed trajectory. These trajectories can be either very simple, such as cubic trajectories [22], or consider the obstacles in the way. Those trajectories are then modified, to minimise a given cost function, that in this work corresponds to the attitude disturbance.

An example optimiser used for space applications is the Stochastic Trajectory Optimization for Motion Planning (STOMP) [22,24], which is an improved version of the Covariant Hamiltonian Optimization for Motion Planning (CHOMP) [25]. One limitation of this algorithm is that it cannot guarantee an optimal solution within finite time. Instead, it can only approximate the global optimum with a certain probability.

2.4.3 Path planning

Finally, the last option is **path-planning algorithms**. Those algorithms don't need a pre-computed path, they just need a representation of their surrounding environment and obstacles as well as a cost function for the path to optimise.

The sampling-based planners, such as PRM [26] or RRT [27,28], work by randomly sampling the configuration space and selecting the samples that score the best. These algorithms are very fast, as they don't have to explore the entire configuration space, and guarantee a solution, however this solution is sub-optimal.

Search-based planners on the other hand solve that issue, by providing optimal paths, but at the cost of additional computing requirements. Examples of such planners are RRT* [29] or A* and its modified versions [30].

2.4.4 Others

Additionally, some approaches try to implement artificial intelligence, such as Support Vector Machine (SVM), as part of the process, to learn from the optimisation and provide a more accurate initial guess for the trajectory and thus reduce the overall computation time [31].

Also, regardless of the selected approach, some algorithms are used offline (before the robot starts moving) while some of them are run online (while the robot is moving the path gets recomputed at each time step). For instance, in the previous examples QP optimises the path at each time step [21] while STOMP optimises it offline [22].

The main advantage of running the algorithms online is to enable dynamic planning, for example in a constantly changing environment or with moving obstacles. However, in space we can assume that there is no dynamic obstacle that would need to be avoided by the arm. Indeed, the only moving objects would be other potential robots, whose paths can be planned in advance to avoid

collisions, and space debris, which would be difficult for the robot to avoid due to their high speed and the complexity of detecting them in time.

2.5 Approach trade-off

Although easier to implement, control strategies pose major issues such as a drastically reduced work envelope and the absence of collision avoidance.

Then, while search-based planners ensure optimality, they are often computationally expensive, especially for manipulators with high degrees of freedom. Thus sample-based planners such as RRT and its variations are often preferred. Although these planners allow to find solutions quickly, they are sub-optimal.

On the other hand, optimisers allow to improve an existing solution, to get closer to the optimal one. However, when given a solution far from the optimal one, they might take some time to converge.

Finally, literature also highlights the combination of those approaches as a complete planning pipeline. This allows to achieve the optimal combination of characteristics from the individual approaches. For example, an LQR-RRT* planner can be used to get a first sub-optimal solution, which is then optimised and smoothed to be closer to the optimal one and finally followed using an MPC controller [32].

3 Problem definition

3.1 Problem

The scenario that will be considered in this work is MARIO, a multi-arm robotic system, crawling along a large structure (Figure 8) mounted on a spacecraft. As it moves, the conservation of momentum causes the robot's motion to be transferred to the structure, resulting in attitude disturbances of the entire system.

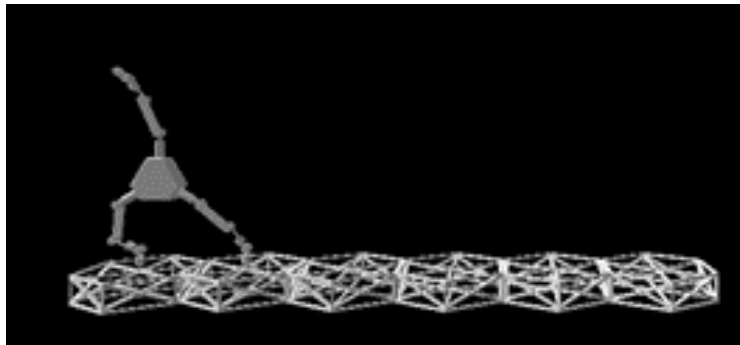


Figure 8: MARIO crawling on a structure [20]

These disturbances must be actively compensated by the spacecraft's attitude control system, using either reaction wheels or propellant-based actuators, in order to maintain the same orbit and communications line of sight with Earth. As propellant is scarce and critical, it's imperative to minimise its use to the strictly necessary manoeuvres. Thus, it is essential to passively reduce the disturbance when planning MARIO's motion.

It will be assumed that the structure will not be externally actuated while the robot is moving, which means that the satellite won't be using propellant or reaction wheels while the robot performs locomotion. Additionally, only two arms will be used for locomotion, while the third arm remains in its stowed position to reduce MARIO's inertia. Finally, MARIO will start already attached to the structure, to focus on locomotion rather than trying to grab the structure while free-floating.

3.2 Transcribing the problem in a lab demonstration

Recreating the conditions found in orbit (free-floating platform and robot in micro-gravity) on Earth is very complex, expensive and resource intensive.

Some solutions such as NASA's Neutral Buoyancy Laboratory (Figure 9) use pressurized suits and balloons in a pool to counteract the Earth's gravitational pull [33,34]. While it allows for a very accurate representation of micro-gravity, it introduces other challenges such as waterproofing all the hardware and requiring trained divers to handle the system.

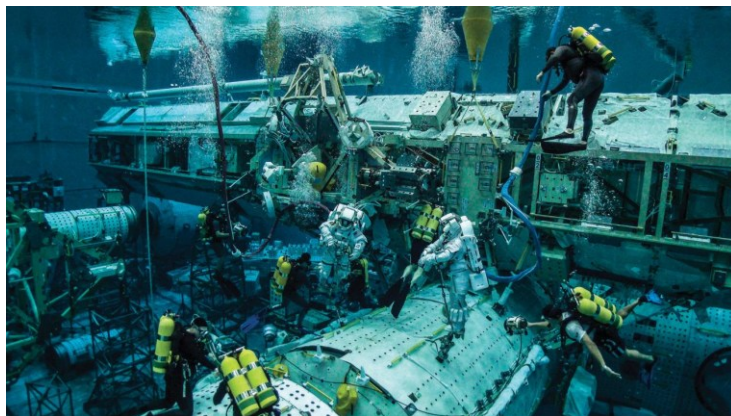


Figure 9: NASA's Neutral Buoyancy Laboratory [35]

Other solutions use cables hanging from the ceiling to carry the robot's weight. This is for example done for the Multi-Arm Robot (MAR) demonstrator (Figure 10). However, this approach reduces the range of motion the robot can achieve, which is an issue when trying to perform locomotion tasks.

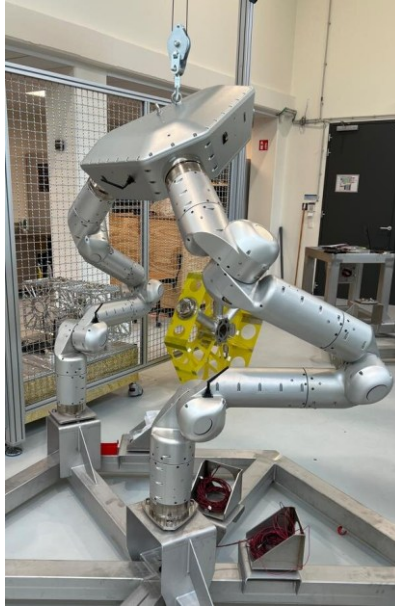


Figure 10: Multi-Arm Robot hanging from a cable [36]

Finally, other solutions use robotic arms to simulate the target's movement based on force and torque sensors [37]. While this solution is really good for testing small subsystems such as a docking mechanism, it becomes unpractical for larger systems. Indeed, this solution would require an arm to carry MARIO and an arm to carry the structure, and as both are pretty heavy, this is not a valid solution. Additionally, this solution also requires actively computing all the necessary reactions to simulate the system.

Our lab demonstration will take place in Cranfield University's ASTRA-Lab. Similarly to ESA's Orbital Robotics Lab, the lab's floor is made to be extremely flat ($\pm 1.5\text{mm}$, Figure 11). By using air bearings, with a constant compressed air supply directed towards the floor, a thin air layer is created, allowing any platform with air bearings to slide on top with almost no friction and thus simulate micro-gravity in 3 dimensions (2 horizontal translations and yaw rotation). Compared to the previous solutions, the micro gravity is simulated passively (no computation required), and it allows a much larger range of movement. A risk assessment for the ASTRA-Lab demonstration can be found in Appendix B.

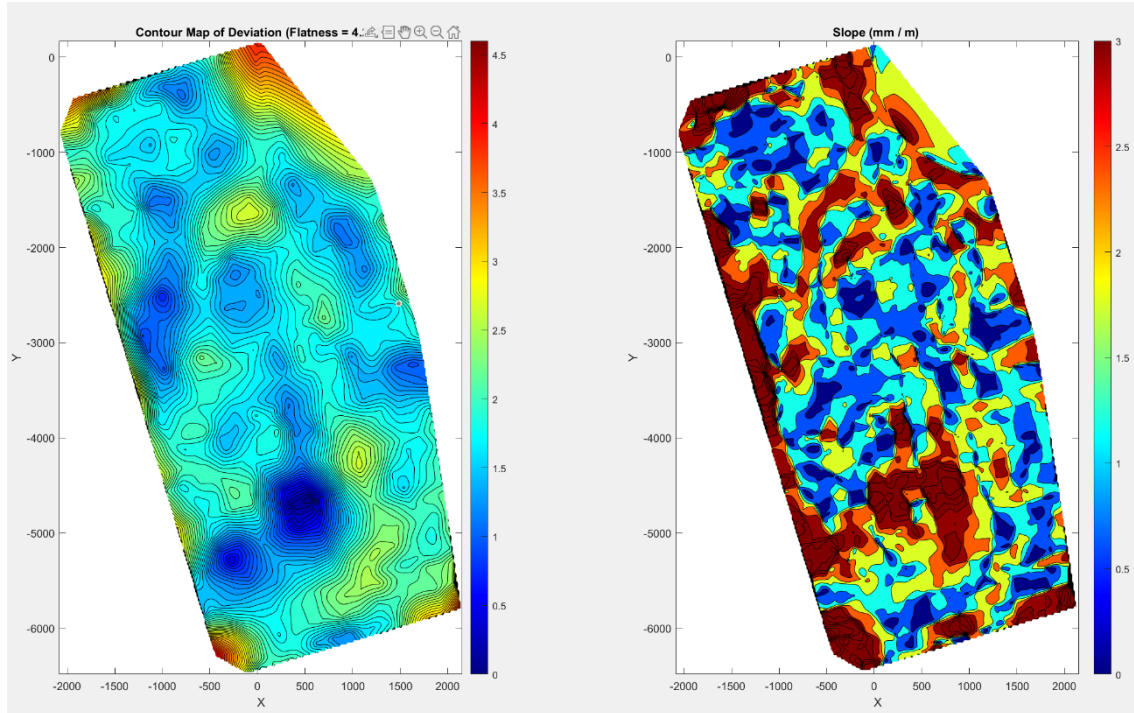


Figure 11: ASTRA-Lab floor slope measurements

By putting MARIO on an air-bearing platform (Figure 12), we can simulate microgravity in the horizontal plane, which is sufficient to highlight and measure any attitude disturbance of the system in the dimensions indicated earlier.

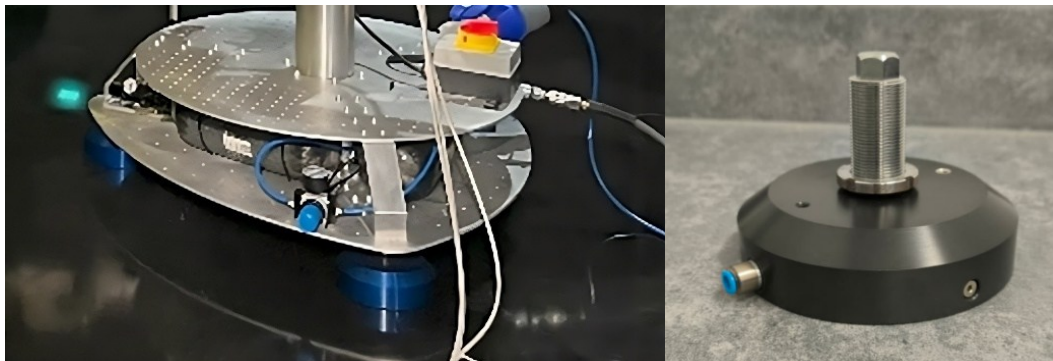


Figure 12: MARIO's air bearing platform

3.3 Lab experimental setup

Since the goal is to demonstrate the robot's ability to crawl on a structure, part of this thesis focused on designing that structure specifically for the robot to

traverse. Multiple docking points will be attached on that structure to allow MARIO to crawl onto it.

