Movement of Modular Hydraulic Propulsion robots: Decentralised and reactive pose control strategies

João V. Amorim Marques

Supervisors: Dr. Roderich Groß
Dr. Bryn Jones

Department of Automatic Control and Systems Engineering
The University of Sheffield

This dissertation is submitted for the degree of
Doctor of Philosophy

September 2020

Abstract

Modular robotic systems consist of a set of reconfigurable units, called modules, which can be combined in a multitude of ways to produce robots of different shaps . One of the challenges in the design of these system is to enable them to perform precise movements in their environment. Control strategies that are centralised or rely on external sensing can limit the robustness and scalability of the system. This thesis focuses on the development of control strategies that allow the position and orientation (pose) of a modular robot to be controlled in a fully decentralised and reactive manner. The strategies are designed for the Modular Hydraulic Propulsion (MHP) system, wich operates in a liquid environment. An MHP robot is made of cubic modules, which create a fluid network when connected together. To move, the robot routes through this network fluid from the environment. A physical implementation of the MHP concept is designed, built and validated. An MHP robot's ability to translate efficiently towards a goal is tested using occlusion based controllers, both with and without communication between modules. The robot is shown to reach the goal using either of the controllers. When using communication, an average of 70% of energy is saved, at the cost of a longer completion time. This thesis proposes multiple minimalistic controllers to control the pose of MHP robots. The robot is required to reach a goal in a preferred orientation. All of the controllers use binary sensing and actuation, with each module using only two bits of sensory information per face. The controllers are proposed for robots moving in 2D and 3D space, and use up to five bits of communication between modules. We prove that robots of convex shape are guaranteed to complete the task. Using computer simulations, the controllers are tested in different environments, using multiple module sizes and under the effect of noise. Additionally, their performance is compared against a centralised controller from the literature. Given the simplicity of the solutions, modules could potentially be realised at scales below a millimetre-cube, where robots of high spatial resolution could perform accurate movements in liquid environments.

Declaration

I hereby declare that, except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

João V. Amorim Marques
September 2020

Acknowledgements

I would like to begin by thanking my supervisor, Dr. Roderich Groß for his constant guidance and support during my PhD studies. The scientific expertise and knowledge he unreservedly passed on to me were essential in my transition from student to researcher. He pushed me to become critical of my own work though his constant feedback. This PhD would not have been possible without him.

I would also like to thank all the members of the Natural Robotics Lab, past and present, that made me feel welcome from day one. To Chris, Fernando, Yuri, Gabriel, Stefan, Matt Doyle, Anıl, Nicole, Yue, Isaac, Matt Hall and Federico, a big thanks you for making me feel at home. I would also like to thank other office-mates and colleagues. To Salah, Mark, Fadl and Aldo, for the all the advice and support. To James, Matthew Watson, Matt Doyle and Haoyu, for the long lunch breaks and random chats. They kept me from pulling my hair whenever I was stuck with work.

To all those that collaborated in my work, I owe you a large debt of gratitude. To Matt Doyle and Anıl for all the help with the MHP the long hours of discussions and feedback.

I would also like to thank all my friends from Sheffield, who where there in the good and bad moments. In particular, Gabriel, Errol and Vino, for the guidance and support in the though times and celebrations in the good. Your company kept me sane and in the right path.

Um grande obrigado à malta portuguesa que me acompanhou nest percurso. Em particular, ao André, Cristiana, Sofia, Pedro e Patricia. Os nossos momentos juntos foram como ter um pouco the Portugal connosco. Ao Filipe, Rita, Tânia e Catarina, obrigado por, mesmo estando longe, estarem disponivies partilhar os bons e maus momentos.

Por fim, e como não podia deixar de ser, quero agradecer à minha familia. Foi o vosso suporte e apoio que me permitiu chegar onde estou hoje. Em especial, quero agradecer aos meus pais que não só tornaram este doutoramento possivel, mas também me apoiaram constante e incondicionalmente durante estes anos. A eles dedico este trabalho.

To my parents, who always encouraged me to keep going further.

—

Para os meus pais, que sempre me incentivaram a ir mais longe.

Table of contents

List of figures

List of tables

Nomenclature

Sets

$\mathcal{A}$	Robot
$\mathcal{E}$	Environment
$\mathcal{O}$	Preferred orientation

Subscripts

e	Index for module number
i	Index for world dimension
j	Index for module/robot face

Vectors

$\mathbf{f}$	Total force acting on the robot
$\mathbf{g}$	Goal position
$\mathbf{r_{ej}}$	Position of p_j for module e
$\mathbf{s_{ej}}$	Position of goal sensor of face j for modules e
$\mathbf{u_j}$	Normal vector to face j
$\mathbf{v_e}$	Position of module

Other Variables

ε	Module's side length

ρ	Generic density
τ	Torque acting on the robot
a_i	Robot dimension along axis i
b_k	k-th shared power line
c_j	Binary state of connection sensor of face j
d_j	Binary state of goal detection sensor of face j
f_p	Force of a single actuator
$h_t(\mathbf{x})$	Rigid body transformation from the world reference frame to the robot reference frame
M	Number of shared power lines
m	Generic mass
N	Number of world dimensions
n	Number of modules
p_j	Binary state of pump of face j

Chapter 1

Introduction

Robotics is a research field with vast potential for improving living standards. Day by day, the presence and impact that robots have in our lives are increasing. By allowing automation of tasks, robots can reduce some of the workloads and risks humans are subject to in their daily routines. Additionally, the use of robots to perform repetitive tasks in a reliable manner allows for the completion of these tasks in a faster and more efficient way. From automated vacuum cleaners to manufacturing processes, the range of the tasks robots are able to perform is broadening. With the increase of areas where robots become essential commodities, research into new robotic systems and control strategies that can complete more varied tasks in an efficient manner has gained more relevance.

With the increased need of efficiency in the completion of tasks, traditional robotic systems tend to become more specialised. Traditional robotic systems have a fixed morphology. This allows them to perform a single or small set of tasks, in well defined environments, efficiently. However, a fixed morphology limits their adaptability changes in the tasks or environment. Additionally, traditional robotic systems usually contain a single point of failure. Therefore a fault in a single component of the system can lead to failure in the completion of the task. To overcome these challenges, research has been performed to increase the adaptability of robotic systems, as well as their robustness to failure, even though this may come at a cost of efficiency.

A concept that has the potential to lead to increased adaptability and robustness is modularity [1, 2]. Modularity is widely observed in nature, and at different levels. Societies are composed of living being, living being are comprised of organs, organs are comprised of cells and even cells are comprised of organelles. Applied to robotics, the concept of modularity involves using groups of robots, rather than a single robot, to complete tasks. As a result, the failure of a single robot does not compromise the completion of the task. Additionally, through the use of multiple robots, cooperative behaviours can be used, which have the potential of completing a larger range of tasks or function in a larger range of environments.

One possible implementation of the modularity concept is swarm robotics. In swarm robotics [3], a group of simple robots work together to complete tasks. The inspiration for these systems comes from swarms present in nature, such as colonies of insect or flocks of birds, which are able to solve collectively tasks that can be too complex for a single individual [4].

Modular robotics is another implementation of the modularity concept [5–7]. Modular robots consist of groups of units, denominated modules, connected into a single structure [8–10]. Each module in the system tends to have limited capabilities. However, by arranging the modules in different configurations, a modular robot could be able to demonstrate capabilities that go beyond a single module [11]. In modular systems, a module can be considered as the equivalent of a cell in a multicellular organism [5, 12]. In multicellular organisms, groups of cells can form organs, that are responsible for specific roles in the survival of the organism. Similarly, modular robots can be assembled into different morphologies, allowing for different capabilities of the system [13, 14]. This potentially allows modular robots to tackle a wider range of tasks than traditional robots. It can also provide robustness to the system, as the failure of a single module does not necessarily cause the failure of the robot. Additionally, in the same way a living organism can replace malfunctioning or dying cells, broken and/or malfunctioning modules can be replaced without having to replace the whole system [8, 9, 15].

1.1 Motivation

The use of robots in environments that are not easily accessible to humans can be beneficial. The benefits can be in terms of safety (e.g. toxic or structurally unstable environments) or cost (e.g environments to which access requires structural changes to the surroundings). As a result, research into robotic systems able to operate environments that present challenges to human presence is gaining relevance [16]. Liquid environments are an example where human presence might be constrained. For instance, the exploration of deep water environments involves the use of breathing apparatus or submersibles capable of withstanding high pressures. Another example are water pipelines, which can be too narrow for a human to access or navigate. The use of robots can improve the ease of access and completion of tasks in these environments [17–20]. However, it is not feasible to predict the full extent of tasks a robot might need to deal with in these situations. Adaptability can therefore be required to solve tasks these environments. The versatility of modular robots makes them potentially better suited for these situations. Modular robots are able to change their shape. As a result, modular systems have the potential to deal with a variety of situations. Additionally, some systems are capable of rearranging their structure independently, which further increases their adaptability. For example, the modules of the robot could disassemble and travel independently through a narrow passageway and then reassemble to restore the robot's original form.

In some applications, the ability of modular robots to move in an efficient manner as a unit might be crucial. For example, a robot may need to navigate a passage without colliding with the walls, or reorient itself to inspect or manipulate objects [21]. These tasks require precise control of the robot's pose (position and orientation). Moreover, the remote operation of the robots might not be viable in certain tasks or environments [17, 22]. As a result, the control of the robot's pose should be performed in an autonomous manner. Even though the capability to control a robot's pose has been demonstrated with traditional robots, realising them at the level of a large ensemble of modules remains a challenge. Due to their reconfiguration capabilities, modular robots can present multiple actuator configurations. Moreover, in aquatic environments, some additional challenges exist. For example, in the presence of turbulent flow, the controllers need to be able to react the fast changing conditions of the environments [22]. As a result, the controllers might also need to demonstrate some level of adaptability, to cope with these situations.

The miniaturisation of modular systems can be advantageous. For example, in some environments might only be accessed by small sized robots. Additionally, the miniaturisation of the modules allows for robots with a higher modular resolution. Increased modular resolution gives rise to the potential for fine grain response from the robot, which could increase the precision of robot's movement. The miniaturisation of modular systems could pave the way for new applications, such as micro-medicine [23–25]. However, the miniaturisation of modular systems poses some challenges. For example, miniaturised modules impose restrictions on the hardware resources available to each module. In turn, the restricted hardware resources limit the complexity of the controllers that can be applied. As a result, a minimalist approach to the development of the control strategies used by these systems might be required. Decentralised reactive control solutions can be advantageous in these situations. Not only do these solutions allow for independent action from each module, they also require little computation resources from each module. Additionally, the minimalistic approach can lead to low energy consumption. Since in small scale robots the energy available is limited, this allows for extending the operation time of the modules, which can be advantageous.

The Modular Hydraulic Propulsion (MHP) robot was proposed by Doyle *et al.* [26]. It is a modular system, where movement is generated by routing fluid through and between modules. Each module uses only binary actuation and sensing. Hence, the miniaturisation potential of the system is high. Additionally, a robot comprised of a large number of modules presents the potential for precise movement, due to the high number of unique actuation states it is capable of.

1.2 Problem definition

The main focus of this thesis is the study of the MHP system and, in particular, its movement capabilities. Each MHP module is actuated by pumps that move fluid between the modules and the environment. The large reconfiguration space potential of the system can allow a large variety of robot configurations. The scalability potential and limited hardware capabilities of the system present advantages for system miniaturisation. An initial hardware prototype of the MHP robot was built in [26], capable of moving on the water surface. However, only the translation capabilities of the system were demonstrated, through the use of an occlusion-based controller. Additionally, the control policy used, and due to limited capabilities of the developed hardware allowed some active actuators to have no impact on the robot's movement. This leads to energy being consumed with no benefit to the robot's movement. As a result, to allow for a more efficient control of the robot's movement, the design of a new system is required.

To achieve precise movement of a robot, control of both position and orientation is required. This thesis investigates the problem of controlling the pose of an MHP robot. In particular, we focus on the development of control strategies that allow an MHP robot to move towards a goal with a desired orientation. This task is to be solved in a liquid environment, containing a goal and a robot. The control strategies to be developed should allow for the pose of both 2D and 3D robots to be controlled.

We are interested in decentralised and reactive control strategies. Centralised control strategies can either introduce a single point of failure in the system, or require all modules to have the computational capabilities required to control the system, so that any module can take control of the system in case of control unit failure. Additionally, control strategies that require external sensing can be impractical for modular systems, as they might not scale easily with the number of modules used in the robot. The control strategies to be developed should generate efficient movement from the robot, while taking into account the modules' limited resources, particularly in terms of sensing and actuation. As a result, reactive control strategies, that is strategies where no deliberation is required, as the sensing state gets directly mapped onto the actuation state, are an attractive solution, as they require no run-time memory and perform little arithmetic computation.

Starting from the problem of translating towards a goal, irrespective of orientation, we build on the existing knowledge of the system to investigate what are the required resources in order to control the robot's orientation. For robots moving in 2D, developed control strategies should be able to control the robot's 3 degrees of freedom. For robots moving in 3D, developed control strategies should be able to control 5 degrees of freedom of the robot. While in the literature there are some works that look into the precise movement of modular robots [27], they are centralised and require the use of complex sensing capabilities and significant computational resources. Additionally, as we are interested in miniaturisation, an analysis between the trade-offs between performance and complexity of controllers (even if requiring little computational resources) could prove useful when designing a system for a particular application.