3.3.1 Structure design

A modular structure was designed for the experiment. It consists of 3 units made from aluminium profiles, which provide both structural integrity and attach points for the docking points.

Each unit (Figure 13) measures 1.2m (height) $\times$ 1.0m (width) $\times$ 0.8m (depth) allowing to attach the docking points around 60cm above the floor (ideal height for locomotion for MARIO), and later attaching elements at 1.2m to perform assembly tasks with the third arm (top arm).

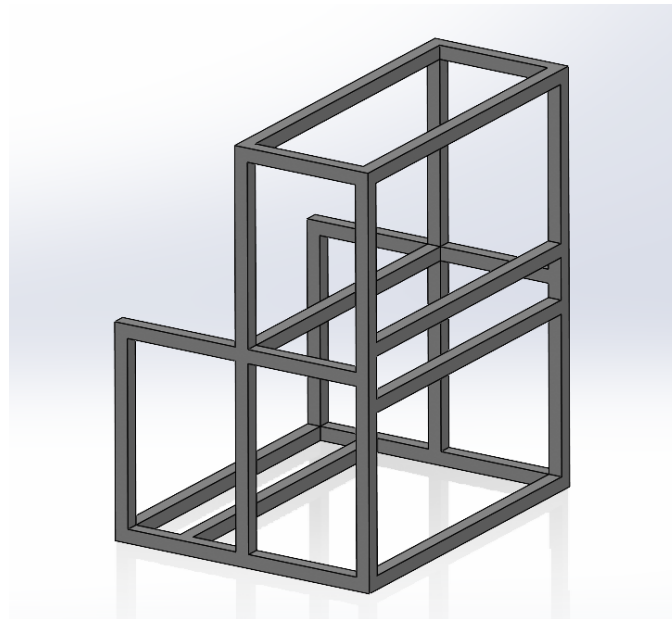


Figure 13: Crawling structure unit

Each unit will have counterweights at the back, to prevent the structure from tipping when MARIO exerts a normal force on it. Calculations (Appendix A) show that at least 12kg are needed to prevent the structure from tipping when the robot exerts a force of 25kg (maximum that the tool's electromagnet can hold).

These units will be able to attach to each other, either as a straight line (3 units in a row, Figure 14), or as a 90° angle (L shape with 2 units on one side and one

on the other, Figure 15). This modularity will allow the structure to be used for demonstrating both linear and rotation movements, as well as any future experiments in the lab (with the possibility to attach elements on top of the structure for the third arm to perform operations on). Indeed, those aluminium profiles can be moved to different heights or reconfigured in different shapes.

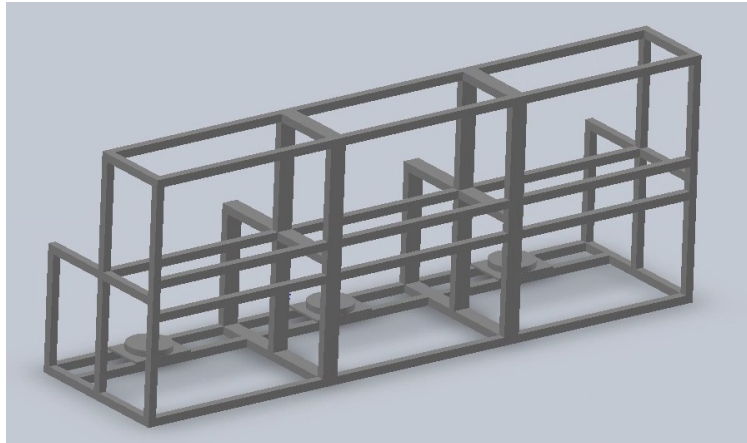


Figure 14: 3 Units in a row

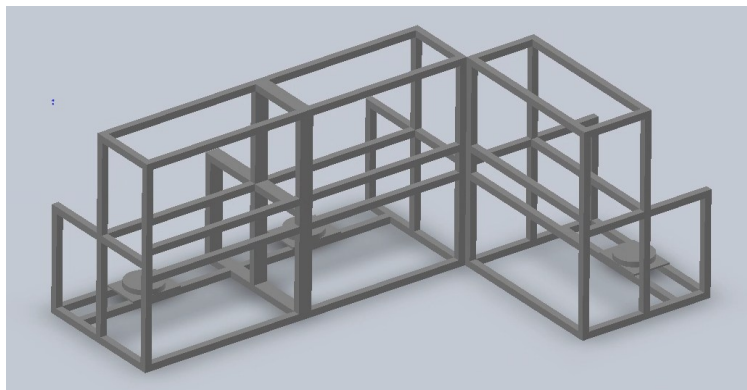


Figure 15: 3 Units in a L shape

3.3.2 Docking interface

The docking interface (Figure 16) is split into 3 main parts:

- **The tool changer:** attached to the robot arm, it allows to easily change the tool mounted on the manipulator.
- **The docking tool:** attached as the end-effector of the locomotion manipulators, it provides mechanical docking.

- **The docking point:** attached to the structure, for the docking tool to attach to

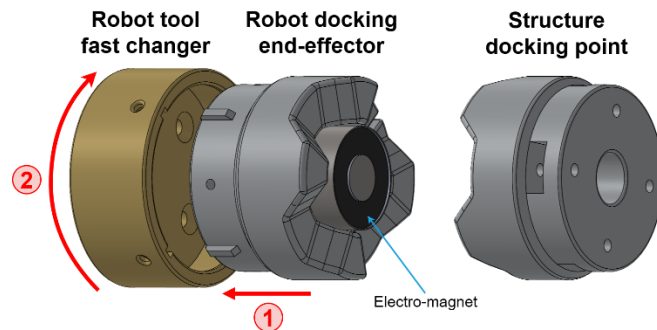


Figure 16: Docking interface

This interface allows mechanical docking in 2 ways:

- The electromagnet provides a strong bonding between the docking tool and the docking point which contains magnet counterparts. This bonding can withstand forces up to 25kg (as per the chosen magnet's specifications).
- The crenelated mechanical design allows for smoothly guiding the end-effector into the docking point, as well as a strong resistance to lateral forces that can be exerted when crawling on the structure.

The tool changer is fastened to the end of the robotic arm (Figure 17), while the docking tool is secured through a sliding-and-rotation motion, followed by tightening lateral screws (Figure 16).

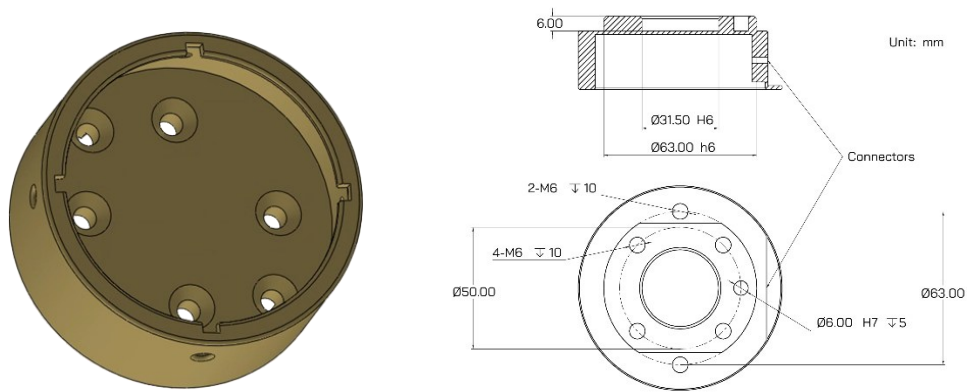


Figure 17: Tool changer (left) and UFactory 850 arm attach points (right)

Once securely attached to the arm, the tool's magnet is wired into the arm's Input/Output system using the wiring diagram in Figure 18. When the tool's Digital Output 0 (Pin 7) is turned on, the control circuit is closed, thus triggering the relay and turning the electro-magnet on. A protective diode is used as the UFactory documentation suggests [38]. This split circuit is used to have a minimum current draw from Pin 7 in the control circuit, as the max current allowed is 5mA [38]. This wiring allows the electromagnet to be controlled directly from the arm's controller, without requiring an external device such as an Arduino or a Raspberry PI.

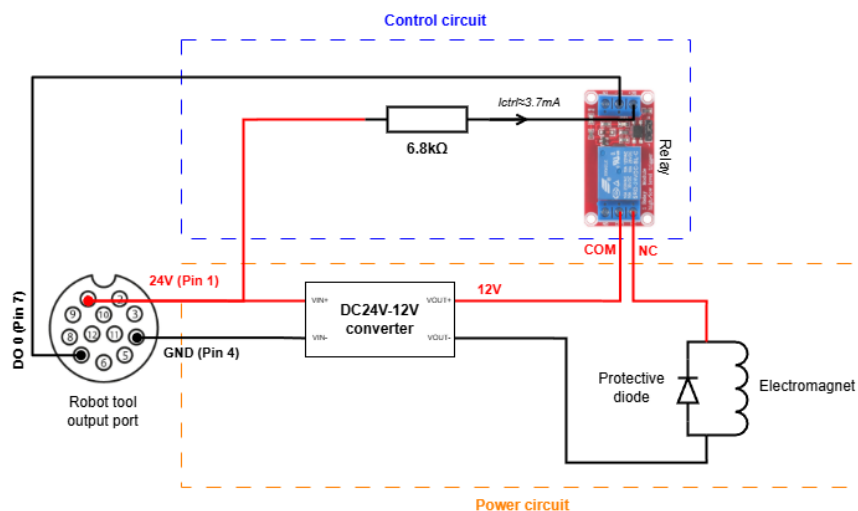


Figure 18: Docking tool wiring diagram

3.3.3 Final setup

The crawling structures were assembled from black aluminium profiles, and gym counterweights are placed at the back (Figure 19).



Figure 19: Built structure

The docking interface was 3D printed with PLA and a 70% infill (for structural integrity during MARIO's movements) and assembled (Figure 20).

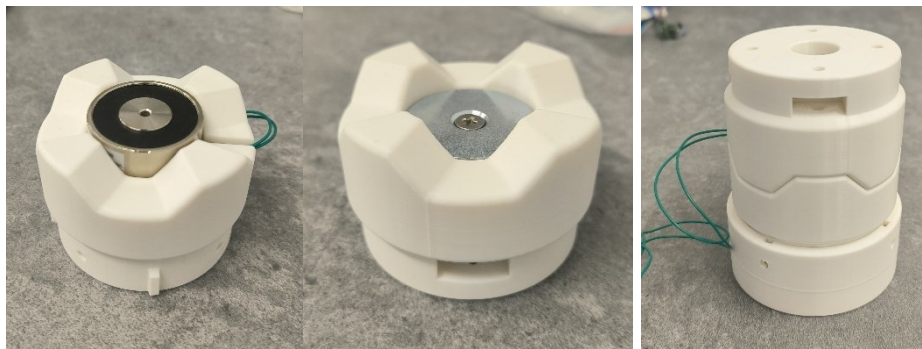


Figure 20: Docking interface (docking tool, docking point, full assembly)

Two possible experiments can be done with those elements:

- MARIO crawling on the static structure, and measuring disturbance through force and torque sensors on the robotic end-effector
- Putting the structure on air bearings, and monitoring the structure's position and attitude through the lab's motion capture system

3.4 Simulation setup

A Gazebo simulation (Figure 21) was built to rapidly iterate, test and validate the results, before doing any experiment on the real robot in the ASTRA-Lab.

To replicate MARIO's air bearing system, both the structure and MARIO are modelled as having almost no friction with the floor ($\mu_1 = \mu_2 = 0.001$) which allows them to slide on the floor.

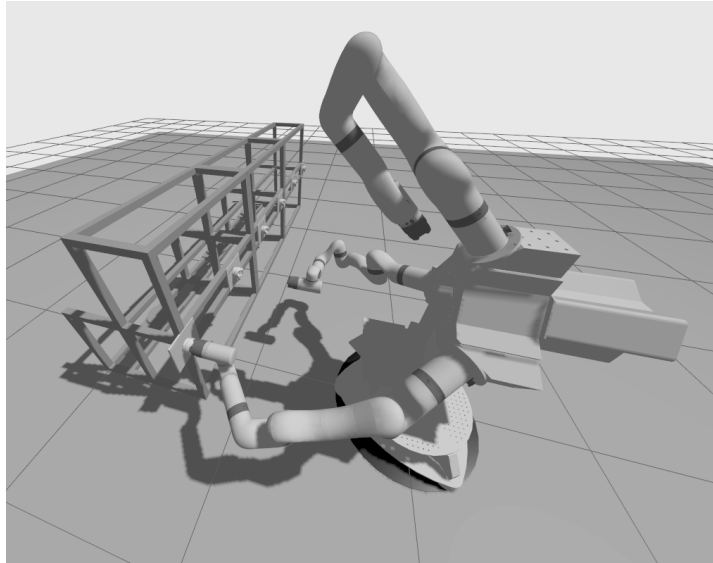


Figure 21: Gazebo simulation

The docking and undocking were simulated using the DetachableJoint Gazebo plugin, which allows to attach an element to the end of a robot arm. In our case, a docking point can be attached and detached from the docking tool, allowing the structure to move along with the arm (Figure 22).

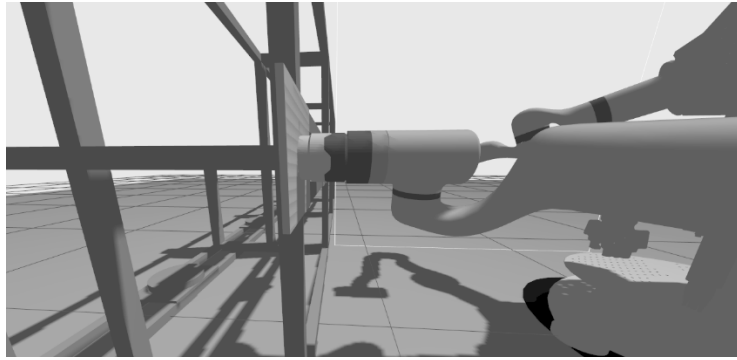


Figure 22: Docking in simulation

4 Design methodology

4.1 Crawling motion

MARIO will be performing an inchworm style locomotion which mimics how an inchworm moves. With a dual arm robotic system, this corresponds to:

1. Anchoring one arm: The arm is fixed in place, using its docking end-effector
2. Extending the other arm: The other arm moves forward, reaching for the next point
3. Anchoring the free arm: The newly advanced arm attaches to the structure
4. Pulling the rest forward: The previously docked arm releases and is pulled toward the other one

Figure 23 illustrates this theoretical movement cycle for MARIO, attaching and detaching from docking points on a structure.

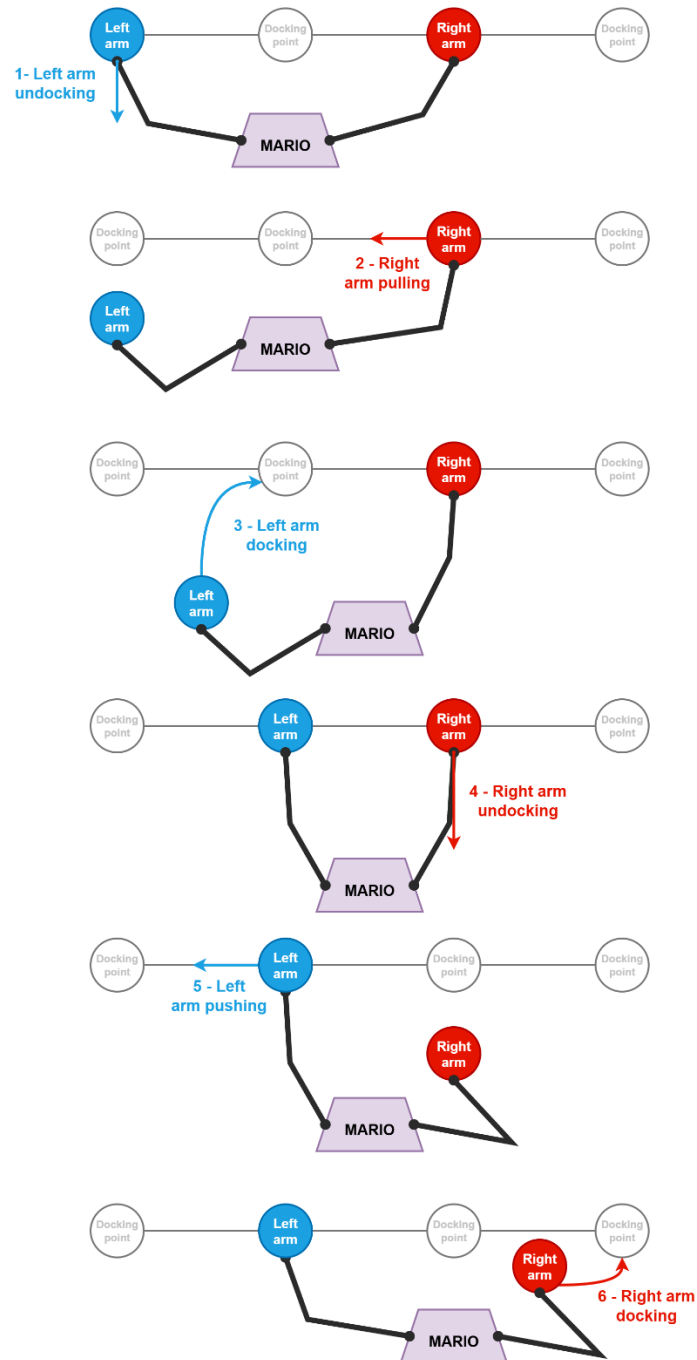


Figure 23: MARIO's crawling cycle

All the positions (docking points and intermediate locomotion positions) are defined relative to MARIO's base frame. That way, if MARIO is properly docked, the crawling cycle can be run starting from any right step in the cycle, provided that the distance between docking points is known.



Figure 24: MARIO crawling on the structure

Figure 24 shows MARIO performing the crawling motion in simulation. Unlike Figure 23, where both locomotion arms begin docked, here only one arm is docked. This setup replicates the planned real-life demonstration in the ASTRA Lab:

1. MARIO will first be setup far from the structure and extend both arms to their desired starting position
2. Once arms are opened, someone will manually position MARIO close to the structure and dock one of its arms into a docking point. Docking both

arms initially could introduce alignment errors, that's why only one arm is docked at the beginning

4.2 Path planning implementation

The literature review highlighted 3 layers to path planning:

- Path planning algorithms
- Optimiser algorithms
- Control layer

As the UFactory 850 arms used by MARIO currently only provide PID control, we cannot minimise disturbance by changing the control layer. Thus, disturbance reduction must happen at the above layers.

Since STOMP has been proved to reduce disturbance of a manipulator on a satellite [22], that's what we will try to implement on MARIO. This will enable to initially validate the algorithm, while still allowing to later introduce a planning algorithm to generate a better initial trajectory for STOMP to optimise.

4.2.1 The STOMP algorithm

The STOMP algorithm is an iterative algorithm that optimises a given trajectory. It is split into 4 steps (Figure 25):

1. First, an initial trajectory (red) is received or generated by the algorithm
2. Neighbouring trajectories (black) are generated by adding noise in joint-space to that trajectory
3. All those trajectories are scored using a cost function
4. The trajectory that scored the lowest is selected, and the algorithm iterates by going back to step 2 until the maximum number of iterations is reached

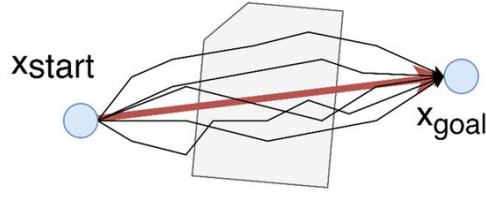


Figure 25: The STOMP algorithm process [39]

4.2.2 Custom cost function

Usually, the default cost function for STOMP only factors collisions with the environment, as well as the robot constraints (joint limits, position and orientation constraints):

$$c_{tot}(q) = c_{collision}(q) + c_{constraints}(q) \quad (4-1)$$

with

$$c_{collision}(q) = \begin{cases} collision_penalty & \text{if collision} \\ 0.0 & \text{if no collision} \end{cases} \quad (4-2)$$

$$c_{constraints}(q) = \begin{cases} distance\ to\ constraints & \text{if violated} \\ 0.0 & \text{if validated constraints} \end{cases} \quad (4-3)$$

That cost function is then summed for all the configurations in a given trajectory to obtain the trajectory's cost. That cost is the one used to select the best trajectory before iterating on the next ones (step 4 of the algorithm, explained above).

$$c_{traj} = \sum_{i=1}^N c_{tot}(q_i) \quad (4-4)$$

To try to minimise the disturbance induced by the manipulator movement, an additional term was introduced to the cost function:

$$c_{tot}(q) = c_{collision}(q) + c_{constraints}(q) + \textcolor{red}{c_{disturbance}(q)} \quad (4-5)$$

This additional cost factors in the sum of joint torques as an indicator of disturbance that we want to minimise, as suggested by Rodríguez et al. [40]:

$$c_{disturbance} = \sum_{j=1}^6 |\tau_j| \quad (4-6)$$

4.2.3 Implementation as a MoveIt2 planner

In the software, the MoveIt2 library is used for planning and control of the arms. MoveIt2 provides an abstraction layer for planning, where all planners inherit from the same base class, allowing them to be seamlessly interchanged. Leveraging this framework, the STOMP algorithm is implemented as a MoveIt planner plugin, enabling its use without modifying higher-level software while providing robust trajectory optimization. By default, STOMP defines the initial trajectory as the linear trajectory in joint space, which serves as the baseline for further optimization.

Since STOMP was introduced only in later versions of MoveIt, the planner plugin was backported from ROS 2 Iron to ROS 2 Humble as part of the work carried out in this thesis. The cost functions were then modified to use the one described in the above section, with an additional term corresponding to joint torques (4-5).

Specifically, that cost function term was modelled using the Kinematics and Dynamics Library (KDL). The URDF of the robot is loaded by the library to create the kinematics chain. Then, by giving the kinematics solver the positions, velocities and accelerations applied on the joints, we can obtain the torques for each of them.

The STOMP planner plugin was then modified to treat the disturbance cost as an optimization criterion rather than as a factor rendering a trajectory invalid. Unlike other costs (collision and constraints), which when non-zero were interpreted as making a joint configuration infeasible, the disturbance cost was instead

incorporated as a value to be minimized, reflecting its role in trajectory quality rather than feasibility.

4.3 Software architecture

The whole architecture relies on ROS2 Humble for communicating between the different software modules, and the MoveIt2 library for planning and control of the 3 arms. All the code was written in C++, because of the limitations of the MoveIt Python library.