1.3 Aims and objectives

In light of the aforementioned problem definition, this thesis main aim is twofold: to design and build a physical MHP system, and to develop decentralised reactive control strategies for the effective pose control of an MHP robot.

The detailed objectives of this thesis are:

- To develop a fully revised physical implementation of the MHP concept. This will require an analysis of the original hardware prototype. The new MHP implementation should allow modules to communicate and improve the reliability of the modules. The hardware will be designed, built and validated.

- To study the impact communication has on the performance of physical MHP robots. In particular, the differences in energy consumption and power usage will be studied.
- To develop decentralised pose control strategies for MHP robots. The control strategies will use different amounts of information sharing between modules, to account for different possible hardware limitations. They will be formally analysed for MHP robots of different shapes. Possible limitations will be highlighted.
- To validate and compare the performance of the control strategies developed using multiple environment conditions and scenarios, including physical experimentation. The results of the comparisons between the different controllers, for each environment and scenario tested will allow for an analysis of the trade-offs between the performance and resources required for each controller.

1.4 Preview of contributions

This thesis presents the following contributions:

- A hardware implementation of the MHP concept (MHP 2), with improved reliability and capabilities. This implementation includes a complete redesign of the module's chassis and circuitry. The extended capabilities include range sensing, communication between modules, capability to receive external messages and determine each module's orientation. Additionally, the mass and volume of each module are reduced by 73% and 63%, respectively.
- An average reduction of 70% of the energy consumed by the MHP 2 robot, in a translation task when using communication. This reduction comes at the expense of a 95% increase in average task completion time.
- An analysis of the range of unique total forces and net torques that can be produced by a 3D convex MHP robot, based on different actuation states. The analysis considers forces and torques along one or more of the robot's reference frame axes.
- A set of decentralised reactive controllers for pose control of 2D and 3D convex MHP robots. The controllers allow the modules to exchange between 0 to 5 bits of information via shared power lines.

- A formal analysis of all of the proposed pose controllers for 2D and 3D convex MHP robots. The controllers that share information are proven to complete the task in finite time, if in their initial position they are able to rotate in place without colliding with the goal. The controllers that do not share information are shown to require a larger initial distance to the goal in order to complete the task.

- A formal analysis of the pose controllers for MHP robots of generic shape. A set of counter-examples is used to show the limitations of the controllers when applied to non-convex robots. The controllers are shown to be correct for orthogonally convex robots, without requiring adaptation.

- Validation of the pose controllers in physical simulations, using 3D convex MHP robots. Studies evaluating the performance of the controllers for different environments, initial positions, robot modular resolution and noise levels in actuation and sensing, between others, demonstrate the usability and limitations of each controller, with the trade-offs between performance and resources described.

- Validation of the pose controllers in physical experiments, using a 2D convex MHP robot. The results are discussed in light of the results obtained for simulations in 3D environments.

1.5 Publications

This thesis includes original contributions to scientific knowledge. The work presented has led to the following publications:

- **J. V. A. Marques**, A. Özdemir, M. J. Doyle, D. Rus and R. Groß. "Decentralized Pose Control of Modular Reconfigurable Robots Operating in Liquid Environments", i*n Proceedings, 2019 IEEE International Conference on Intelligent Robots and Systems, IEEE, in press*

- M. J. Doyle, **J. V. A. Marques**, I. Vandermeulen, C. Parrott, Y. Gu, X. Xu, A. Kolling and R. Groß. *"Modular Fluidic Propulsion Robots", conditionally accepted in IEEE Transactions on Robotics*

1.6 Thesis outline

The remainder of this thesis is organised as follows:

- Chapter 2 provides a review of the relevant literature to place the work in this thesis in context. This starts with a review of modular robotics concepts and existing systems in Section 2.1, with a focus on systems that work in fluid environments. Section 2.2 covers the current control algorithms that allow a modular robot to control its movement.
- Chapter 3 presents the development process of the MHP 2 and a study on the impact of the addition of communication to an occlusion based controller. Section 3.1 introduces the MHP concept. In Section 3.2 the list of requirements for a physical implementation of the MHP are presented. Section 3.3 presents an analysis of the initial hardware prototype is performed, with the limitations and main areas of improvement being presented. Section 3.4 covers the revisions made to the system. Both the circuit and chassis designs are described in detail. Section 3.5 details the experiments conducted with the MHP 2, the implementation of the controllers used and the results obtained. Section 3.6 summarises the chapter.
- Chapter 4 develops the new control strategies for pose control of an MHP robot. Section 4.1 introduces the chapter. An analysis of the number of actuation states that generate unique total forces and net torques for a 3D convex MHP robot is conducted in Section 4.2, for robot translation and rotation. Section 4.3 defines the problem to be solved, including the objective. Controller solutions for the problem described are presented in Section 4.4, both for 2D and 3D convex robots. The mathematical analysis of the proposed controllers for convex robots is performed in Section 4.5, and extended for robots of generic shapes in Section 4.6. Section 4.7 summarises the chapter.
- Chapter 5 validates the controllers, for robots moving in 3D, through the use of physics simulations. Section 5.1 introduces the chapter. Section 5.2 describes the simulation studies performed, including simulator, controller implementation, default simulation set up and results for 3D robots. Section 5.3 presents the validation performed through physical experiments, describing experimental setup, controller implementation and results. Section 5.4 summarises the chapter.
- Chapter 6 summarises and discusses the contributions of the thesis, and proposes potential avenues for future research in Section 6.1.

Chapter 2

Background and Related Work

This chapter reviews the current literature relevant for the work presented in this thesis. In Section 2.1, a review of relevant modular robotics concepts and modular systems is presented. The focus is on systems that work in fluid environments, as they have the potential to move in 3D, controlling up to 6 degrees of freedom (DoF). While our main interest lies in systems that work in liquid environments, aerial systems operate in fluidic environments and present the same number of controllable DoF. Some of the most relevant land based systems are covered for contrast. Section 2.2 presents a review of relevant control strategies for modular systems. The focus is on control strategies that allow for the control of the movements of modular robots as a unit. Some multi-robot control strategies, which aim to generate object transportation behaviours, are also presented.

2.1 Modular robotics

Over the last decades various modular systems have been developed [5]. However, modular robotics as a research field still faces various challenges [8, 12]. Although this thesis is concerned with modular systems that work in liquid environments, the analysis of systems that operate in other environments can provide insight into the trade-offs required in the development process. For example, land based systems are able to prioritise mobility (as units or as a robot), scalability or reconfiguration space. However, due to limitations of the environment, it is uncommon for all of these characteristics to be prioritised in a single system. Additionally, the movement these systems can perform is often constrained to 2D. On the other hand, aerial systems are able to move in 3D. However, they are usually limited on their reconfiguration space. Systems for liquid environments have the potential for 3D movement without compromising their reconfiguration space.

Modular robotic systems can be classified in different architecture types, based on the type of structures the modules are able to form [6, 10]. The most common architecture types are:

- **Chain architecture:** the modules are only able to form linear or branching structures [28]. This is a result of the position and number of connections each module is able to establish. Systems of this architecture are versatile, being potentially able to reach any point and orientation, due to their large number of controllable DoF. However, while the kinematics of these systems is well understood, on-line control is challenging. The large number of controllable DoF leads to computationally expensive control strategies. Additionally, hardware inaccuracies can accumulate through the chain.
- **Lattice architecture:** the modules are able to form 2D or 3D grids, by allowing each module to connect to multiple modules at the same time [29]. For example, in 2D, square and hexagonal lattice structures are possible. In 3D, multiple cubic lattices exist: simple (modules are the cube vertices), body-centred (one module in placed at the center of the cube) and face-centred (modules can be placed either in the vertices or aces of the cubic structure). The regularity of the grids these modules can form represents an advantage in reconfiguration processes, as the positions each module can occupy are discrete.
- **Mobile architecture:** the modules are capable of efficient self-locomotion [30]. This allows them to either work individually, in groups of individual units, much like a swarm, or as a single, more complex structure. While the individual movements of the modules allows for more versatile reconfiguration, it is also more complex in terms of control.

If more than one of the above architecture types is present in a system, it is denominated as an hybrid architecture [15].

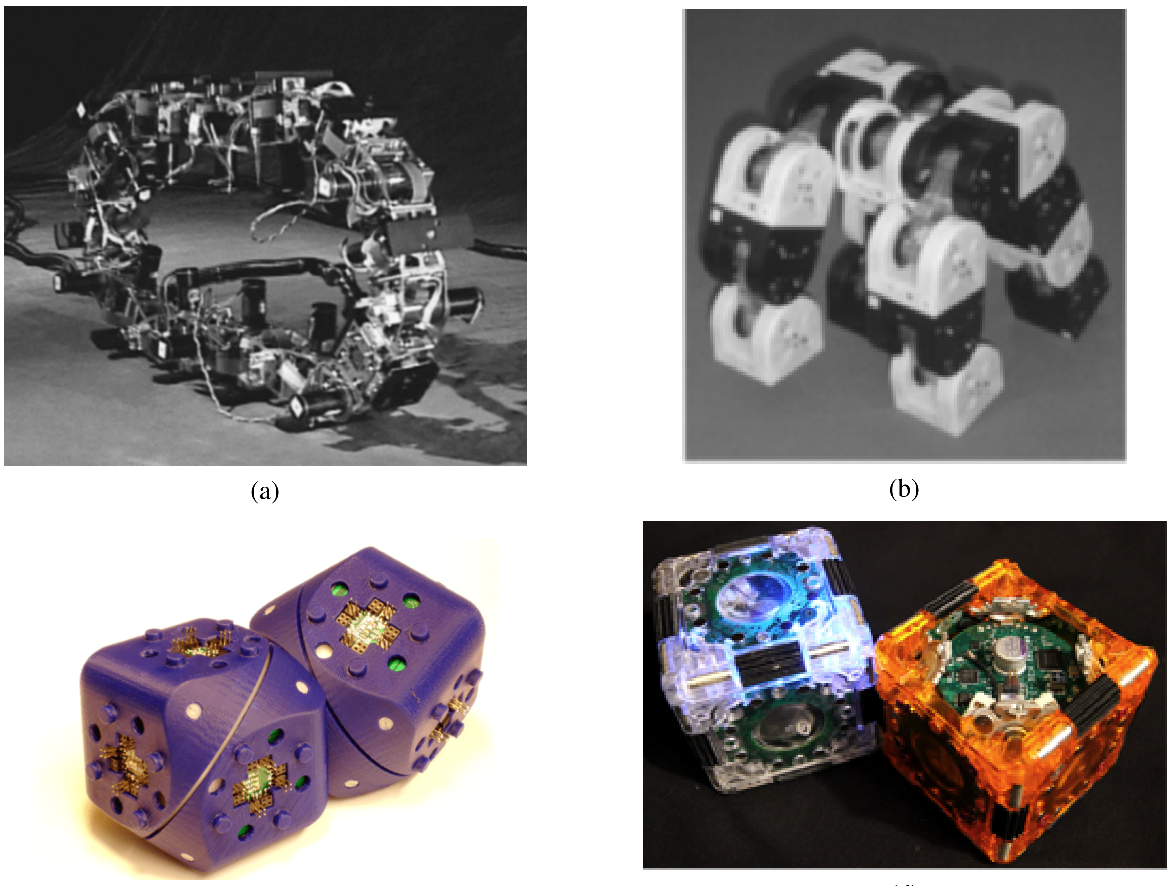
Additionally, modular systems can be divided into self-reconfigurable and manually reconfigurable systems [1]. Self-reconfigurable systems are able to perform changes in the robot's configuration without the intervention of outside help. The reconfiguration process can either be deterministic or stochastic [8]. In deterministic reconfiguration processes each module is given a specific position it should occupy. Based on the module's initial and final position, a trajectory is planned. As a result, the position of all modules is either known at all times, or it can be computed. On the other hand, in stochastic reconfiguration the robot only has information about the positions of the modules when attached to the main structure. The movements of the modules, once disconnected, use stochastic processes (e.g., Brownian motion) and are unknown to the robot. While the reconfiguration times can be guaranteed using deterministic reconfiguration, the process does not scale well, as an increased number of modules requires increased computational power.

2.1.1 Modular land based systems

In recent years many modular and reconfigurable systems have developed. Arguably the first modular system to be developed was the CEBOT in 1988 [31]. The system was designed with industrial applications in mind. Each module is capable of independent movement and connection to the structure. The initial system envisioned modules self-organising in a container, with the structure being dependent of the task at hand. Approximation, alignment and connection between two modules was experimentally demonstrated by the authors in [32]. Additionally, an algorithm for determining optimal structure based on task was developed and tested [23, 33].

The Polypod system [37, 38] was developed by Yim *et al.*, and is capable of forming long chains. The system is comprised of two module types: nodes and segments. Node modules are rigid cubic structures and are passive modules, housing the system's power source. They are able to establish up to six connections with other modules. The segment modules are actuated, with 2 DoF, that are able to perform both rotation and translation (through compression and expansion). They only allow for two connections with other modules. Robots are formed by connecting chains of segment modules to node modules. This allows for multiple configurations to arise. One such configuration is the rolling track, that in later works was shown to be a viable method of locomotion [39]. The Polypod system was later followed by the Polybot system [28, 40]. This system exchanged the Polypod's ability of compressing and expanding for an increased rotation range (Figure 2.1a).