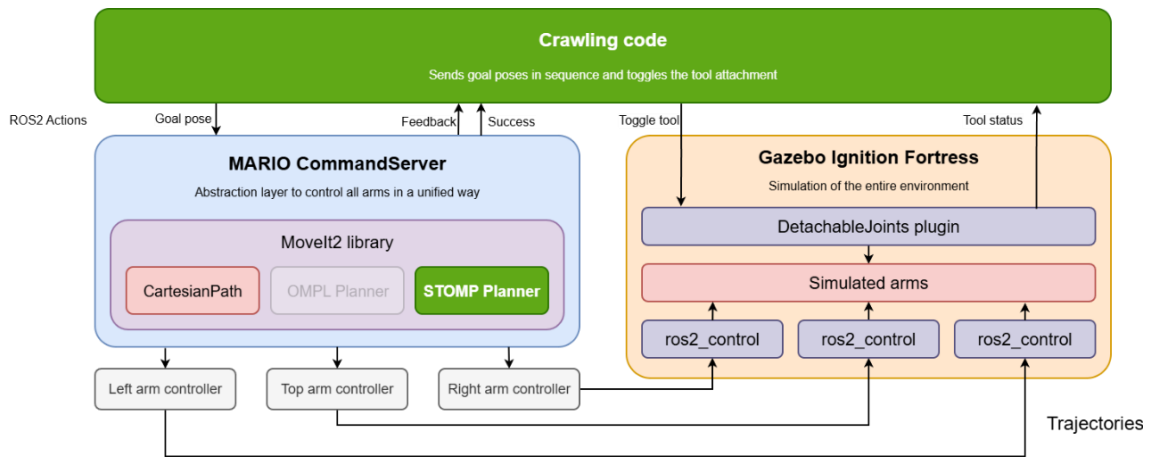


Figure 26: Complete software architecture in simulation

Figure 26 describes the whole planning architecture, which is organised into three main components: MARIO CommandServer, Gazebo and the crawling code. Although this work primarily focuses on the Crawling code, the CommandServer and the STOMP planner, minor modifications were made to all other components of the architecture.

4.3.1 MARIO CommandServer

The MARIO CommandServer is an abstraction layer that allows to control all the arms in a unified way. It provides three main entry points, defined as ROS2 action servers:

- **PoseCommand**

- **JointCommand**
- **CartCommand**

This work only makes use of the PoseCommand and JointCommand, thus the CartCommand will not be explained here.

The PoseCommand enables the specification of a target pose (position and orientation) in Cartesian space relative to a defined reference frame. The request also includes motion constraints and the selected motion type. If there are constraints, they are applied using MoveIt's "setPathConstraints" method. Then, based on this motion type, the server either generates a path using the "computeCartesianPath" method, which produces a linear trajectory **in Cartesian space**, or the "plan" method, which computes a trajectory through the selected planner plugin (STOMP in this work). Figure 27 shows the flow diagram of the process.

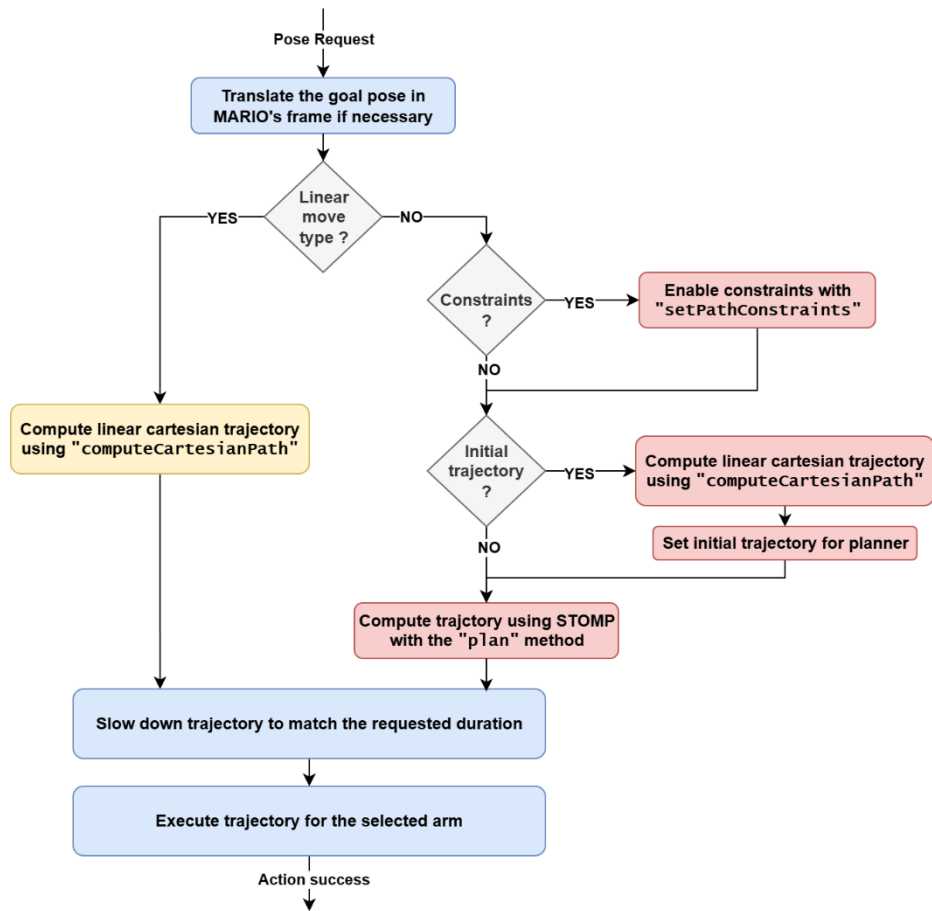


Figure 27: PoseCommand flow diagram

The JointCommand on the other hand plans a trajectory **in joint space** between the current position and the request's target joint configuration using the selected planner plugin (STOMP) through the "plan" method.

Finally, both commands receive the selected arm's id as a parameter. Thus, once a trajectory is planned, it is sent to the corresponding arm controller to be executed. When the trajectory completes, the ROS2 action reports success, notifying the requester that the task has been accomplished.

4.3.2 Gazebo simulation

We use Gazebo for simulating the whole system: MARIO on its air bearings, the structure, and the plane on which they are gliding. Two major Gazebo plugins are used in this work: **DetachableJoint** and **IgnitionRos2Control**.

The **DetachableJoint** plugin is responsible for the docking and undocking of the robot on the docking point. This plugin allows to dynamically add and remove a link from the robot's kinematic chain. In our experiment, the last link of an arm can be connected and disconnected to any of the structure's docking point (Figure 28).

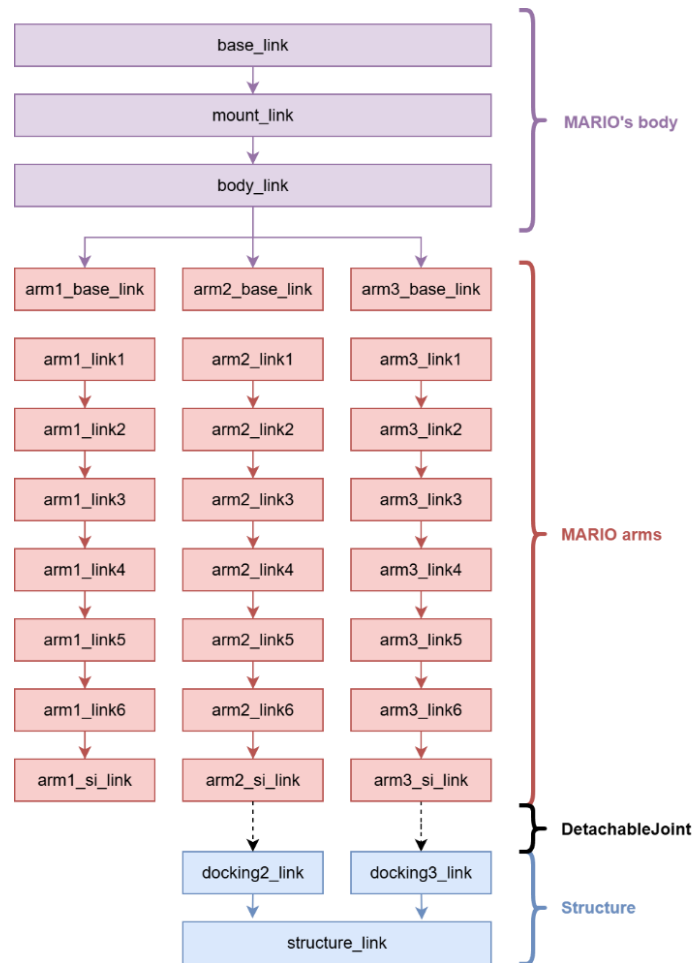


Figure 28: Robot links and DetachableJoint

This approach allows to have a fixed connection, ensuring that the structure follows the exact same trajectory as the end-effector, as though it were an integral part of it. However, since this connection is fixed, the interaction forces and torques between the end-effector and the docking point are not modelled and thus cannot be studied.

The **IgnitionRos2Control** plugin on the other hand is responsible for simulating the hardware control layer. It can receive a trajectory, and send the right joint commands to follow it, ensuring the actual joint positions match the desired positions using a PID controller, and ensuring the motor constraints (maximum angle, velocity and acceleration) are respected.

4.3.3 Crawling code

The “Crawling code” section corresponds to the custom-built ROS2 node that defines the crawling motion sequence, by relying on the two other blocks for execution.

A TaskScheduler class was built to handle the execution of these tasks in a sequenced way. When started, the scheduler runs the first task in the queue, by running the task’s “execute” method. Then, when a task is done, it can call the “nextTask” method of the scheduler, to start the execution of the next task. This is repeated until the queue is empty. Figure 29 illustrates the sequence of tasks handled by the TaskScheduler.

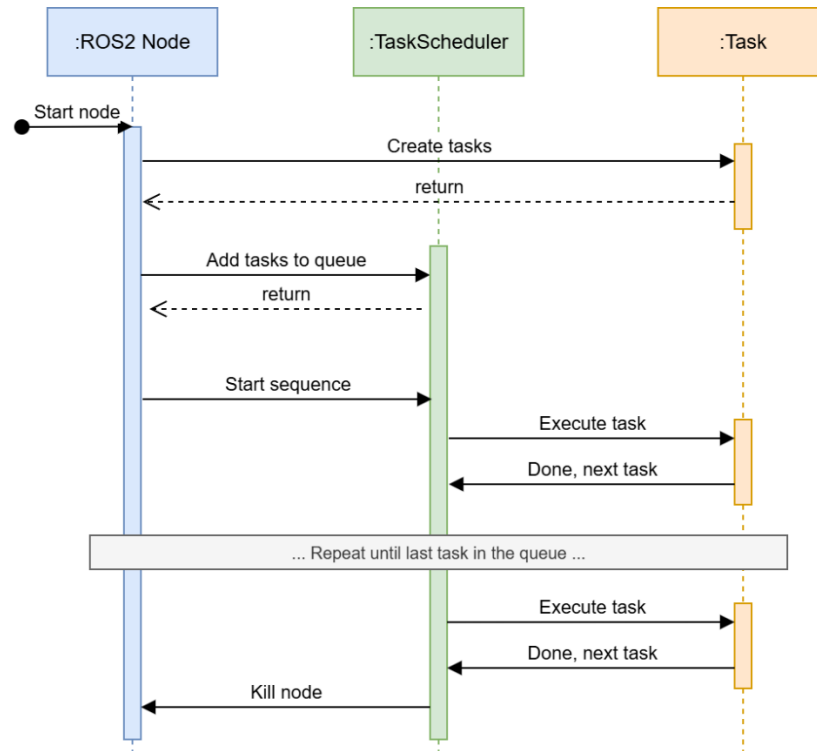


Figure 29: TaskScheduler sequence diagram

Two types of tasks are defined for the crawling motion, that both inherit from the same base class:

- **MoveTask**
- **ToolTask**

The **MoveTask** is responsible for sending goals to individual arms. It sends a ROS2 action request to the CommandServer, either through the PoseCommand or the JointCommand interfaces (depending on its attributes). Once the arm reaches its target position, a success status is received, which triggers the next task in the queue.

The **ToolTask** on the other hand is responsible for toggling the docking end-effector ON (attaching to the docking point) or OFF (detaching), by sending the right commands to the Gazebo DetachableJoint plugin. Once the plugin responds by sending the new tool state, the next task in the queue is triggered.

5 Results and Discussion

The testing process was iterative, involving multiple experiments to achieve the optimal results. Each of the following sections highlights an individual experiment, its outcomes, and a discussion of the findings.

All the following experiments were done in the Gazebo simulation, as time didn't allow for a real-life demonstration in the ASTRA-Lab. During the experiments, the positions, velocities and accelerations of the structure and robot are monitored, as well as the joint values, velocities and accelerations. From this data, 5 main criteria are computed:

- Yaw deviation: difference between initial and final yaw
- Position deviation: difference between the initial and final position
- Maximum linear acceleration throughout the whole trajectory
- Maximum angular acceleration throughout the whole trajectory
- Execution time: including docked movements, planning time, and free arm movements

The aim is to minimise the absolute value of all those measurements (excluding time, as it is not critical in space), with a particular emphasis on the attitude (yaw) of the structure and angular acceleration. Indeed, the spacecraft's attitude is critical in space, as even small deviations can lead to a loss of communication with Earth and reduced power generation due to misalignment of the solar panels with the Sun.

In all experiments, three crawling cycles are performed, following a standard control-test experimental strategy by comparing the measurements between two trajectories:

- **Linear trajectory (control):** Generated using linear cartesian paths for all motions, serving as the reference trajectory
- **Optimized trajectory (test):** Employing the STOMP algorithm, tailored for each experiment

In the next sections, the numerical values will appear in green if they have improved compared to the linear reference, in black if they are similar and in red if they have degraded.

5.1 STOMP constrained planning

5.1.1 Experimental setup

The goal in this experiment was to use STOMP on both locomotion arms, for all motions. However, to do so, it is necessary to constrain both position and orientation of the docked end-effector. The position constraint ensures the end-effector remains in the horizontal plane during the whole trajectory, to prevent it from trying to lift the structure (increase in z) or even drive it into the floor (decrease in z). Similarly, the orientation constraint guarantees that the end-effector only rotates in the yaw axis (relative to MARIO's base frame), ensuring the structure doesn't collide with the floor.

Specifically, the position constraint is defined as a plane that contains both start and end points, with dimensions $5\text{m} * 5\text{m}$ and a 5mm normal tolerance. The orientation constraint is defined as allowing free rotation on the x axis, but almost no rotation (tolerance of 0.05 radians) on the other axes.

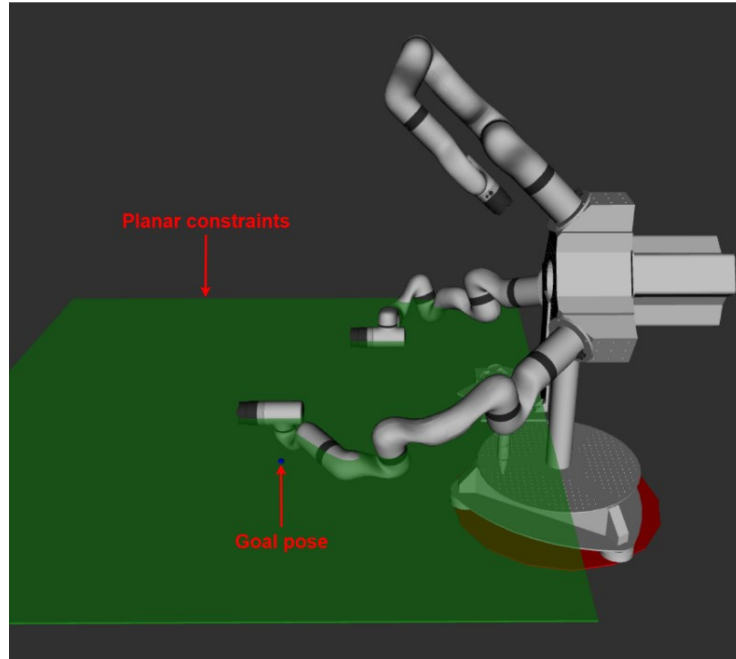


Figure 30: Constrained planning for the left arm when docked

The planar position constraints are illustrated in Figure 30, as the green plane in which the end-effector must stay during the trajectory. The blue square corresponds to the target pose (position and orientation) that the end-effector must reach.

5.1.2 Findings

Two issues were encountered while doing this experiment. First, MoveIt2 for ROS2 Humble doesn't support multiple constraints yet, as it was only implemented in later versions. Thus, the position and orientation constraints could not be enabled at the same time, as it was causing MoveIt to crash.

Additionally, even when using only position constraints, neither OMPL (the default planning library) nor STOMP were able to find a valid trajectory that satisfied all constraints. Increasing the plane's z-axis tolerance from 5mm to 2 cm allowed the planners to find solutions, but only inconsistently—and 2 cm is far beyond the desired tolerance (which would be below 1cm).

One potential approach to overcome this limitation could be the design of a compliant docking point, which relies on elastic deformation to passively absorb variations along the z-axis. However, such a solution may introduce additional challenges, including vibrations caused by the docking point's flexibility, increased mechanical complexity, and reduced docking precision, which makes it impractical.

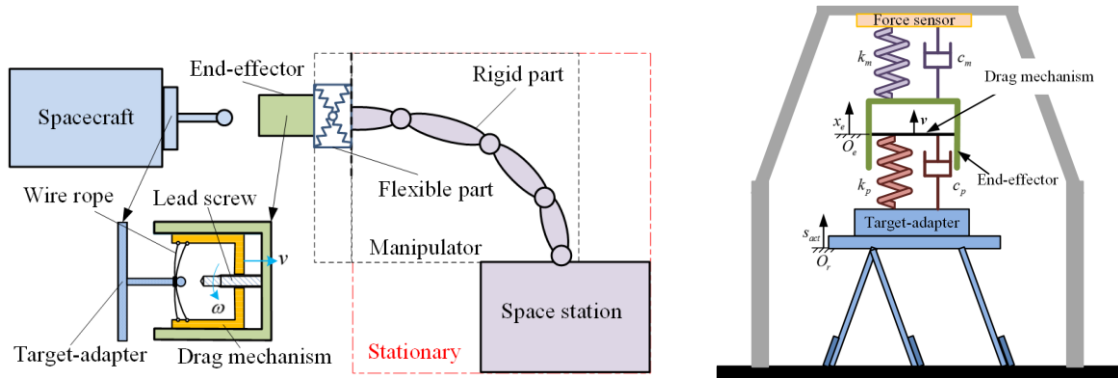


Figure 31: Compliant docking system [41]

5.2 STOMP on the free arm

5.2.1 Experimental setup

As STOMP could not find a path when using planar constraints, it cannot be applied to the docked arm for the moment. Thus, in the rest of the experiments STOMP will be applied only on the free arm (Figure 32), while keeping the linear cartesian path for the docked arm. Although STOMP is used only on the free arm, the experiment evaluates the entire trajectory, capturing the combined effect of the free arm's STOMP-optimized motion and the docked arm's linear motion. With this experiment, the aim was to demonstrate the effectiveness of the STOMP algorithm, while not having to consider end-effector constraints.

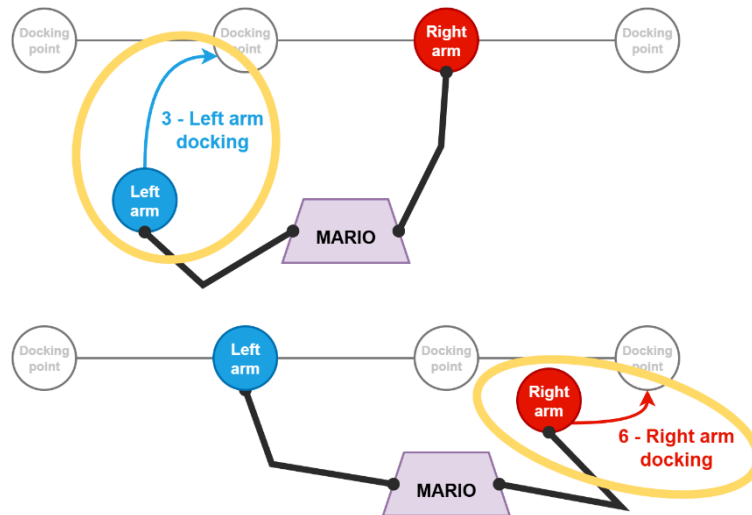


Figure 32: Free arm motion (yellow circles)

5.2.2 Findings

Table 1 shows the results obtained in this experiment. The results indicate a slight increase in yaw deviation and a small decrease in position deviation compared to the linear reference trajectory. Although the maximum linear acceleration is significantly reduced, this reduction does not translate to a decrease in deviation.

Table 1: STOMP Free arm - Structure results

	Linear cartesian space trajectory	Optimised trajectory
Yaw deviation	-31.94°	-35.42°
Position deviation	0.70m	0.67m
Max linear acceleration	143.56m/s²	44.80m/s²
Max angular acceleration	7.48rad/s²	11.16rad/s²
Execution time	375.61s	1306.02s

In this experiment, the free moving arm weighs 15kg and the rest (MARIO's body, static arms and structure) weighs 230kg. As the arm's weight is negligible compared to the rest of the system, its movement almost doesn't impact the trajectory, which was predictable. Indeed, most of the disturbance is induced by the docked arm, while the movement of the free arm has absolutely no impact on the structure's yaw. This is illustrated in Figure 33, with the long plateaus (white

intervals) that correspond to the planning and execution of the free arm trajectory. This minimal impact of the free arm explains why the disturbance results in Table 1 are similar between linear and optimised trajectories.

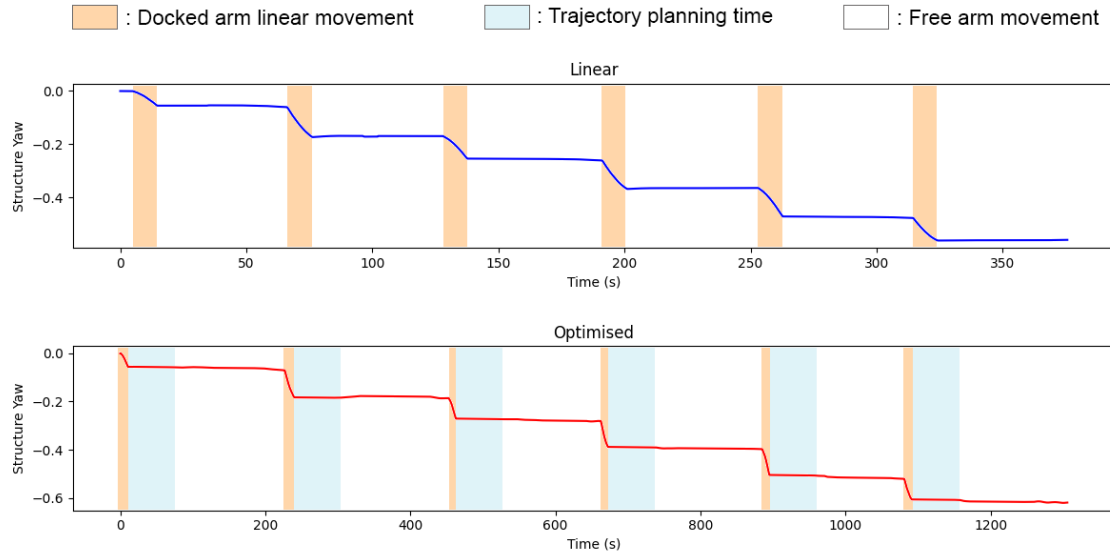


Figure 33: STOMP on free arm - Structure yaw over time

Additionally, with that weight distribution, part of the disturbance was absorbed by the robot itself, causing it to deviate from its initial position (Figure 34). Thus, a new experiment with all disturbance being applied on the structure could be interesting.