The M-TRAN system [15] is a hybrid system, that has become highly influential in the modular robotics field. This modular system has had three iterations in its development, making it one of the most well developed modular systems [34, 41, 42]. Each module in this system was composed of two connected cube-like blocks. The blocks are connected



(a) (b) (c) (d)

Fig. 2.1 Examples of modular systems. (a) PolyBot robot configured in a rolling chain. (b) MTRAN-III performing quadruped walk. (c) Two Molecubes modules joined together. (d) A pair of 3D M-Blocks modules. Images reprinted from [28], [34], [35] and [36], respectively.

by a rigid bar that acts as a rotational joint at the points of connection. Each cube is able to connect with up to three other modules. The design of the system allows it to form both chain and lattice configurations. The last iteration of the system, M-TRAN III, demonstrated quadruped walking (Figure 2.1b), crawling of snake configurations and self-reconfiguration between the two configurations. [34].

The SMORES system [43] is another interesting hybrid system, as it can form both chain and lattice structures, with the modules being able to move independently. Each module is a cube and possesses 4 DoF. Provided a surface that a connected structure can attach to, the system is able to move in 2.5D.

A Molecube [35] module consists of a cube that is divided in two halves along its longest diagonal. Each unit contains a single actuator that allows rotation between the halves to occur.

The initial design featured only two connections, meaning only chain structures could be formed. The connections between modules were active, allowing for self-reconfiguration. In later versions [44, 45], the two active connections were replaced with six passive connections. As a result, the system traded its reconfiguration capability with a larger reconfiguration space(Figure 2.1c).

The Roombot [46] is a system based in a similar design, where two split cubes are permanently connected together by a rotation joint. The system was envisioned to be able to create adaptive furniture that would change based on the needs of the user [47, 48]. Later work demonstrated the movement and reconfiguration capabilities of the system [49].

M-Blocks [50] are cubes able to form lattice structures. The modules are able to move and reconfigure based on a novel motion system that uses flywheels to build up angular momentum. To generate movement, the flywheels undergo rapid braking to create a high torque that flips the module. This can be used both to move the module in the environment or to move the position of the module in the robot's configuration. A version capable of generating 3D structures is presented in [36](Figure 2.1d)

The ATRON system [29, 51] is also able to form lattice structures in 3D. A module is made up of two hemispheres, connected by a rotational joint. While the system has minimalistic actuation, it is highly reconfigurable. Movement of the robot can be performed by using the modules as wheels or by self-reconfiguration.

The HyMOD [52] is another example of a system able to form 3D lattice structures. Each module is able to connect up to four modules, by making used of one HiGen connector [53] per face. The modules consist of two symmetrical halves, connected by a rotation joint. Each module has 3 rotational DoF, which allows the modules to rotate in place, while performing reconfiguration. Additionally, the modules are able to move in an environment individually, through the use of two connectors able to rotate as differential drive wheels. Multiple extensions for the system were developed, where each extension is considered a different type of module [54].

A Swarm-bot [30, 55] consists of multiple robots (s-bots) connected together in a single structure. Each unit can move by using differential drive wheels and tracks. This allows for both accurate movements and rough terrain navigation. Each s-bot is equipped with a gripper that allows connections between the units to be formed. Both self-assembly and cooperation between s-bots has been demonstrated [56, 57].

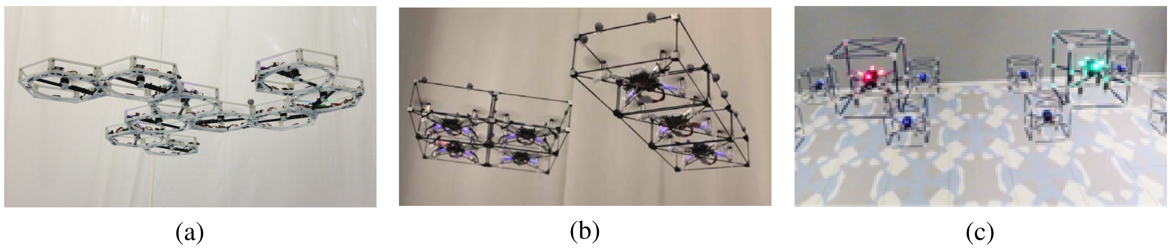


Fig. 2.2 Examples of aerial modular systems. (a) Distributed Flight Array in flight, in a configuration with 10 modules. (b) ModQuad system in flight, in two different configurations. (c) Group of GridDrones in coodinated flight. Images reprinted from [63], [64] and [65], respectively.

Calytronics [58] is a system of modules, designated as catoms (claytronic atoms), that is inspired by the concept of programmable matter [59]. A group of catoms is able to self-organise into multiple shapes. An initial prototype was developed, consisting of circular centimetre sized modules able to reconfigure in 2D. In [60] a set of 1 mm catoms, that are able to roll over each other in order to reconfigure was developed. A new, quasi-spherical catom design was proposed in [61], capable of efficient movement between grid positions.

The Robot Pebbles [62] are cubic modules, with 1 cm side length. While being impressively small, the modules are able to include a connection mechanism, communication and power sharing with connected modules in each face. On the other hand, the modules are unable to propel themselves and reconfiguration capabilities are limited to passive subtractive reconfiguration.

2.1.2 Modular aerial systems

Aerial systems are able to move in 3D, and generally possess 6 DoF. However, compared to systems for other environments, they have a limited reconfiguration space, generally only being able to form 2D structures. In this section, we describe some of the most relevant aerial systems developed.

The Distributed Flight Array (DFA) [66, 67] is a reconfigurable modular system, with hexagonal modules, capable of forming 2D structures. Each module has a set of omnidirectional wheels that allow it to move on the ground to connect with other modules. The modules use permanent magnets to form connections. The magnets ensure the robot stays connected in flight, but allow for modules to disconnect by pulling away from each other on the ground. Every module contains a rotor, that spins either in a clockwise direction (CW module) or in a counter-clockwise direction (CCW module). In order for a robot to be able to take flight it needs to be composed of two pairs of modules of each type (CW and CCW modules). In [63], structures with a higher number of modules were shown to be able to take flight (Figure 2.2a).

The ModQuad system [64] uses cubic modules that are able to form 2D structures while flying. Each module is a quadrotor encased in a cuboid frame (Figure 2.2b). The sides of the vertical faces of the modules have four permanent magnets that allow the modules to form connections. As each module is a functioning quadrotor, they are able to fly as individual units. As a result, a structure capable of flight has no required minimum number of modules. A later version of the system [68] makes use of vision-based docking methods to reconfigure in outdoors environments during flight.

Naldi *et al.* [69] propose a modular aerial robot that uses ducted-fans (fans surrounded by tubes) as modules. The lift of the modules is controlled through a set of actuated vanes placed on the ducted fans. In contrast with other systems, the design of this system allows for reconfiguration in 3D, while maintaining the ability to fly. While no specific connection mechanism was proposed, it was assumed to be rigid for the analysis of the system. In [70], experiments using two horizontally connected modules were conducted.

The GridDrones system [65] aims to implement the programmable matter concept in an aerial environment. Each unit is composed of a nanocopter surrounded by a cubic structure. The system is designed to form 2.5D relief maps. While the units do not connect with each other, topological relationships between the nanocopters can be assigned on the fly. This allows the maps to be altered by the operators through physical interaction with the nanocopters.

2.1.3 Modular systems in liquid environments

When compared to land-based and aerial systems, systems operating in liquid environments have the potential to show versatile 3D movements, coupled with a 3D reconfiguration space. However, while developing modular systems for these environments present advantages, there are also challenges that need to be tackled. Examples of these challenges are the waterproofing of the systems, and potential difficulties liquids can cause in communication.

In this section, we cover modular systems that are designed to operate in liquid environments. These systems are divided into two groups: self- and externally-propelled systems.

Self-propelled systems are able to move, without the intervention of external elements. On the other hand, externally-propelled systems use external intervention to generate their movements, for example fluid mixing. In this thesis we are interested in self-propelled systems, due to their ability of controlling their own movement. Externally-propelled systems are presented for comparison.

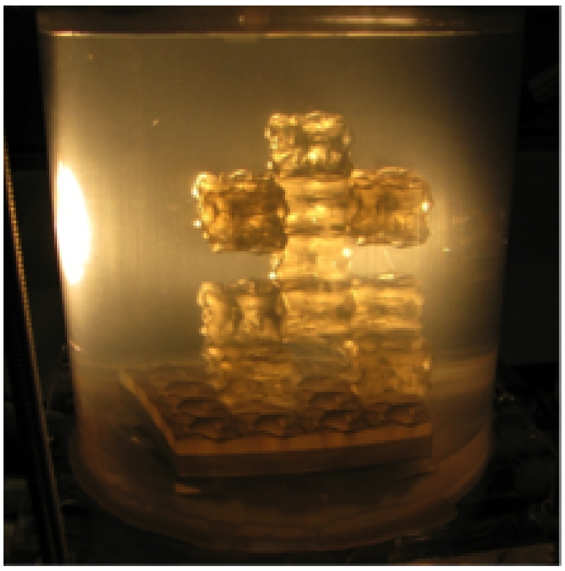
2.1.3.1 Externally propelled systems

White *et al.* [71] propose a modular self-reconfigurable system, based on an initial concept presented in [72]. The modules are neutrally buoyant, unpowered cubes that are immersed in an oil tank, and form connections through the use of passive magnets, along with an electromagnet. A substrate that acts as a base for self-reconfiguration is placed on the bottom of the oil tank. Additionally, it is able to power connected modules. Each module has a pipe network connecting the module faces that allows fluid to move through the module. Each module face has valve that allows or prevents the fluid flow through that face.

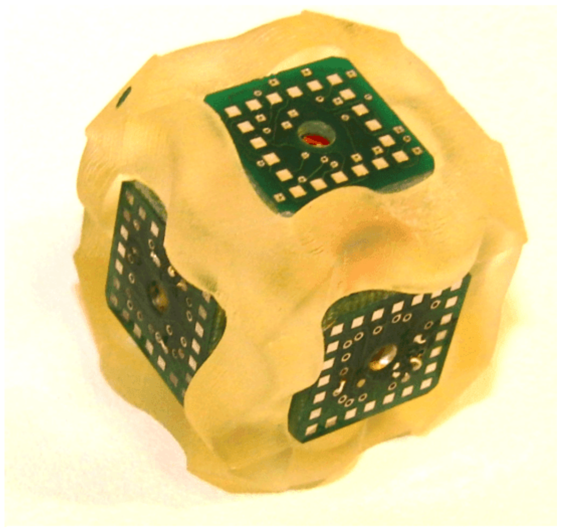
Tolley *et al.* [77] presented a system of smaller size, based on the the same concept (Figure 2.3a). The modules are completely passive, and move in response to the fluid movements of the oil tank. A substrate placed in the bottom of the tank pumps the oil, generating a pressure gradient. As the fluid moves through the modules and substrate, the pressure differences act as a connection mechanism between modules and between the modules and the substrate. In later works [73], the modules have valves that can be closed or opened to selectively form connections between modules and with the substrate. The valves in the modules are powered by the substrate, when connected to a structure.

Neubert *et al.* [74] present yet another self-reconfiguring system based on the same concept (Figure 2.3b). In this system, instead of a substrate, there is a powered seed module, capable of computation, from which the structure is formed. The modules are brought close to the seed by the movement of the fluid flow. If modules are correctly aligned with the seed (face-to-face), a connection can be formed by soldering the modules together, on the fly. As the only element in the system capable of computation, the seed is responsible for accepting or rejecting connections.

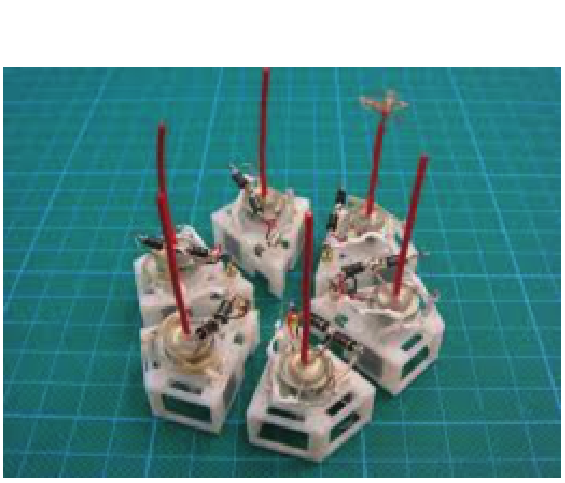
The Tribolon [75] is yet another system incapable of predefined motion. These modules move in a motion close to Brownian motion, from the action of a vibrating motor, placed on top of each module. The modules are externally powered and are brought together by a magnet placed at the bottom of each module. Each module has two electrodes: a cathode



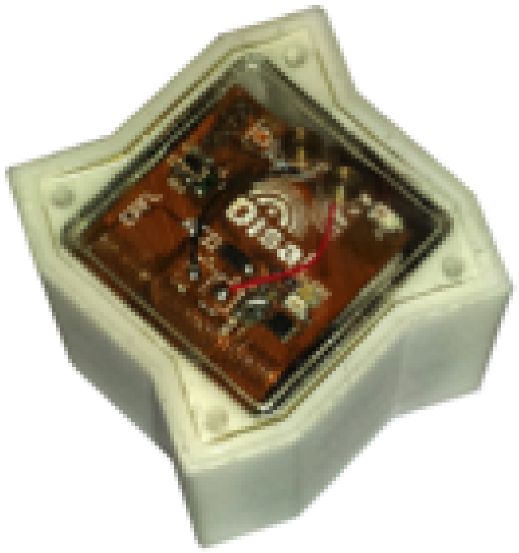
(a)



(b)



(c)



(d)

Fig. 2.3 Examples of externally-propelled aquatic modular systems. (a) Configuration of 10 passive modules proposed by Tolley *et al.* (b) Passive module of Neubert *et al.* (c) Six Tribolon modules. (d) Lily module. Images reprinted from [73], [74], [75] and [76], respectively.