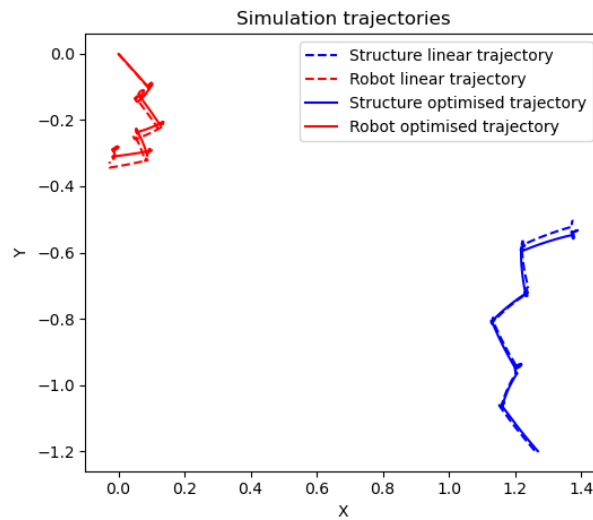


Figure 34: Robot and structure position trajectories in space

5.3 STOMP on the free arm with modified weights

5.3.1 Experimental setup

In the previous experiment, the weight distribution prevented the observation of the free arm's impact on the structure. To address this, the component weights were modified in the simulation for this experiment. Specifically, the end-effector weight was increased from 300g to 20kg, while the structure weight was reduced from 100kg to 1kg.

This adjustment results in a less imbalanced system, with the free moving arm weighing 35 kg and the rest (MARIO's body, static arms and structure) weighing 150 kg, compared to the previous configuration of 15 kg and 230 kg respectively (Figure 35). Although this setup does not reflect real-world conditions, it is primarily designed to highlight the effect of the STOMP algorithm.

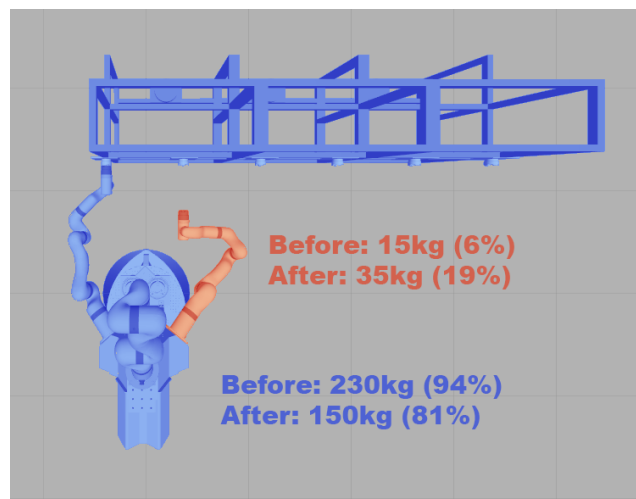


Figure 35: Weight distribution (Red: free arm moving, Blue: static elements)

5.3.2 Findings

With this new weight distribution, most of the disturbance is transferred to the structure, while the robot itself exhibits minimal displacement (Figure 36), which shows that weight distribution is a major disturbance factor.

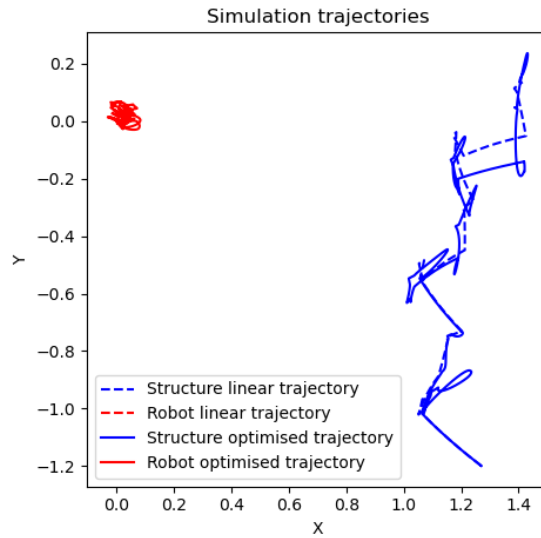


Figure 36: Robot and structure position trajectories in space

Table 3 shows the results obtained from this experiment. While both maximum linear and angular accelerations are reduced, this almost doesn't show in the position and yaw deviations. Thus, even though the results are better than with the default weight distribution, it is still not enough.

Table 2: STOMP on the heavy free arm - Structure results

	Linear cartesian space trajectory	Optimised trajectory
Yaw deviation	-20.84°	-20.52°
Position deviation	1.34m	1.34m
Max linear acceleration	209.06m/s²	88.86m/s²
Max angular acceleration	28.60rad/s²	14.24rad/s²
Execution time	395.75s	1310.27s

A similar experiment was performed, but this time using a linear trajectory in joint space (the initial trajectory that STOMP tries to optimise). This experiment allowed to decrease yaw deviation, as well as the maximum linear and angular accelerations, with a slight increase in position deviation (Table 3). As the

spacecraft's attitude is more important than its position, those results are promising.

Table 3: Linear trajectory in cartesian space VS joint space – Structure results

	Linear cartesian space trajectory	Linear joint-space trajectory
Yaw deviation	-20.84°	-10.78°
Position deviation	1.34m	1.52m
Max linear acceleration	209.06m/s²	118.72m/s²
Max angular acceleration	28.60rad/s²	17.75rad/s²
Execution time	395.75s	614.31s

While this approach resulted in the best disturbance reduction results, it is only a positive side-effect. Indeed, those results were obtained without using STOMP and its custom cost function that factors in joint torques. Thus, those results are not guaranteed in any other scenario. However, this still allowed to demonstrate the potential reduction that can be achieved. Because of these encouraging results, this weight distribution is kept in the next experiments.

Finally, this approach also has other limitations. First, because of the high range of the motors (2 full rotations for the shoulder, elbow and wrist joints), multiple Inverse Kinematics (IK) solutions can be found for the same target pose. However, the IK solver doesn't always return the same joint configuration. Thus, by running the same program twice, we can obtain very different trajectories and results, which is something we want to avoid. These solved configurations are also often far from the current configuration, requiring the arm to make extreme rotations (Figure 37 and 38) compared to the linear trajectory. This needs to be avoided, both to reduce the strain on the motors and to reduce collision risks.

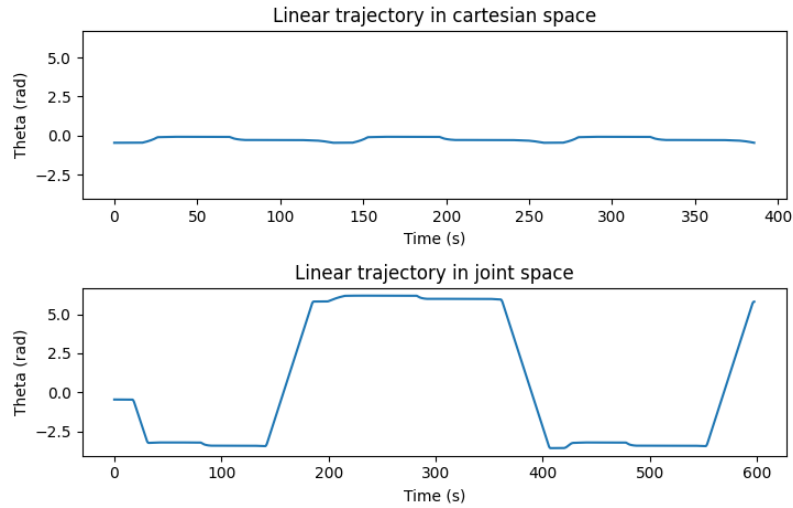


Figure 37: Right arm shoulder joint values over time

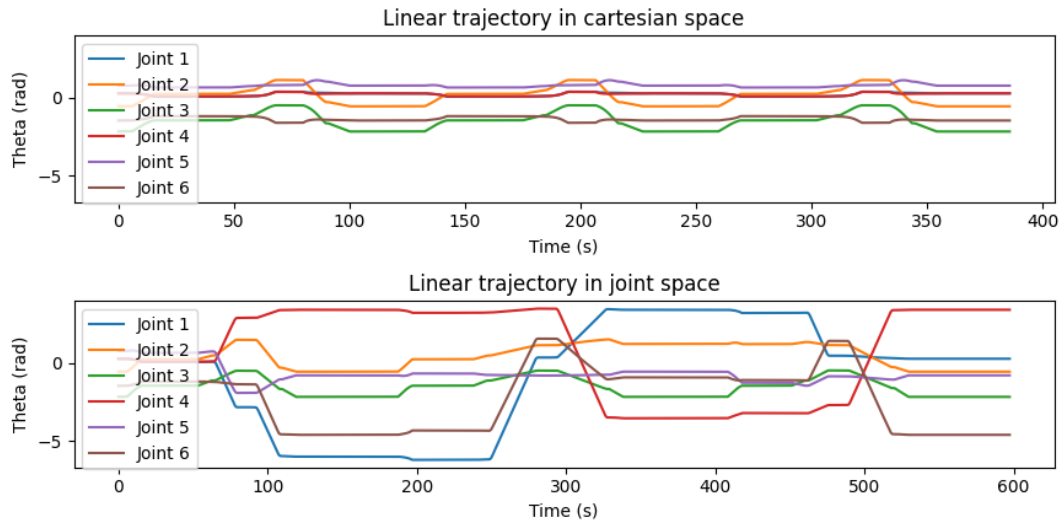


Figure 38: Left arm joint values over time

Additionally, collisions with the floor and the structure were not considered in the trajectory planning, thus sometimes the arm collides with the floor. Similarly, this is not acceptable for a real-life demonstration.

5.4 STOMP on the free arm with joint targets

5.4.1 Experimental setup

Since the IK Solver doesn't always return the same solution, instead of sending target poses, this experiment consists in sending joint targets. These joint targets

are extracted from the linear cartesian trajectory: each pose of the crawling cycle is sent, a cartesian path is computed, and the final joint configuration for that pose is saved in a file.

Thus, this experiment aims to only highlight the effect of the trajectory on the disturbance, by keeping the same start and end joint configurations as the linear trajectory, rather than relying on the IK solver's choice of solution. It also increases the repeatability of the experiment, by having very similar trajectories when running the algorithm multiple times (the only differences are caused by the randomness of STOMP, but they are relatively insignificant compared to the difference between IK solutions).

5.4.2 Findings

Table 4 shows the results obtained in this experiment. While the position deviation is slightly reduced, all other key factors increased. Since position deviation is the less important measurement (as attitude disturbance is more important), this solution is not acceptable, despite producing more predictable and safe trajectories.

Table 4: STOMP on the heavy arm with joint targets – Structure results

	Linear cartesian space trajectory	Optimised trajectory
Yaw deviation	-20.84°	-25.57°
Position deviation	1.34m	1.25m
Max linear acceleration	209.06m/s²	312.2m/s²
Max angular acceleration	28.60rad/s²	182.32rad/s²
Execution time	395.75s	1079.40s

A positive observed side-effect is also the reduction of collisions, due to simpler trajectories without extreme rotations. As illustrated in Figure 39 all joints remain within the ± 2 rad range—similar to the Cartesian path—whereas earlier trajectories extended to ± 6 rad.

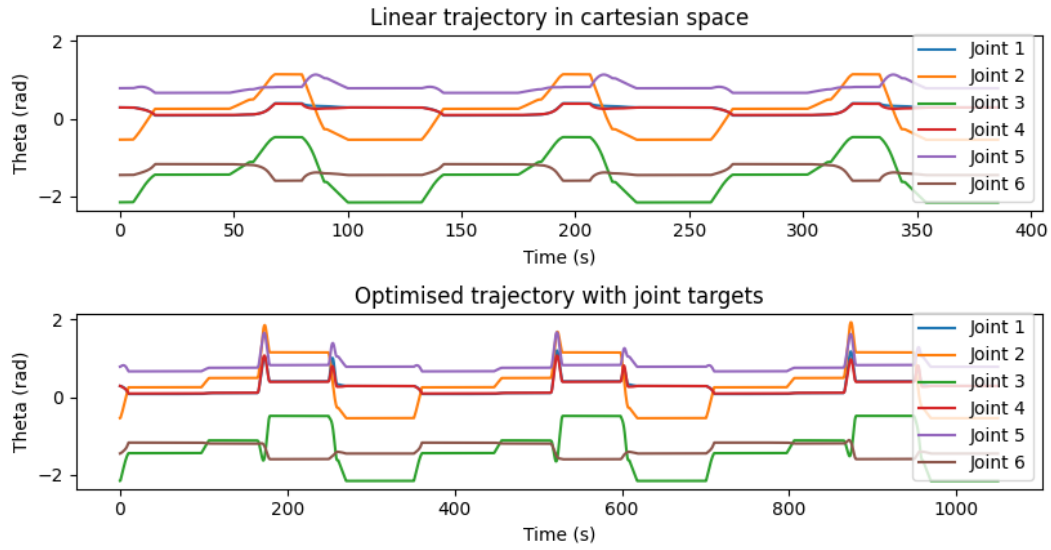


Figure 39: Left arm joint values over time when using STOMP with joint targets

However, the linear trajectory in joint space again outperformed the STOMP algorithm (Table 5), underscoring the limitations of the custom cost function in modelling the disturbance, especially when restricted to the free arm.