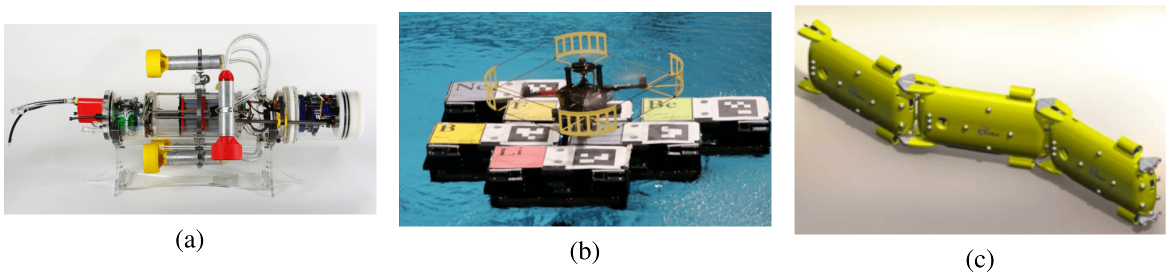


Fig. 2.4 Examples of self-propelled aquatic modular systems. (a) AMOUR system, with four thrusters. (b) TEMP robot with a group of modules acting as a landing platform. (c) Rendering of ANGELS modules in a chain configuration. Images reprinted from [79], [80] and [81], respectively.

and an anode. The cathode is placed under the module, fully submersed in water (salted to become conductive). The anode is placed on the top of the module, making contact with an aluminium ceiling. Power is provided to the modules by applying a potential difference between the ceiling and the water. A new version of the modules(Figure 2.3c) is presented in [78], with a new mechanism to connect the modules: a peltier based freezing connector.

The Lily modules [76] (Figure 2.3d) are passive and float on the surface of water. The environment is agitated, by fluid pumps, generating semi-random movement from the modules. Each module is cubic in shape, and can connect to other modules using electro-permanent magnets (EMPs) placed on their faces. This allows them to form 2D structures. Inductive communication between connected modules is facilitated by the EMPs. Each module contains a light sensor and communicates with a centralised computer.

2.1.3.2 Self-propelled systems

The AMOUR (Autonomous Modular Optical Underwater Robot) system was introduced by Vasilescu *et al.* [82] . The system was initially designed as a modular robot, and is composed of heterogeneous modules of cylindrical shape. Each type of module was responsible for different tasks, such as propulsion, power source, buoyancy, sensing and computation. A single robot could contain one or more of each module type. The heterogeneity of the modules presented both advantages and disadvantages. On one hand, each robot could be built for the specific task at hand. This greatly simplified the system while making it more specialized for the task. On the other hand, the system became limited in the ability to repair itself in operation, as a broken module had to be replaced by the exact type of module. In a situation where the number of modules of a specific type is limited, this could present a problem. In later works the system was changed in into a single cylindrical hull, with reconfigurable thrusters [79](Figure 2.4a). The system is capable of both horizontal and vertical motion as well as hoovering underwater.

The Tactically Expendable Modular Platform (TEMP) system [80, 83], consists of a group of autonomous boats, based in the concept of Modular Sea Bases (Figure 2.4b), that can self-assemble on water surface. Each boat is able to move on its own by using thrusters placed on each corner, and a group of boats can be assembled to form floating structures. These structures are then able to work as temporary bridges or landing points. In order to avoid significant tipping or capsizing, each module makes use of a bulb keel. The connections between modules are ensured by a hook and loop connection system. This system presents a certain lack of flexibility in the connections and shapes it can assemble into. Its ability to form complex structures becomes limited in environments with restricted manoeuvring space, mainly because a staggered pattern was needed when the connections are made. The reconfiguration of the structures and the modules movements are planned and controlled via a central computer.

The Roboat [84] is a modular system comprised of large boats (4 m$\times$ 2 m) that function as modules. Due to their large size, the prototypes are built at an approximate scale of 1:4. As with TEMP, this system is designed for self-reconfiguration on the water surface. The movement of the boats is performed by four thrusters, placed in a cross-shaped configuration, able to generate efficient movement. Each boat uses a multitude of sensors, such as GPS, IMU and a 3D laser scanner. The system was tested both indoors and outdoors, with docking experiments between modules, in environments with turbulent flow, being reported in [85].

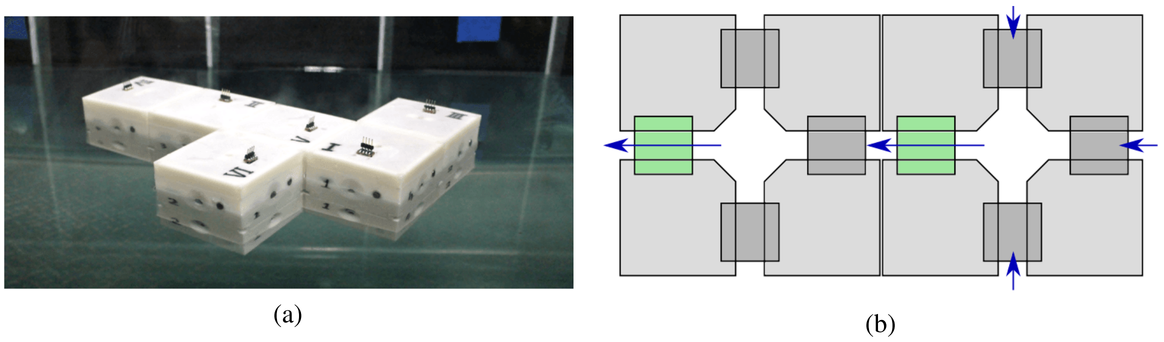
The ANGELS system (ANGuilliform robot with ELectric Sense) [81] is a modular system inspired by the sensing capabilities of fish (Figure 2.4c). Each module of the robot is able move on its own through the use of propellers. These propellers allow for the control of 3 DoF: forward and backward translation and pitch and yaw angles. Connected modules move using the relative motion of the modules, like an eel. The system was designed in such a way that the assembled robot is able to travel long distances and each module is responsible for more accurate movements. The modules were designed to be neutrally buoyant and to be able to reach a depth of 3 m, without losing water tightness. The depth control is achieved through a dedicated buoyancy control system, that by displacing the lower part of the shell of the module, alters its volume. An electric field is created by electrodes placed on the chassis of each module. The measurements of the disturbances in the electric field can be used for navigation.

The Hydron system [86] was designed to study distributed self-assembly. The modules are close to spherical in shape and are suspended in water. Horizontal movement is guaranteed by four nozzles on the module's sides, that push water drawn from the environment by an impeller at the bottom of the module. To perform vertical movement, a syringe draws or expels water to control the module's buoyancy. The modules are equipped with optical sensors and transmitters that are used for communication between modules and the alignment and docking process. While no explicit connection mechanism is used, the modules are considered docked to each other if the optical sensors are closer than a threshold value. As individual modules are unable to rotate, situations can arise where it is not possible to correctly align the modules for docking.

The Neubots system [87] consists of modules in the shape of faceted spheres, able to perform six connections with other modules. Two connected modules form a rotational joint and groups of modules are able to create 3D structures. While each module is unable to move on its own, a group of connected modules can move by generating bio-inspired swimming motions. The control strategies for the system, as well as the robot's morphology, are evolved to ensure the resulting robot is capable of efficient movement.

The REMORA project [88] explores the use of modular Underwater Autonomous Vehicles (UAVs) for inspection and maintenance of offshore structures. In concept, this is a system with multiple UAV types, with different capabilities. Each UAV is able to move independently. Reconfiguration is possible in 3D, in principle, and is used to solve task that require multiple UAV types. Furno *et al.* [89] designed and built a prototype version of the system. Additionally, a control strategy for reconfiguration that minimises energy consumed in the reconfiguration process was proposed.

The Modular Hydraulic Propulsion (MHP) system [26] (Figure 2.5a) is comprised of cubic shaped modules that move on the water surface. Each module uses four small pumps, one in each face for movement. The modules contain an internal fluid reservoir and move by exchanging water between the environment (or connected module) and their internal reservoir, through the use of the pumps (Figure 2.5b). The modules are connected using permanent magnets (two per face) and are incapable of self-reconfiguration. The system has limited sensing capabilities: magnetic and light sensing. A physical implementation of the system was built, using only binary sensors and actuators, and tested. The work presented on this thesis is focused on the MHP system; further details regarding this system are given in Chapter 3.



(a) (b)

Fig. 2.5 Modular Hydraulic Propulsion (MHP) modules. (a) Six modules in arbitrary configuration on the water surface. (b) Fluid path when two modules are connected. Active pumps are shown in green, inactive pumps are shown. Arrows show the path of the fluid. Images reprinted from [26]

2.2 Control strategies for movement of modular systems

To control modular systems, the behaviour of each module, whether physically connected in a single structure or as individual units, need to be coordinated. One of the most researched control problems in the field of modular robotics is self-reconfiguration, and how the modules should move during this process [90–99]. However, there research has been performed in the control of the movement of modular robots as a unit. Some of the strategies developed control the robot's movement by reconfiguration of the system [58] or by generating crawling, inchworm or rolling-track motion [37]. In this section, we review control strategies developed to control the movement of modular systems as well as control strategies for multi-robot system transportation, as they can be adapted for modular robots. We restrict multi-robot control strategies to those that do not make use of changes in the relative position of the modules, either by reconfiguration or joint actuation, to generate movement.

The design of the control strategies can be strongly linked to the capabilities of each unit. Control strategies for modular robotics can be broadly divided in two categories: centralised or decentralised. In centralised control, each module receives instructions from a single control unit, which can be a specific module or external to the robot. Centralised control strategies can be advantageous in implementing algorithms that require global knowledge of the sensing and actuation states of the system, as well as the robot configuration. Traditional centralised control had the disadvantage of there being a single point of failure, that is failure from the control unit/module leads to the failure of the whole system. More robust implementations of centralised control can be performed, that allow control unit/modules to fail, by changing the unit/module responsible by controlling the system. However, this requires that every module has the resources required to control the system (computational or otherwise). Additionally reliable communication between the control unit and the modules is required.

On the other hand, decentralised control strategies allow each module to control its own actions. This requires every module to be able to determine its own behaviour, based on its sensing state. As a result, every module is required to have the computational capabilities required to run the controller. Additionally, communication between neighbouring modules might be required, in order to enable collective behaviours. In general, decentralised control strategies do not require knowledge of the global state and configuration of the robot. This can be advantageous, as the the performance of the robot is not significantly impacted by the failure of one module. On the other hand, producing complex behaviours when using decentralised control strategies can be more challenging, as to the lack of global information might limit the behaviours the modules are able to produce.

2.2.1 Control of modular systems

Doniec *et al.* [27] proposed a centralised motion control for the AMOUR system. The proposed controller supports an arbitrary number and position of thrusters, although it requires the robot to have knowledge of these variables and to be able to determine its own pose (position and orientation). The controller allows for simultaneous control of rotation and translation movements in 3D. It uses the current states of the robot as inputs, and outputs speed commands for each of the thrusters. The errors between current and desired position and orientation are computed and used as inputs for two separate proportional-integral-derivative (PID) controllers, for each thruster. The two controllers compute the thrust required for rotation and translation of the robot separately. The use of the two PID controllers, allows for a good approximation of the robot's dynamics, which are not otherwise considered in the controller.

In later works [100], the requirement of knowledge of the number and position of the thrusters is relaxed. The controller is extended to allow the robot to estimate its current thruster configuration. This is performed by evaluating the impact that activating each thruster has on the pose of the robot. From that information, the controller is able to approximate the configuration of thrusters. From experimental results, this controller was shown to be able to estimate the robot's thruster configuration within 40 s, with performance similar to the results obtained when the thruster configuration was manually tuned.

When the Roboat system was proposed, the control of the position of each boat was performed using nonlinear model predictive control (NMPC) [84]. In later works [101], structures comprised of multiple boats use a feedback control system to control the pose of the robot. The robot is given a reference trajectory that it is required to follow. A coordinator module is elected by the controller to determine both the next position of the robot and the outputs for each thruster. The feedback loop is divided in two components: coordination and robust control. The robust control component is responsible for determining the total force and torque required to be applied on the robots center of mass (CM), so that the reference trajectory is followed. This is performed by a proportional-derivative controller (PD). The coordination component of the controller is responsible for computing the actuation state of each propeller to obtain the desired force and torque acting on the CM.