Table 5: Linear trajectory in cartesian space VS joint space – Structure results

	Linear cartesian space trajectory	Linear joint-space trajectory
Yaw deviation	-20.84°	-19.19°
Position deviation	1.34m	1.37m
Max linear acceleration	209.06m/s ²	134.54m/s ²
Max angular acceleration	28.60rad/s ²	68.80rad/s ²
Execution time	395.75s	279.28s

5.5 STOMP on the free arm with a cartesian initial trajectory

5.5.1 Experimental setup

Finally, to build on the previous experiment, instead of providing only the start and end joint configurations to the STOMP algorithm—which then generates an initial linear trajectory in joint space—we provide a full initial trajectory.

In this experiment, the initial trajectory is generated by the “CartesianPath” planner. As a result, STOMP starts with an initial trajectory that is linear in Cartesian space and then optimises it.

5.5.2 Findings

Table 6 shows the results of these experiments. Again, most of the key measurements show an increase (position deviation, maximum linear and angular accelerations). However, the yaw deviation, which corresponds to the structure’s attitude disturbance is reduced. As this is the main criterion, these results are promising. Additionally, this experiment outperforms the linear joint-space trajectory from the previous experiment in terms of yaw deviation.

Table 6: STOMP on the free arm with initial trajectory - Structure results

	Linear trajectory	Optimised trajectory
Yaw deviation	-20.84°	-16.87°
Position deviation	1.34m	1.40m
Max linear acceleration	209.06m/s²	627.30m/s²
Max angular acceleration	28.60rad/s²	248.73rad/s²
Execution time	395.75s	1098.68s

5.6 Summary

A series of simulation experiments were conducted to evaluate factors influencing system disturbances. Results indicate that weight distribution significantly affects disturbance levels. Moreover, trajectory modification of the manipulator demonstrated substantial potential, achieving nearly a 50% reduction in disturbances.

However, the introduction of a joint-space planner introduced multiple challenges including IK solver reliability, limited trajectory repeatability and collision risks. The best trade-off between repeatability and disturbance reduction was achieved

by bypassing the IK solver and directly providing joint positions with a Cartesian linear initial trajectory to the STOMP algorithm. Finally, the custom cost function did not effectively minimize disturbances, as the best results were obtained without using it. This outcome could either be attributed to the cost function's inadequate modelling of disturbances, or to its restricted application to only the free arm. Further experiments could answer this question.

While simulations offer useful insights into trajectory planning and induced disturbances, they cannot fully capture real-world complexities. Factors like friction variations, actuator imperfections, sensor noise, and docking misalignments can amplify disturbances in practice. Collision dynamics and computational delays further limit the direct transfer of simulation results. Thus, while simulations are valuable for development and initial testing, physical experiments are essential to validate improvements in realistic conditions.

Given more time, additional experiments could have explored different cost functions, integrated obstacles in MoveIt's planning scene, and enabled constrained planning to apply STOMP to the docked arm. These steps would then have been followed by real-world validation to confirm the simulation results.

Therefore, the current work primarily demonstrates the promising potential of trajectory modification for disturbance reduction, rather than providing a fully practical, repeatable and reliable implementation.

6 Conclusion & Future work

This thesis investigated the optimisation of multi-arm robotic motion to passively minimize satellite disturbances during in-orbit assembly. Using the MARIO platform as a representative system, the research addressed the challenge of maintaining spacecraft attitude while enabling robotic locomotion.

A comprehensive literature review was conducted to understand the locomotion challenges associated with in-orbit assembly and to evaluate motion-planning algorithms proposed for multi-arm robots. Based on these findings, a locomotion scenario was defined, and a custom trajectory optimiser was developed, implemented, and tested on the MARIO platform. The proposed approach was validated through simulation using ROS 2 and Gazebo, revealing both the strengths and limitations of the selected algorithm and providing valuable insights into their effectiveness in addressing the identified challenges.

Results showed that weight distribution strongly influences disturbance, making mass balancing critical. Changing the trajectory proved to be a promising approach, by reducing disturbances by up to 50%. However, optimised trajectory planning introduced issues such as IK inconsistency, repeatability, and collisions. Bypassing the IK solver with joint targets improved reliability, especially with Cartesian linear initial trajectories in STOMP, but did not achieve as much disturbance reduction as the basic linear trajectory in joint space.

The findings show the potential of integrating optimisation algorithms into the motion planning pipeline for space robotics but also highlight the complexity of defining a cost function characterising that disturbance. Future work should focus on refining the cost function, potentially integrating sampling-based planners like RRT* to provide a better initial trajectory, and enabling constrained planning to fully exploit STOMP's capabilities on all crawling motions. Additionally, a collaboration with the manipulator's manufacturers could be considered, allowing lower-level access to the hardware controller to implement MPC. Finally, real-world validation in the ASTRA-Lab and the exploration of cooperative strategies,

such as using the additional top arm for actively compensating the remaining disturbance, could further advance the state of the art.

Overall, this research contributes to the growing field of in-orbit assembly by providing practical insights into the interaction between robotic motion planning and spacecraft dynamics, offering a foundation for safer and more efficient space operations.

References

1. Arney D, Mulvaney J, Williams C, Hernandez WAR, Friz J, Stockdale C, Nelson J, Vargas RR. In-space Servicing, Assembly, and Manufacturing (ISAM) State of Play-2024 Edition. 2024;
2. Dorsey J, Watson J. Space Assembly of Large Structural System Architectures (SALSSA). In: AIAA SPACE 2016. Reston, Virginia: American Institute of Aeronautics and Astronautics; 2016.
3. Kulu E. In-Space Manufacturing - 2024 Industry Survey, Trends, Economics and Enablers. In: IAF Microgravity Sciences and Processes Symposium. Paris, France: International Astronautical Federation (IAF); 2024. p. 383–416.
4. Komlev VS, Parfenov VA, Karalkin PA, Petrov S V., Pereira FDAS, Koudan E V., Levin AA, Goldberg MA, Fedotov AYU, Smirnov I V., Kaprin AD, Sergeeva NS, Sviridova IK, Kirsanova VA, Akhmedova SA, Mamin G V., Gafurov MR, Gurin AN, Khesuani YD, Urlichich YM. Space manufacturing of a bone tissue destined for patients on Earth? Biomedical Technology. 2025 Mar;9:100064.
5. Cozmuta I, Rasky DJ. Exotic Optical Fibers and Glasses: Innovative Material Processing Opportunities in Earth's Orbit. New Space. 2017 Sep;5(3):121–40.
6. Saunders C, Lobb D, Sweeting M, Gao Y. Building large telescopes in orbit using small satellites. Acta Astronaut. 2017 Dec;141:183–95.
7. Oegerle WR, Purves LR, Budinoff JG, Moe R V., Carnahan TM, Evans DC, Kim CK. Concept for a large scalable space telescope: in-space assembly. In: Mather JC, MacEwen HA, de Graauw MWM, editors. 2006. p. 62652C.
8. Lillie CF. On-orbit assembly and servicing of future space observatories. In: Mather JC, MacEwen HA, de Graauw MWM, editors. 2006. p. 62652D.
9. Lee N, Backes P, Burdick J, Pellegrino S, Fuller C, Hogstrom K, Kennedy B, Kim J, Mukherjee R, Seubert C, Wu YH. Architecture for in-space robotic assembly of a modular space telescope. J Astron Telesc Instrum Syst. 2016 Jul 11;2(4):041207.

10. Rodgers E, Sotudeh J, Mullins C, Hernandez A, Gertsen E, Joseph N, Le H, Smith P. Space Based Solar Power. In: AIAA AVIATION FORUM AND ASCEND 2024. Reston, Virginia: American Institute of Aeronautics and Astronautics; 2024.
11. Glaser PE. Power from the Sun: Its Future. *Science* (1979). 1968 Nov 22;162(3856):857–61.
12. Jefferies SA, Arney DC, Jones CA, Reeves D, Moses RW, Bowman L. Impacts of In-Space Assembly as Applied to Human Exploration Architectures. In: 2018 AIAA SPACE and Astronautics Forum and Exposition. Reston, Virginia: American Institute of Aeronautics and Astronautics; 2018.
13. Collier CP, Marshall J. Next Generation Space Interconnect Standard (NGSIS): A modular open standards approach for high performance interconnects for space. In: 2015 IEEE Aerospace Conference. IEEE; 2015. p. 1–11.
14. Letier P, Siedel T, Deremetz M, Pavlovskis E, Lietaer B, Nottensteiner K, Roa Garzon MA, Sánchez Garcia J, Corella JL, Gancet J. HOTDOCK: Design and Validation of a New Generation of Standard Robotic Interface for On-Orbit Servicing. In: 71st International Astronautical Congress, IAC 2020 [Internet]. IAF; 2020. Available from: <https://elib.dlr.de/139963/>
15. Díaz-Carrasco Díaz M, Guerra G, Gala J, Viñals JJ. SIROM electronics design: Current state and future developments. *Acta Astronaut* [Internet]. 2023 Jan 1 [cited 2025 May 20];202:742–50. Available from: <https://www.sciencedirect.com/science/article/pii/S009457652200532X>
16. Deremetz M, Grunwald G, Cavenago F, Roa MA, De Stefano M, Mishra H, Reiner M, Govindaraj S, But A, Nieto I, Gancet J, Letier P, Ilzkovitz M, Gerdes L, Zwick M. Concept of Operations and Preliminary Flight Model Design of a Modular Multi-Arm Robot using Standard Interconnects for On-Orbit Large Assembly. In 2021.
17. Deremetz M, Debroise M, Boitte R, De Stefano M, Mishra H, Brunner B. MIRROR—A Modular and Relocatable Multi-arm Robot Demonstrator for

- On-orbit Large Telescope Assembly. In: 17th Symposium on Advanced Space Technologies in Robotics and Automation. 2023.
18. Rodriguez A, Romeo P, Colmenarejo I, Barrientos J, Estremera J, Lucia M, Rusconi A, Sangiovanni. Ground Validation Testing of a Re-Locatable Manipulator For On-Orbit Assembly. In: 17th Symposium on Advanced Space Technologies in Robotics and Automation. 2023.
 19. Letier P, Yan XT, Deremetz M, Bianco A, Grunwald G, Roa MA, Krenn R, Muñoz M, Dissaux P, Sánchez García Casarrubios J, Lucini R, Filippis L, Porcelluzzi G, Post M, Walshe M, Perryman P. MOSAR: Modular Spacecraft Assembly and Reconfiguration Demonstrator. In: 15th Symposium on Advanced Space Technologies in Robotics and Automation, ASTRA 2019 [Internet]. 2019 [cited 2025 May 18]. Available from:
https://www.researchgate.net/publication/336349639_MOSAR_Modular_Spacecraft_Assembly_and_Reconfiguration_Demonstrator
 20. Dillikar SR, Leslie C, Upadhyay S, Felicetti L, Tang G. A ROS-Based Control Framework for Simulating Locomotion of a Multi-arm Space Assembly Robot. In 2025. p. 93–105.
 21. Lutze JP, Schuller R, Mishra H, Rodríguez I, Roa MA. Optimization of multi-arm robot locomotion to reduce satellite disturbances during in-orbit assembly. In: 2023 IEEE Aerospace Conference. IEEE; 2023. p. 1–11.
 22. Rodríguez I, Lutze JP, Mishra H, Lehner P, Roa MA. Hybrid Planning to Minimize Platform Disturbances during In-orbit Assembly Tasks. In: 2022 IEEE Aerospace Conference (AERO). 2022. p. 1–11.
 23. Pisculli A, Felicetti L, Gasbarri P, Palmerini GB, Sabatini M. A reaction-null/Jacobian transpose control strategy with gravity gradient compensation for on-orbit space manipulators. *Aerosp Sci Technol* [Internet]. 2014 Oct 1 [cited 2025 Apr 28];38:30–40. Available from:
<https://www.sciencedirect.com/science/article/pii/S1270963814001448>
 24. Kalakrishnan M, Chitta S, Theodorou E, Pastor P, Schaal S. STOMP: Stochastic trajectory optimization for motion planning. In: 2011 IEEE International Conference on Robotics and Automation. 2011. p. 4569–74.