The controller proposed for the ModQuad system is able to control 4 DoF: three translations, and yaw rotation [64]. The controller requires each module to know the geometry of the robot and its own position in the structure, and that all modules share a common reference frame. In order minimise the force of each rotor has to exert, the total thrust force and robot moments is evenly distributed along all thrusters. The total force acting on the robot, as well as desired attitude, are determined by a central trajectory planner, that sends this information to the robot. As every module has knowledge of the robot's configuration, as well as, its position on it, they compute their own control inputs independently, based on their position. The modules are able to determine their attitude, which is compared to the desired attitude. Each module then computes the difference between its current and desired orientation and feeds it to a PD controller, which gives the control inputs for each rotor.

The controller developed for the DFA is able to control the system's 6 DoF [63]. The modules are able to sense their current altitude and attitude, as well as, angular velocities. The horizontal position of the robot, as well as, linear velocities, are determined by external sensors that send the information to each module. The system uses a decentralised feedback control strategy, adapted for vertical take-off and landing (VTOL) vehicles. This allows for stable hovering of the assembled robot. The control strategy used, allows for each DoF to be decoupled. The close-loop response of each DoF is then forced to behave as a second-order mass-spring-damper. The parameters for each DoF are optimised externally due to the limited computational capabilities of each module. Previous to flight, the optimised parameters are stored in each module. While a single set of parameters can be used for multiple robot configurations, the optimal performance for each configuration requires a different set of parameters. However, for configurations with a large number of modules, the differences in performance are negligible.

2.2.2 Control of multi-robot systems

In addition to strategies developed specifically for modular systems, control strategies developed for multi-robot systems can be relevant, and applicable, to modular robots. In particular, strategies that require a group of robots to move or manipulate objects in a cooperative manner can be applicable to controlling the movements of modular robots. In these tasks, the transported object can be seen as the modular robot, and the robots transporting the object as the actuators. In this analogy, the sensing information each robot obtains from the environment corresponds to the sensing information collected by each module.

In general, transportation of objects by a multi-robot system can be divided in two phases: the approximation phase, where the group of robots searches and approaches the object, and the transportation phase, where the robots cooperatively move the object. When adapting control strategies for multi-robot systems to modular systems, the focus is the transportation phase, as the actuator positions are defined by the robot's structure. However, the approach phase can give insight on rules for actuator activation, particularly if the positions of the group of robots are predefined.

Nicholson *et al.* [102] propose a decentralised control strategy to manipulate the orientation of large objects. The movements of the objects are performed by a swarm of robots moving on the water surface. Each robot has information about the object's orientation, the desired orientation, their own position in relation to the CM of the object and their orientation in relation to the object. However, they have no information about the number and position of the remaining robots. Each robot computes the force it needs to exert on the object boundary in order to rotate it towards the desired orientation. This controller uses a feedback loop, based on the dynamics of the system, position of the robot, in relation to the CM of the object, and error between current and desired orientation.

Braganza *et al.* [103] present a controller solution for a similar problem. A group of tugboats is required to manipulate a disabled vessel into a predefined position and orientation. To control the vessel's 3 DoF, a minimum of six robots is required, positioned in aligned symmetrical positions. Each robot is unable to communicate with other robots, and only has information about the vessel's current position and orientation, and its own position and orientation in relation to the vessel. The controller uses of a feedback loop to allow each robot to compute its required thrust, based on a desired trajectory, and is shown to perform well when the position of the robots along the vessel is not know to each robot. The controller assumes that the positions and orientations of the tugboats with respect to the vessel are always constant.

Esposito *et al.* [104] expand the previous controller [103], to more flexible distributions of the robots along the vessel's boundaries and remove the requirement of the number of robots in the swarm. The net thrust required to move the vessel along the desired trajectory is determined and used to compute each robot's thrust value, through an optimisation process. The optimisation is performed in order to minimise the difference between the summed thrust of all robots and the net thrust for the desired trajectory. However, this requires the number of tugboats in the swarm to be known. In [105] the work is further extended to allow optimal placement of the tugboats around the object.

Sartoretti *et al.* [106] propose another control strategy for the transport of an object by a group of robots. The object needs to reach to a goal point, with a predetermined orientation. The proposed controller is decentralised, and based on the concept of conditional caging [107], where the robots distribute themselves around the object, in order to manipulate it. The control strategy is divided in two separate phases: approach and grasp, and manipulation. In the first stage, the robots approach and surround the object to be moved. Each robot is given a specific point of the object boundary as a goal, that they are required to reach simultaneously. In the manipulation stage, the robots cooperatively move the object. The controller is developed for robots possessing two actuators (left and right), that can be controlled independently, to generate 2D movement. The output of each actuator is determined based on the robots position and orientation in relation to the target point and desired orientation. The control laws for each actuator generate spiral trajectories. It is shown that by using a maximum of two consecutive spiral arcs for each stage, the robots are always able to correctly grasp the object, and transport it to its destination point, with the predefined orientation. The parameters for each arc are determined by the robots at the beginning of each stage. In the manipulation stage, the object's orientation and position are constantly changing.

Hu *et al.* [108] propose a control strategy for moving an elongated box underwater. A group of autonomous fish robots, capable of sharing sensory information, is used to solve the task. The fish move through the use of three fins, with one rotational degree of freedom each. Each robot is able to detect the object and the target location through the use of a camera mounted on the front of the robot. The robots identify the object and target location through colour markings placed on them. While the target location has a single colour, the object uses two different color markings positioned on the center of one face and on the ends of the opposite face. Simple image processing algorithms are used to determine the position and orientation of the object and target location, with respect to the robot. This control strategy requires a minimum of three robotic fish: one observer and two pushers (left and right, one for each side). The sub-task of each robot is dynamically allocated. Once the allocation is complete, the robots move in circular trajectories to locate the object or the goal, depending on their sub-task. As each robot's target is located, the robot's movement changes to approach the object/target location. Upon reaching the target, the observer robot searches for the object by performing circular trajectories with small radius. From a position close to the target, in which the object is detectable, the observer determines the object's position and orientation, sharing this information with the pusher robots. The pusher robots, upon reaching the target and with the information from the observer, determine the velocity they need to reach to push the object into the target location.

Chen *et al.* [109] proposed a cooperative transportation controller for miniature mobile robots. The task that the robots have to solve involves transporting an object towards a goal, irrespective of orientation. Each robot is able to detect and differentiate the goal and the object, using its forward facing camera. The camera image is processed to to determine if the robot detects the object and/or the goal, and their position(s) relative to the robot's current heading [110]. The proposed controller uses occlusion as a key element in determining the behaviour of the robots. Each robot starts by moving randomly, while searching for the object to be transported, moving towards it once it is found. Once the object has been reached, the robot searches for the goal, by rotating on the spot. If the goal is not detectable, it is assumed to be occluded by the object. In this scenario, the robot pushes the object. While in contact with the object or the goal is not detectable, the robot keeps pushing the object. The process starts from the beginning once the goal is detectable or the robot loses contact with the object. The controller can be implemented as a finite state machine, where the processing of the camera image is the most computationally expensive process. It is shown that, when using this strategy for objects of convex shape, the distance between the object's CM and the goal monotonically decreases. This controller requires no communication between the robots.

An adapted version of the previous controller [109] was used by Doyle *et al.* [26], to control the translation movement of an MHP robot. Using the same occlusion principle, an actuator is active if its respective sensor does not detect the goal. This emulates the actions of multiple robots pushing an object, if the goal is not perceived.

2.3 Summary

In this chapter, a review of relevant literature on modular robotic systems was performed. Systems that move in fluid environments are of particular interest as, in general, they possess a larger number of controllable DoF. Additionally, a review of relevant control strategies that can be applied to control the movement of modular robots was performed. This includes some control strategies for multi-robot systems.

Land-based systems correspond to the larger fraction of modular systems developed to date. However, even though they present a large theoretical potential, the evidence of their successful use in field applications is limited. Additionally, land-based systems, while potentially able to reconfigure in 3D, are generally limited to 2D movement. Air based systems on the other hand, can perform movements in 3D, but are usually limited to reconfigurations in 2D. Liquid environments present the potential of 3D movement coupled with a high reconfiguration space.

There is a growing interest in modular systems that can be deployed in liquid environments. However, fewer systems have been developed for this environment. Additionally, a significant number of systems developed for liquid environments are incapable of self-propulsion, as individual units or as a connected structure. This means that their movement needs to be assured by external elements.

The majority of control strategies developed for modular systems deal with the self-reconfiguration problem, while only a small amount of control strategies deal with the movement of the robot as a whole. However, some control strategies used for multi-robot systems that solve object transportation tasks can be adapted for the control of the movement of modular robots. This can be performed by creating a parallel between the modular robot and the object, where the multi-robot system acts as the modular robot actuators.

Centralised control strategies [27], or strategies that assign sub-tasks to a single module [108], present a single point of failure, which can compromise the completion of the task. Systems and control strategies that require the use of external sensing [63, 84] are not easily scalable. Additionally, controllers that require either complex sensing and communication [104, 106], or high computational power, might not be feasible in systems that have limited capabilities. Similarly, previously proposed decentralised controllers that make use of the robot's geometry and/or the module's position within the robot's geometry [64], require either some level of computation or that information to be manually introduced in the controller. On the other hand, while there are some controllers capable of generating movement in systems with limited capabilities [26, 109], they are unable to control the orientation of the robot. The aim if this thesis is to fill this gap, through the investigation of control strategies that are able to efficiently move the robot using minimal computation, without requiring external or complex sensing.

Chapter 3

MHP 2: Hardware Revision and Communication Study

This chapter is based and extends on the author's original contributions to the publication [111].

3.1 Introduction

As we have described in Chapter 2, one of the advantages of modular robotic systems comes from their ability to be fault tolerant and that they can easily replace malfunctioning units [8, 16]. However, it is still expected that each unit functions in a reliable manner, without human intervention, to guarantee successful completion of tasks.

Doyle et. al [111] introduced the concept of Modular Fluidic Propulsion (MFP). An MFP robot consists of a set of connected modules that work in fluid environments. Each module is a square (or a cube in 3D), with a set of magnets on each face, used to form connections with other modules. Each module contains an internal fluid reservoir that forms a fluid network when connected to other modules. The modules propel themselves by moving fluid between the environment and their internal reservoir. There are two possible implementations of the concept [111]: the Modular Hydraulic Propulsion (MHP) and the Modular Pneumatic Propulsion (MPP). In this thesis, we are interested in the MHP concept.

The MHP is a hardware implementation of the MFP concept for liquid environments, and was first introduced in [26]. Each module is actuated by pumps, where each pump is connected to a port positioned at the center of each module's face. When active, a pump extracts water from the reservoir and discharges it through the corresponding port into the environment (or a neighbouring module). Similarly, fluid is drawn into the reservoir from the environment (or a neighbouring module) via ports connected to inactive pumps. This fluid routing process provides the motive force for the module.

In [26], an initial hardware prototype of the MHP concept was build. However, the prototype presents some limitations. In this chapter, we design, build and test a new implementation of the MHP concept—the MHP 2. In this implementation, the modules are designed for movement in the 2D space. We use the term *faces* to refer to the 4 vertical faces of the module only. Additionally, we use the robot to complete a translation task, where the robot needs to move towards a goal.

The remainder of this chapter starts by listing the requirements of the MHP 2 system, in Section 3.2. Section 3.3 details the limitations of the initial hardware prototype that need to be overcome in the new design. Section 3.4 describes the design and build process of the electrical and mechanical components of the MHP 2. Section 3.5 presents an evaluation of the performance of the MHP 2, performing the translation task, using two different controllers. Section 3.6 concludes the chapter.

3.2 Requirements for the MHP 2

In [26], an MHP prototype was shown to be able to move towards a target. However, it does not allow for control of the orientation of the robot. One of the aims of this work is to develop effective pose control strategies, which can require coordination between the modules, and additional information. Additionally, in [111], variants of the occlusion controller were proposed, which require additional capabilities from the modules. As a result, the purpose of building the MHP 2 is not only to overcome any issues of the initial hardware prototype had, but also increase the module's capabilities. The desired increased capabilities consist of the ability to:

- Communicate with connected modules;
- Detect obstacles within a certain range;
- Sense the module's orientation.

Joining the capabilities listed above with the ones already present in the initial hardware prototype, we are able to create a list of requirements for the design of the MHP 2 modules:

1. Reliable module connection detection: connections only detected if the modules are separated less than 0.5 cm (i.e. sufficiently close to attract each other);
2. Goal detection: modules capable of detecting changes in light conditions from light sources up to 120 cm away;
3. Obstacle detection: modules capable of detecting obstacles from 10 cm away;
4. Communication between connected modules: modules capable of exchanging 1 bit of information every 10 ms;
5. Sensors and actuators accessible in 90 s of disassembly;
6. Rechargeable batteries;
7. External port to allow charging the batteries without requiring the disassemble the module;
8. Ease of replacement of malfunctioning sensors and actuators;
9. Ease of access to the modules batteries;
10. Simultaneous operation start for all modules;

In addition to the aforementioned requirements, and to keep in line with the minimalistic concept, we aim to reduce the size of the modules. However, this decrease in size needs to be balanced with the robot being able to float on water.

3.3 Limitations of the original MHP modules

To ensure the MHP 2 represents a significant improvement over the original version a detailed analysis of the initial hardware prototype is required. This will help to identify its main limitations and areas of improvement. Figure 3.1 shows the initial hardware prototype presented in [26].