25. Ratliff N, Zucker M, Bagnell JA, Srinivasa S. CHOMP: Gradient optimization techniques for efficient motion planning. In: 2009 IEEE International Conference on Robotics and Automation. IEEE; 2009. p. 489–94.
26. Kavraki LE, Kolountzakis MN, Latombe JC. Analysis of probabilistic roadmaps for path planning. IEEE Transactions on Robotics and Automation. 1998 Feb;14(1):166–71.
27. Wei K, Ren B. A Method on Dynamic Path Planning for Robotic Manipulator Autonomous Obstacle Avoidance Based on an Improved RRT Algorithm. Sensors. 2018 Feb 13;18(2):571.
28. James F, Shah S V., Singh AK, Krishna KM, Misra AK. Reactionless Maneuvering of a Space Robot in Precapture Phase. Journal of Guidance, Control, and Dynamics. 2016 Oct;39(10):2419–25.
29. Noreen I, Khan A, Habib Z. Optimal path planning using RRT* based approaches: a survey and future directions. International Journal of Advanced Computer Science and Applications. 2016;7(11).
30. Dai Y, Lv W, Li S, Zong M. Improving the Lifelong Planning A-star algorithm to satisfy path planning for space truss cellular robots with dynamic obstacles. Robotica. 2025 Feb 24;1–15.
31. Santos RR, Rade DA, da Fonseca IM. A machine learning strategy for optimal path planning of space robotic manipulator in on-orbit servicing. Acta Astronaut [Internet]. 2022 Feb 1 [cited 2025 May 8];191:41–54. Available from: <https://www.sciencedirect.com/science/article/pii/S0094576521005543>
32. Doerr B, Linares R. Motion Planning and Control for On-Orbit Assembly using LQR-RRT* and Nonlinear MPC. 2020 Aug 6;
33. Neufeld MJ, Charles JB. Practicing for space underwater: inventing neutral buoyancy training, 1963–1968. Endeavour [Internet]. 2015 Sep 1 [cited 2025 Jul 28];39(3–4):147–59. Available from: <https://www.sciencedirect.com/science/article/pii/S0160932715200567>

34. Davis K, Meginnis I. Testing of the NASA Exploration extravehicular mobility unit demonstration (xemu demo) architecture at the neutral buoyancy laboratory. In 2019.
35. Bird J. Underwater at NASA's Neutral Buoyancy Laboratory [Internet]. 2015 [cited 2025 Jul 28]. Available from: <https://www.scubadiving.com/nasas-underwater-lab>
36. Deremetz M, Boitte R, De Stefano M, Mishra H, Brunner B, Grunwald G, Roa MA, Reiner M, Závodník M, Komarek M, D'Amico J, Cavenago F, Letier P, Gancet J, Ilzkovitz M, Ramisa M. Preliminary Test Results of a Relocatable Robotic Demonstrator for In-space Telescope Servicing and Assembly. In 2024.
37. Yoshida K, Nakanishi H, Ueno H, Inaba N, Nishimaki T, Oda M. Dynamics, control and impedance matching for robotic capture of a non-cooperative satellite. *Advanced Robotics*. 2004 Jan 2;18(2):175–98.
38. UFACTORY 850 User Manual [Internet]. Vol. 2.1.0. UFactory; [cited 2025 Aug 14]. Available from: <https://www.ufactory.cc/wp-content/uploads/2023/07/UFactory-850-User-Manual-V2.1.0.pdf>
39. Dobiš M, Dekan M, Sojka A, Beňo P, Duchoň F. Cartesian Constrained Stochastic Trajectory Optimization for Motion Planning. *Applied Sciences*. 2021 Dec 9;11(24):11712.
40. Rodríguez I, Quintero S, Lutze JP, Lehner P, Roa MA. Hybrid Motion Planner for a Multi-Armed Robot Performing On-orbit Loco-manipulation Tasks. In: 2024 IEEE Aerospace Conference. IEEE; 2024. p. 1–9.
41. Yu S, Han J, Qu Z, Yang Y. μ -Based Theory in Compliant Force Control for Space Docking. *IEEE Access*. 2018;6:47131–43.

Appendices

Appendix A Structure counterweights calculation

The CAD design software defines the position (in m) of the centre of gravity (Figure 40) of a single unit as

$$CG = \begin{pmatrix} x_G \\ y_G \\ z_G \end{pmatrix} = \begin{pmatrix} 0 \\ 0.30876 \\ 0.51975 \end{pmatrix}$$

3 different forces will be applied to the structure (Figure 40):

- The weight of the structure F_{weight} applied at the centre of gravity
- The maximum force F_{robot} that the robot can apply at the docking point before the electromagnet disconnects
- The weight of the counterweights : $F_{counter}$ applied at the back of the structure

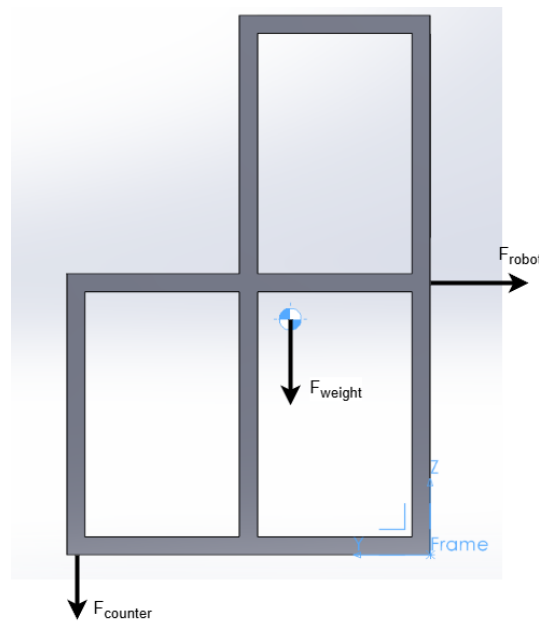


Figure 40: Structure force diagram

The numerical values of the forces are:

$$F_{weight} = 18.436 * 9.81 = 180.86N$$

$$F_{robot} = 25 * 9.81 = 245.25N$$

And $F_{counter}$ depends on the unknown mass in kg: $F_{counter} = m_{counter} * 9.81$

To prevent the structure from tipping, the moment of the forces at the tipping point (frame origin in our case) needs to be null. If we write $\vec{M}_{F/A} = \vec{AF} \times \vec{F}$ for the moment of the force F at A, we get the following equation:

$$\vec{M}_{F_{robot}/A} + \vec{M}_{F_{weight}/A} + \vec{M}_{F_{counter}/A} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad (6-1)$$

Each of these moments can be found using the following equations:

$$\vec{M}_{F_{robot}/A} = \begin{pmatrix} 0 \\ 0 \\ 0.6 \end{pmatrix} \times \begin{pmatrix} 0 \\ -245.25 \\ 0 \end{pmatrix} = \begin{pmatrix} 147.15 \\ 0 \\ 0 \end{pmatrix} \quad (6-2)$$

$$\vec{M}_{F_{weight}/A} = \begin{pmatrix} 0 \\ 0.30876 \\ 0.51975 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ -180.86 \end{pmatrix} = \begin{pmatrix} -55.8423336 \\ 0 \\ 0 \end{pmatrix} \quad (6-3)$$

$$\vec{M}_{F_{counter}/A} = \begin{pmatrix} 0 \\ 0.8 \\ 0 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ -m_{counter} * 9.81 \end{pmatrix} = \begin{pmatrix} -m_{counter} * 7.848 \\ 0 \\ 0 \end{pmatrix} \quad (6-4)$$

By substituting equations (6-2), (6-3) and (6-4) in (6-1), we get:

$$147.15 - 55.8423336 - m_{counter} * 7.848 = 0 \quad (6-5)$$

$$m_{counter} = \frac{147.15 - 55.8423336}{7.848} \approx \mathbf{11.64kg} \quad (6-6)$$

Thus, the counterweights needed for each structure need to be at least **11.64kg**.

Appendix B ASTRA-Lab demonstration - Risk assessment

Risk Assessment Number:	IRP-THS-SYS-01	Task/Activity assessed:	Crawling locomotion of MARIO		
Name / job role of people consulted during assessment:	Lucas NAURY, Robotics Student	Date of Assessment:	29/08/2025	Review Date:	29/08/2025

Acknowledgements, Sign off and Authorisation				
	Acknowledgement	Name	Signature	Date
Risk Assessor:	By signing this risk assessment I acknowledge my responsibility as the Risk Assessor for conducting this risk assessment in accordance with CU-SHE-PROC-3.01, Risk Assessment Procedure.	Lucas NAURY	Lucas NAURY	29/08/2025
Checked by: (where required)	By signing this risk assessment acknowledge my responsibility as the checker for this risk assessment in accordance with CU-SHE-PROC-3.01, Risk Assessment Procedure.	Leonard Felicetti, Saurabh Upadhyay, Gilbert Tang		
Authorising Person:	By signing the risk assessment, I acknowledge my responsibility as the department management for reviewing and approving this risk assessment and communicating controls and any additional controls to staff/students (as appropriate).	Leonard Felicetti, Saurabh Upadhyay, Gilbert Tang		

Tasks / Operational steps / Sub tasks / Events		Significant hazards:	Who is affected and how:	What are your existing controls? (Reference all Safe Systems Of Work (SSOW), Standard Operating Procedures (SOP) and Emergency Procedures)	Existing Risk Rating (Consequence x Likelihood = Total)			Are additional controls needed? Y/N (If Yes, RAMP required)
					C	L	Total	
1	Access to and working at Site	<ul style="list-style-type: none"> Physical hazards Slip / trip – cuts broken bones 	Injury to third party such as members of the public	<ul style="list-style-type: none"> Wear suitable footwear Look carefully where you are treading on uneven ground 	2	1	2	N
		<ul style="list-style-type: none"> Slips/trips 	Injury to third party such as members of the public	<ul style="list-style-type: none"> Do not leave any equipment lying around where it may cause a trip hazard Ensure that any gates/access points are locked/closed to prevent members of the public entering unsafe or non-public areas 	2	1	2	N
2	Experiment setup	<ul style="list-style-type: none"> Heavy lifting of the structure 	Back injury of the person lifting the structure	<ul style="list-style-type: none"> At least two people lifting the structure, reducing the amount of weight carried by each 	2	1	2	N
		<ul style="list-style-type: none"> Cuts 	Sharp edges of the structure can cut people that are around it	<ul style="list-style-type: none"> Wear gloves when carrying the structure Don't move the structure around other people 	2	1	2	N

		<ul style="list-style-type: none"> Heavy falling counterweights 	Foot injury of anyone there	<ul style="list-style-type: none"> Wearing safety shoes when carrying the weights 	2	1	2	N
--	--	--	-----------------------------	--	---	---	---	---

Emergency Planning Arrangements relating to operations / event

Minor cuts and stings – First Aid kit with visit leader

Minor/major accident - nearest A&E location known (see below) and call emergency services

Client Project manager details including name and contact number(if applicable)		Client Office Contact Name and Details	
Cranfield Project Manager		Cranfield Office Contact	
Fieldwork Leader (with mobile phone)		Date of Site Visit	
Fieldwork Attendee		Fieldwork Attendee	
Fieldwork Attendee		Fieldwork Attendee	
PPE Provided (e.g. life jacket, hard hat, throw rope, first aid kit)		Time Logged in	
		Time Logged out	