The unreliability of the modules is the problem that has the most noticeable impact on the robot's performance. One of the most relevant factors in decreasing the module's reliability is

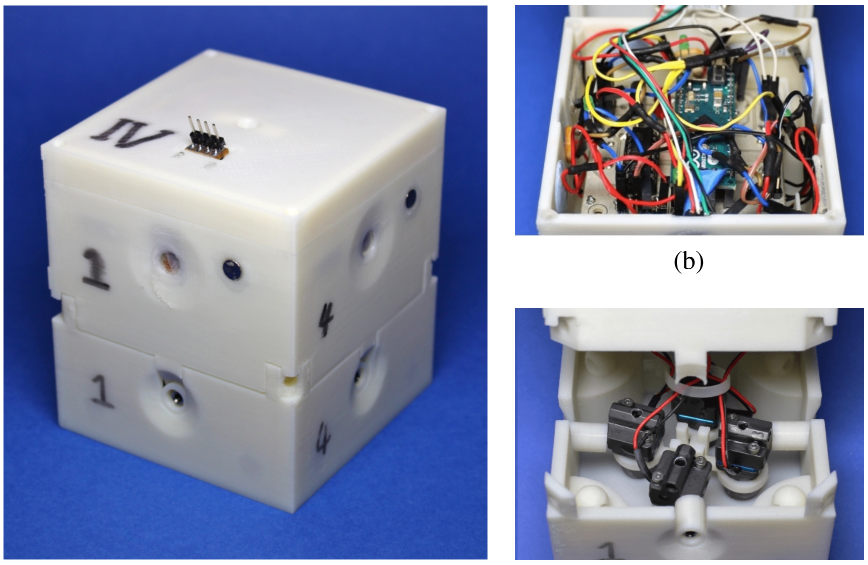


Fig. 3.1 Initial hardware prototype of an MHP module. (a) Assembled view. (b) Module's internal circuitry. (c) Module's internal reservoir and pumps. Reprinted from [26].

the electrical connections. The use of cables, as seen in Figure 3.1b, increases the likelihood of an incorrect or unreliable connection between different electrical components—increased possibility of human error. Loose connections in the modules lead to a need for frequent maintenance. The presence of the cables also has the negative effect of increasing the complexity of the maintenance of the modules.

The connection sensor is a second contributor to the unreliability of the initial hardware prototype. As part of an autonomous modular robot, each module is expected to reliably detect connections with other modules. The initial prototype used magnetic switches as connection sensors. These were unreliable, and led to the need for hard-coding the connections in the control strategy when the robot configuration changes.

Another limitation of the system is the permanent placement of the sensors and actuators. To ensure that the sensors are kept in place and no water gets in contact with the electronics during the robot's operation, they are permanently attached to the chassis of the robot. As a consequence, it is not possible to replace malfunctioning sensors or broken actuators. This limits the modules long term usability, particularly regarding the actuators, as they have a higher fail rate.

The utilisation of a non-rechargeable battery provides an additional cause for the module's maintenance. Ideally, the modules should only be disassembled when replacing defective components or performing other type of maintenance. As the battery is placed under the electrical circuit, every time it needs to be replaced a partial disassembly of the module is required. This, as a consequence, has the increased possibility of electrical component failure.

The last, although minor, limitation we identified in the system is the incapability of having all the modules start operating at the same time. The system uses a magnetic switch to allow the batteries to power the modules. As a result, a magnet needs to be placed in each module for it to activate and start operation, leading to different starting times. While the differences in starting time do not seem to have a significant impact on the usability of the modules, correcting this issue can improve the overall performance of the robot.

3.4 Design of the MHP 2

The design of the MHP 2 involves the design of a new version of the electronics and chassis of the modules. This section details the design and build process of the MHP 2.

3.4.1 Redesigning the circuit

In order to fulfil the given list of requirements for the circuitry of the system, appropriate components need to be chosen. For the detection of connections between modules we use a Hall-effect sensor. This sensor is capable of detecting changes in magnetic field in its proximity. A magnet–sensor pair is placed in each of the module's faces to provide the ability to detect neighbouring modules. The chosen sensor has a range of 1 cm. This is an appropriate range as we only want to detect established connections.

As we use a light source as a goal, we use a phototrasinstor as a goal sensor. A total of four sensors are used per module, one on each of the module's faces. Each sensor has a detection beam of 120 °. In order to maximise the range of the sensor, we use an operational amplifier. The amplification was determined so that the sensor gets saturated if the goal is closer than 3 cm and to be able to reliably detect changes in the environment lighting up to 90 cm.

As we have limited space, we use a single sensor for both the communication and obstacle sensing. We use four infra-red (IR) transceivers per module, one per face. The functionality of the transceiver—communication or obstacle detection—is determined by the presence of a neighbouring module for each face. If a module's face is connected to another module, the transceiver can be used for communication, allowing one bit of information to be shared between connected modules. If the face is free of connections, the transceiver can be used as an IR range sensor to detect obstacles. Each transceiver has a detection beam of 45 $^{\circ}$. In order to obtain an adequate obstacle detection range we use an operational amplifier. The amplification was determined so that the sensor can detect obstacles in a range of 5 cm to 10 cm.

To determine the orientation of each module we use an electronic compass. The compass returns the angular offset of the module, relative to the orientation of the module when powered on.

We power each module with two rechargeable 3.7 V lithium polymer batteries. These batteries can be charged via a set of battery charging pins, when the robot is turned off. Each module has a mechanical switch to change the connections of the batteries between the charging pins and the circuit. In addition, we use an infra-red receiver to receive external information. This allows the modules to receive information from an external source such as a conventional television remote control.

The combination of the infra-red receiver and the power switch allows for three operating modes: the `Off/Charging` mode, the `Standby` mode and the `Active` mode. In the `Off/Charging` mode the batteries are disconnected from the rest of the circuit. In the `Standby` mode the batteries are connected to the circuit allowing the sensors and LEDs to be active, but the actuators do not respond to sensor readings. In the `Active` mode the actuators respond to the sensor readings in accordance to the controller implemented.

Each module is actuated by four micropumps (TCS micropumps, model M200-Sub). Each pump has dimensions of 29 mm $\times$ 16 mm $\times$ 16 mm, and is capable of a maximum flow rate of 11 ml/s. To drive each pump we use an H-bridge motor driver.

Each module is controlled by an ATMega 324P processor. The processor was chosen by taking into account the number of analog-to-digital converter (ADC), digital, pulse-width modulation (PWM) and serial ports required for all the components above described. Additionally, we use multiple light emitting diodes (LEDs) to signal active pumps, non-connected faces and for debugging purposes.

In order to ensure the reliability of the system we placed all components into printed circuit boards (PCBs). This not only eliminates the need for easily detached cabling, and all

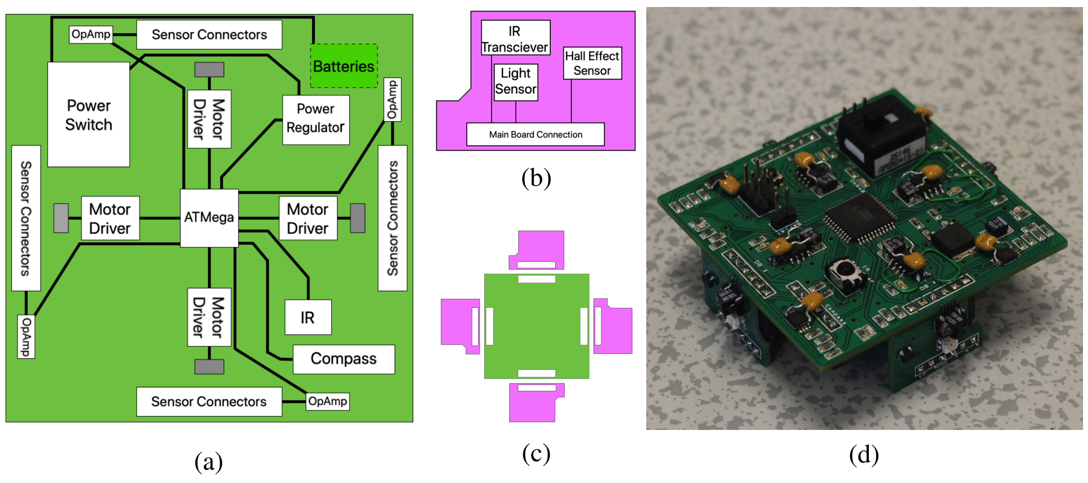
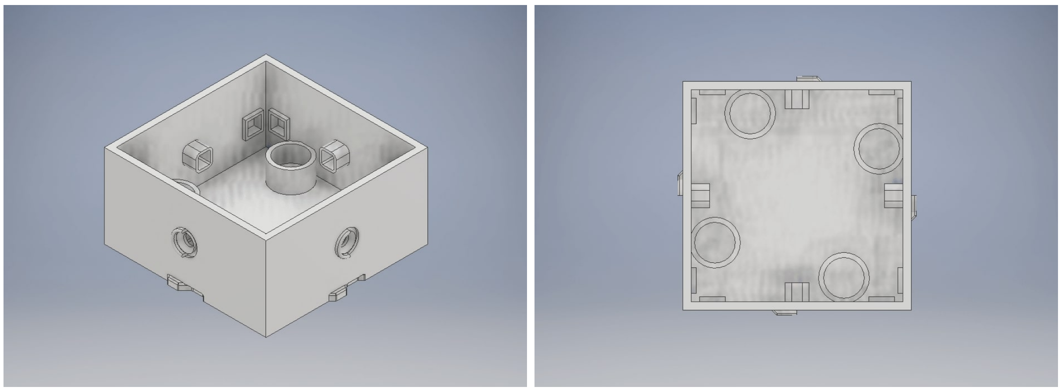


Fig. 3.2 Final circuit design. (a) Simplified schematic of the main PCB, containing all of the components with the exception of face sensors. (b) Simplified schematic of the daughter PCB, containing the sensors for each face. (c) Relative positions of all the boards for a single MHP module. (d) Produced PCBs; one main board and four daughter boards.

the aforementioned problems associated with it (unreliable, incorrect or loose connections), but also ensures a more reliable connection between the components of the circuit.

Each module uses a total of five PCBs: one main board and four daughter boards. The main board contains all of the electrical components with the exception of the sensors for each face. The four daughter boards are identical and contain the sensors used in each face—phototransistor, IR transceiver and Hall-effect sensor. By separating the sensors and the rest of the components into different boards, easy replacement in case of sensor failure is obtained. The four daughter boards connect perpendicularly with the main board and are placed at the centre of each face. Figure 3.2 shows a simplified version of the PCBs (Figures 3.2a and 3.2b), their relative position (Figure 3.2c) and their implementation (Figure 3.2d).

The final size of the PCBs was determined concurrently with the design of the module's body. The final shape and size of the main board is a square with 58 mm side length. The final daughter PCBs shape is close to a rectangle with a width of 21.75 mm and a height of 15.25 mm. We made the design of the two PCBs using a software developed by RS Components—Design Spark. Different versions of the circuit were implemented and tested to obtain the final version used on the modules.



(a) (b)

Fig. 3.3 Computer Aided Design (CAD) representation of the *lower hull* of the chassis. (a) Isometric view. (b) Top view.

3.4.2 Redesigning the chassis

Placing all the components in PCBs not only increases the reliability of the system but also reduces the footprint of the electronics. As a consequence, the module's size is more easily decreased. The body of the modules (chassis) was redesigned using Autodesk Inventor Professional 2016 and obtained through 3D printing.

The module's chassis redesign involves the redesign of two different components: the *lower hull* and the *upper hull*. The *lower hull* houses the actuators and the internal water reservoir. The *upper hull* houses the electronics and power of the module. In addition, we need to ensure that no liquid moves from the *lower hull* into the *upper hull*, under the risk of damaging the electronic components of the module.

As we need to place four pumps in each module, the size of the actuators becomes the biggest limiting factor in the module's size reduction. At the time of development, to the best of our knowledge, the micropumps we use were the smallest, fully submersible pumps commercially available.

In both implementations of the MHP, the *lower hull* has a square cross-section and has four ports, one per face, that connect the internal reservoir and the environment. In the original configuration, the pumps were placed in the center of the water reservoir, pointing towards the faces (Figure 3.1c) and directly connected to the ports of each face. In order to save space, each pump is placed on an internal wall of the *lower hull*. Each pump outlet is then connected via a pipe to the port in its respective face. Eight cut-outs (two per face) were made to house the magnets that connect the modules. By rearranging the placement of the

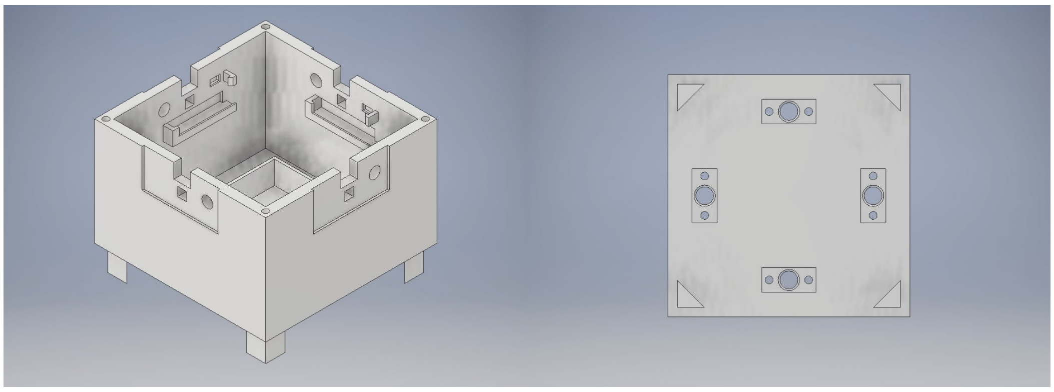


Fig. 3.4 CAD representation of the *upper hull* of the chassis. (a) Isometric view. (b) Bottom view.

pumps, we are able to reduce the module's side length close to 20%—from 80 mm to 65 mm. Taking into account requirements from other components, the final side length of the module in 67.5 mm. This corresponds to a reduction in side length of approximately 15%. The final height of the *lower hull* is 35 mm, with a 4 mm bottom wall. A complementary tab and notch per face help to align connected modules. Figure 3.3 shows two different perspectives of the design of the *lower hull*.

The design of the *upper hull* required careful calculation of where all the sensors will be placed. Three openings per face were made for the phototransistor, IR sensor and magnet. A cut-out was made in the internal side of each face for the Hall-effect sensor. On the internal walls of each face, a support for the daughter boards was created; the daughter boards latch into these supports. The main board is then supported by the four daughter boards. Four rectangular cut-out were made in the center of each face. These cut-outs are covered with a layer of transparent polyethylene. This allows the sensors to retrieve information, while preventing water from entering. Four openings were made in the bottom of the section to allow cables form the pumps to go from the internal reservoir into the *upper hull*. Complementary pieces (hereby designated pump links) to these openings are glued to the cables of the pumps. By using these links, we are able to replace malfunctioning pumps when necessary, without having to replace larger components of the chassis. Four triangular legs make the connection with the *lower hull*. The two hulls and the pump links are hold in place by friction. Figure 3.4 shows two different perspectives of the design of the *lower hull*.

To protect the top of the electronics from any water that might overflow during operation, we designed a lid to the module. It consists of a thin layer of plastic, with openings for the

Table 3.1 Waterline level for different module heights.

Robot height	**Submerged height**	**Distance between water and top of the robot**
65 mm	61.4 mm	3.6 mm
75 mm	58.1 mm	16.9 mm
80 mm	57.1 mm	22.9 mm

mechanical switch, programming pins and IR receiver. The pump and connection LEDs are visible through the closed module lid, allowing the status of the module to be ascertained during operation.

To determine the final height of the module we need to take into account requirement 13, that is, that the module should float on the water surface. To do this we need to consider the Archimedes' principle:

$$\frac{\rho_{object}}{\rho_{fluid}} = \frac{m_{object}}{m_{displaced fluid}}, \tag{3.1}$$

where ρ and m represent the density and mass, respectively, of the object and fluid.

We tested different dimensions for the height. Table 3.1 shows the level of the water for the three heights tested. To avoid the water interfering with sensor readings, we need the water line to be below the sensors. The water line was below the sensors only for module heights of 75 mm and 80 mm. To avoid the risk of water getting into contact with the electronics, the final module height was chosen to be 80 mm.

All of the parts of the chassis were 3D printed, using a Stratasys Mojo 3D printer, with ABS (Acrylonitrile butadiene styrene) plastic. While ABS is waterproof, initial designs indicated that, if the modules were on water for extended periods of time (more than two hours), water was able to get in. The leak points were identified to be the small gaps between the upper hull and the small piece used for passing pump cables (pump link), as well as the small intervals between printed layers. As a result, in order to ensure that no water gets in contact with the electronics, the upper hull receives a coating of resin, to ensure full waterproofing.

Figure 3.5 shows the final version of the new implementation of the MHP module. A total of six modules were built.

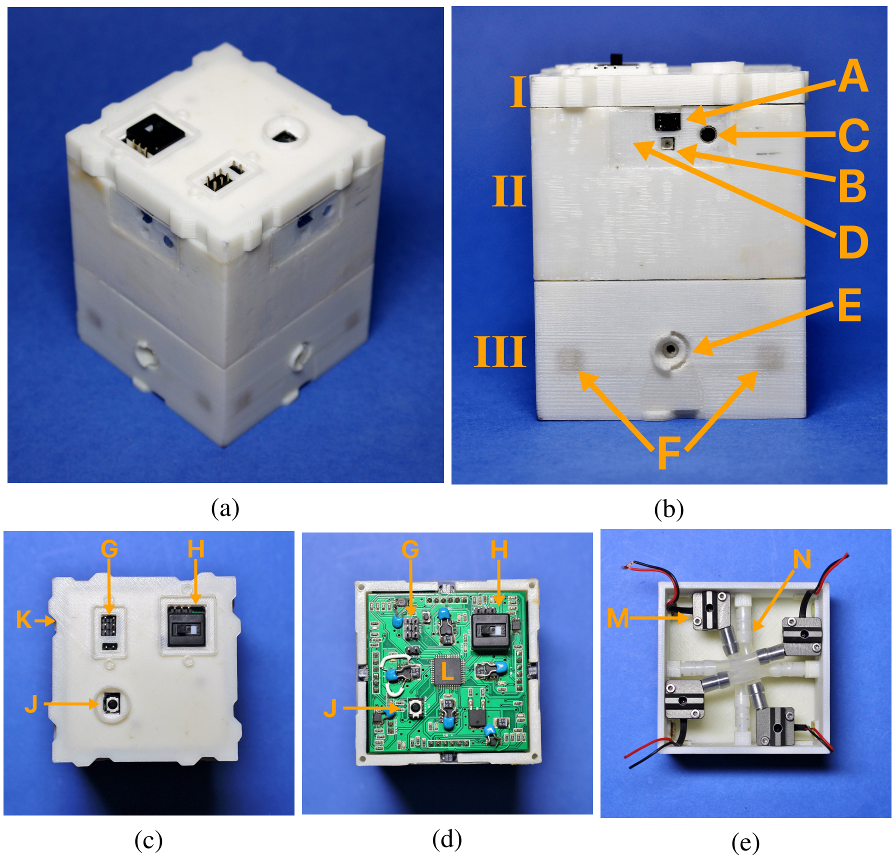


Fig. 3.5 Physical implementation of the MHP 2. (a) Assembled module. (b) Side view. (c) Top view. (d) Internal circuitry. (e) Internal reservoir and pumps. I: lid. II: *upper hull*. III: *lower hull*. A: IR transceiver (communication and range sensing). B: Phototransistor (goal detection). C: Magnet for connection detection. D: Hall-effect sensor. E: Pump input/output port. F: Attachment magnets. G: Programming pins. H: Module switch and charging pins. J: IR receiver (remote control). K: Alignment features. L: Microcontroller. M: Micropump. N: Pump-port connecting pipe.

3.4.3 Comparison between the initial hardware prototype and MHP 2

With the new dimensions and component placement, the MHP 2 robot has 63.3% of the volume and 73.2% of the mass of the initial hardware prototype. Table 3.2 summarises the differences between the two implementations.

Table 3.2 Comparison of characteristics between the two MHP module implementations.

Characteristics	Initial hardware prototype	MHP 2
Dimensions [mm]	$80 \times 80 \times 90$	$67.5 \times 67.5 \times 80$
Weight [g]	272	199
Processor	ATMega 32U4	ATMega 324p
Actuator (pumps)	TCS M200-sub	TCS M200-sub
Light detection	Photo resistor	Phototransistor (Vishay VEMT4700)
Connection detection	Magnetic switch	Hall-effect sensor (Honeywell SL353HT)
Obstacle detection	✗	IR Transceiver (Vishay TCND5000)
Communication	✗	
Simultaneous start	✗	IR Recievere (Vishay TSOP6438TR)
Rechargeable batteries	✗	✓

3.5 The MHP 2 in action: Impact of communication

While we improved the capabilities of the MHP 2 modules, they are still fairy limited. As a result simple control strategies are required to operate them. A study of the movement capabilities of an MHP robot was conducted in [26]. The robot was shown to perform well in a translational task with an occlusion-based controller. The robot achieved high success rates and showed robustness to actuation noise [26]. However, the controller used allowed some actuators to be active, with no impact on the robot's movement.

In this section, we perform physical experiments to validate the MHP 2 robot and explore the impact that the inclusion of communication in the controller can have on the performance of the robot when completing a translation task. We are particularly interested in the effects communication can have on energy consumption.

We compare the performance of the occlusion based controller used in [26] (adapted from [109]), and a variant controller, with communication between modules. We denote the occlusion based controller as `dec` and the variant presented in [111] as `dec-com`. Using the `dec` controller, an external pump is active if its respective face does not detect the goal. On the other hand, the `dec-com` controller only activates an external pump if the goal is not

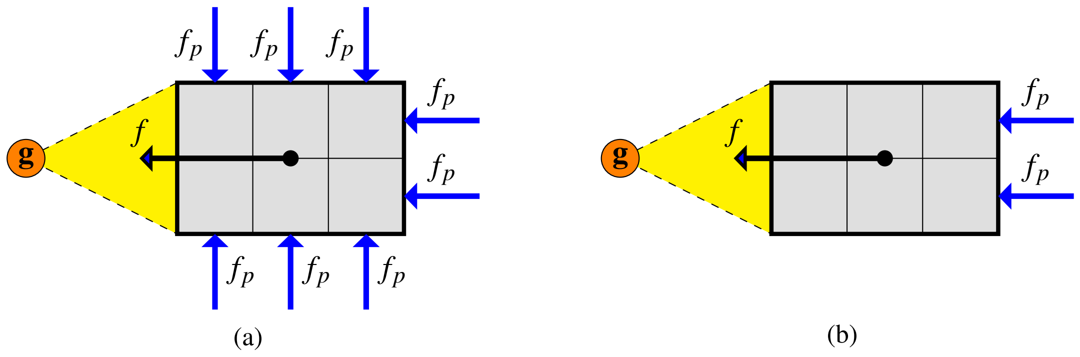


Fig. 3.6 Comparison of the forces acting on the robot for both controllers. The goal point is represented by **g**. The arrows represent the actuation force caused by each active pump (blue) and the net force acting on the robot (black). (a) Forces acting on the robot for the `dec` controller. (b) Forces acting on the robot for the `dec-com` controller

detected by its respective face, while being detected by a paired (parallel) external face. This would not be possible if the modules were not able to exchange information.

Figure 3.6 shows an example of the active pumps of a robot, using the `dec` (Figure 3.6a) and `dec-com` (Figure 3.6b) controllers, in the same situation. For both controllers, the net force acting on the robot is the same. This suggests that the velocity and trajectory of the robot should not change, in a given situation, when changing the controllers. As the forces acting on the robot are symmetric with respect to the center of mass, the robot is expected to perform a purely translational movement.

3.5.1 Experimental setup

To evaluate the impact of the presence of communication we undertake physical experiments with the MHP 2 system. We compare the performance of the occlusion based controller, both with and without communication. For all trials undertaken we use a robot with four modules, arranged in a 2×2 configuration.

Figure 3.7 shows an illustration of the experimental environment, a water tank of 115 cm length and 55 cm width. The water level is deep enough such that the robot can float freely without touching the bottom of the tank. The environment is unobstructed, containing only a single robot. In the pre-trials performed, if the robot hit the glass boundaries of the water tank, it stopped moving. We hypothesise that the friction between the robot and the glass walls were higher than the thrust force the robot is capable of generating. To counteract this

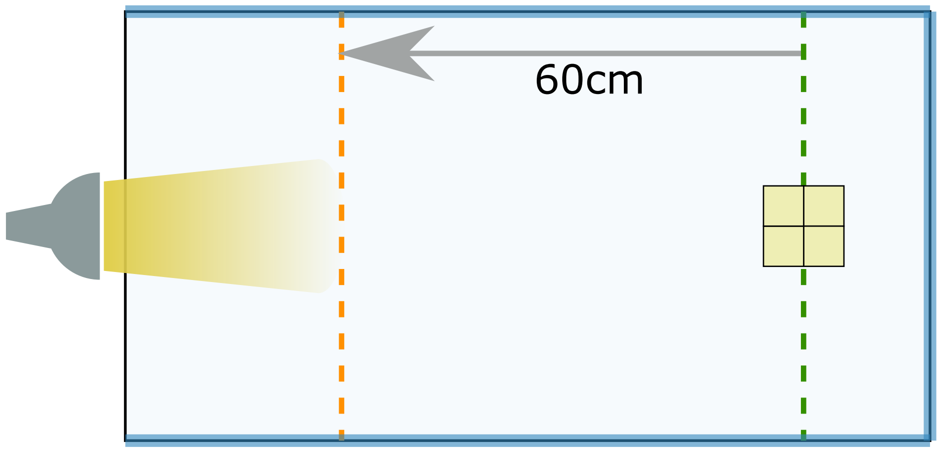


Fig. 3.7 Illustration of the experimental setup. A water tank of dimensions 115 cm and 55 cm with a 2×2 robot shown on the right. The goal is represented as a lamp on the left, outside of the tank. The dashed green and orange lines represent the start and finish lines, respectively. Thick blue lines indicate the presence of an elastic-band boundary.

effect, elastic bands are attached to three sides of the tank—two large sides and small side closer to the starting position.

At the beginning of each trial the robot starts at one end of the tank, approximately equidistant to the tank's long boundaries, as illustrated in Fig. 3.7. The robot's starting orientation is chosen randomly. The robot is tasked to move towards the goal, which is represented by a white LED lamp (806 lumens) near the other end, but outside, of the tank. All trials are recorded using an overhead camera.

Each trial lasts 120 s. A trial is considered successful if, before the time limits, the robot's centroid reaches the finish line. The finish line is placed 60 cm nearer to the goal than the line where the robot started; the start and finish lines are determined by post-analysis of the video recordings. We perform two sets of experiments, composed of 30 trials each. One set of trials is performed for each controller.

3.5.2 Controller implementation

The two controllers we use are fully decentralised and reactive [111]. Each module runs exactly the same controller. The controllers use binary sensor readings. For the goal detection, this is realised by using pre-defined threshold values, which were calibrated experimentally to suit the environmental setup. Every module face is numbered, from 1 to 4, in a counter-clockwise manner.

Table 3.3 Truth table for the dec controller. We use p to represent the pump activation state; d and c represent the light and connection sensor readings of the face, respectively. Active pumps and sensors are represented with a 1. Inactive pumps act as inlets.

c	d	p
0	0	1
0	1	0
1	0	0
1	1	0

As mentioned above, in this section, we are focussed on the differences in energy consumption between controllers. As the vast majority of the literature on modular robotics is focused on self-reconfiguration/self-assembly, no direct comparison with controllers from the literature is performed. In [111], simulation results are used to compare the performance of these controllers with a centralised controller from the literature. However, as this controller requires global positioning and the modules do not have this capability, it would require the robot to be tethered to a computer, performing the tracking of the robot, to determine the pump outputs, which could impact the movement of the robot.

3.5.2.1 Occlusion based controller (`dec` controller)

The dec controller is an adaptation of the controller proposed in [109]. The controller is designed to run on each face; the activation of the pump (p) for each face is independent of the remaining faces. The controller uses only information from the light (d) and connection (c) sensors. Table 3.3 shows the truth table of the controller for each face based on the sensor readings. The dec controller can run on a face level and does not require run-time memory. If a face is external (no connection, $c = 0$) and does not detect the goal ($d = 0$), the pump connected to the output of that face fires ($p = 1$). In any other situation, the pump is inactive ($p = 0$). As a result, occluded faces push the module, while faces detecting the target do nothing.

The controller implementation is described in Algorithm 1. Lines 6–14 implement the truth table of the controller. The sensor values of the current face are read, and used to determine whether to activate the current face's pump. Lines 15–19 ensure that a single module will not fire all pumps simultaneously. Once the status of all pumps is set, if all pumps are active, they are deactivated. If all pumps fire, there would be no water inlet, creating a very low pressure in the water reservoir. This has the potential risk of damaging the actuators. By adding this condition, we ensure that all pumps cannot fire simultaneously.

Algorithm 1 `dec` controller implementation

```
1: $c_k$, $d_k$ and $p_k$ be the states of the sensors and pump of face $k$
2: $n_{faces} = 4$
3: $t_{cycle} = 100$                                   ▷ Time in milliseconds.
4: loop
5:     $t_1 \leftarrow$ time
6:     for $k \leftarrow 1, n_{faces}$ do                ▷ Go through all faces.
7:         $c_k \leftarrow$ Connection sensor reading      ▷ Get connection sensor readings.
8:         $d_k \leftarrow$ Goal sensor reading            ▷ Get light sensor readings.
9:         if $c_k = 0$ and $d_k = 0$ then     ▷ If external face and occluded, activate pump.
10:            $p_k \leftarrow 1$
11:        else                                  ▷ Otherwise, deactivate pump.
12:            $p_k \leftarrow 0$
13:        end if
14:    end for
15:    if $p_1 = p_2 = p_3 = p_4 = 1$ then        ▷ If all pumps active, deactivate them.
16:        for $k \leftarrow 1, n_{faces}$ do
17:            $p_k \leftarrow 0$
18:        end for
19:    end if
20:    $t_2 \leftarrow$ time
21:    if $t_2 - t_1 < t_{cycle}$ then    ▷ If the loop was faster than the minimum cycle time, wait.
22:        wait($t_{cycle} - (t_2 - t_1)$)
23:    end if
24: end loop
```

Algorithm 1 is run on a continuous loop over time. Due to the limited response time of the pumps, we ensure the pump states are only updated up to ten times per second (lines 21–23).

3.5.2.2 Occlusion based controller with communication (`dec-com` controller)

The `dec-com` controller variant uses the same information as the `dec` controller. Additionally, it takes into account sensor information from the parallel face [111]. As a result, a pump of a given face is only active ($p = 0$) if it is occluded ($d = 0$) and the parallel external face is not occluded ($d' = 1$).

Table 3.4 shows the truth table of the controller for each face based on the sensor readings and information from the paired face. When two modules are connected, the information is propagated through the communication between the modules.

Table 3.4 Truth table for the `dec-com` controller We use p to represent the pump activation state; d and c represent the light and connection sensor readings of the face, respectively, and d' the sensor reading of the paired external face.

c	d	d'	p
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

The controller is implemented as described in Algorithm 2. For each face the controller reacts not only based on its own sensor readings, but also based on the whether the paired external face detects the goal or not. As with the previous controller, the `dec-com` can run on a face level and requires no run-time memory. However, in our implementation we use memory to store messages coming from neighbouring modules. As the controller cycles through the faces, and a connected face receiving a message will not use it, the message need to be stored until the paired face requires that information to assess its pump activation state.

Lines 7–19 implement the steps necessary for each module to react in accordance with the presented in Table 3.4. If a connected module is detected by the current face, the module starts by reading the message coming from the neighbouring module. The message is received by the IR transceiver, and corresponds to a 1 or 0, for whether the parallel face of the neighbouring module is able or not to detect the goal, respectively. The message is stored in place of the current face's light sensor reading. The module then sends the paired face sensor reading to its neighbour. If the current face is not connected, the reading of the light sensor is done as for the `dec` controller. Then, the module makes a decision on whether to activate the pump based on the connection sensor and the light sensor of the current face, as well as the light sensor of the paired face. For this controller, we also ensure the pump states are only updated up to ten times per second (lines 22–24). In this case, is not only due to the limited response time of the pumps, but also to minimise errors in the communication.

Algorithm 2 `dec-com` controller implementation

```
1:  Let $k$ and $l$ be the current and paired faces; $c_k$, $d_k$ and $p_k$ are the states of the connection
    and light sensors and pump of face $k$, respectively.
2:  Le $m_k$ be the message sent by the module's face $k$.
3:  $n_{faces} = 4$
4:  $t_{cycle} = 100$                                        ▷ Time in milliseconds.
5:  loop
6:      $t_1 \leftarrow$ time
7:      for $k \leftarrow 1, n_{faces}$ do                   ▷ Go through all faces.
8:          $c_k \leftarrow$ Connection sensor reading        ▷ Get connection sensor readings.
9:          if $c_k = 1$ then                                ▷ If connected to another module.
10:             $d_k \leftarrow$ Message                      ▷ Read message from neighbouring module.
11:             $m_k \leftarrow d_l$     ▷ Send paired face sensor reading to neighbouring module.
12:         else
13:             $d_k \leftarrow$ Goal sensor reading          ▷ Get light sensor readings.
14:         end if
15:         if $c_k = 0$ and $d_k = 0$ and $d_l = 1$ then    ▷ If external, occluded and paired.
16:             $p_k \leftarrow 1$
17:         else                                             ▷ Otherwise, deactivate pump.
18:             $p_k \leftarrow 0$
19:         end if
20:     end for
21:     $t_2 \leftarrow$ time
22:     if $t_2 - t_1 < t_{cycle}$ then   ▷ If the loop was faster than the minimum cycle time, wait.
23:         wait($t_{cycle} - (t_2 - t_1)$)
24:     end if
25: end loop
```

3.5.3 Results

In this section we present and discuss the results obtained in the physical experiments. We compare the performance of the two controllers using three metrics—time taken, energy consumed and power used. We also compare the trajectories the robots follow with the trajectories predicted under our assumptions. The `dec` and `dec-com` controllers had a success rate of 90% (27 out of 30 trials) and 100% (all 30 trials), respectively. Figure 3.8 shows a series of snapshots from a typical trial using the `dec` controller.

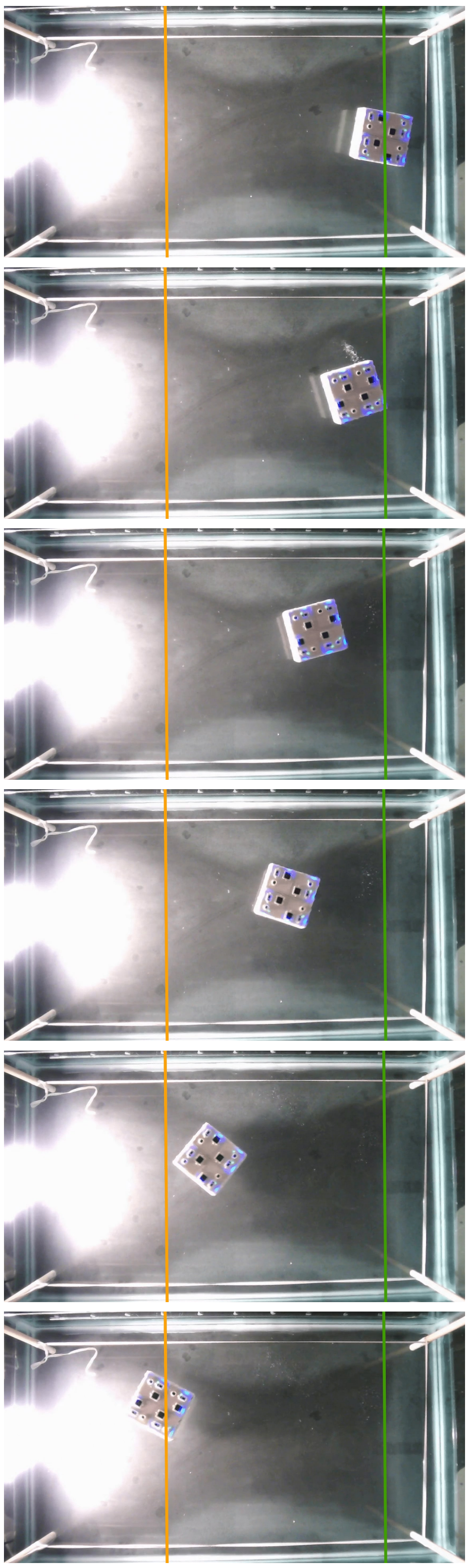


Fig. 3.8 Snapshots of a trial with a physical 2×2 MHP robot using the `dec` controller, taken at (from top to bottom) $t = 0\,\text{s}$, $t = 4\,\text{s}$, $t = 9\,\text{s}$, $t = 12\,\text{s}$, $t = 18\,\text{s}$, and $t = 22\,\text{s}$. The green and orange lines represent the start and finish lines respectively.

Table 3.5 Summary of the results for time, energy and power for the two controllers.

Metric	Controller	Min.	Max.	Avg.
Time [s]	`dec`	12.0	32.0	19.5
	`dec-com`	13.0	95.0	38.0
Energy [eu]	`dec`	53.0	214.0	118.9
	`dec-com`	18.0	84.0	34.8
Power [pu]	`dec`	4.42	7.21	6.03
	`dec-com`	0.37	2.06	1.09

3.5.3.1 Time, energy and power results

To obtain an estimate of the energy expenditure we define 1 energy unit (eu) as the amount of energy expended by a single pump in a single frame (1 s)[1]. The energy $E(t)$ (in energy units) used by an MHP robot, comprised of N modules, at a given time frame t is given by:

$$E(t) = \sum_{e=1}^{N} \sum_{j=1}^{f} p_{ej}, \tag{3.2}$$

where p_{ej} corresponds to the activation state of the pump belonging to module e in face j, and f to the number of faces of each module ($f = 4$ for robots moving in 2D and $f = 6$ for robots moving in 3D). The total energy E_T used in a trial with T frames is given by:

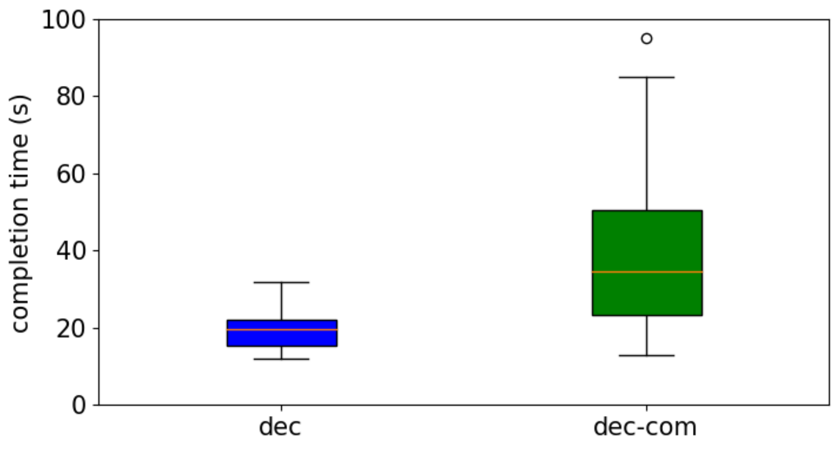
$$E_T = \sum_{t=1}^{T} E(t), \tag{3.3}$$

Additionally, we define 1 power unit (pu) as the number of energy units expended by the MHP robot per frame. The power P (in power units) used by a 2×2 MHP robot in a trial is given by:

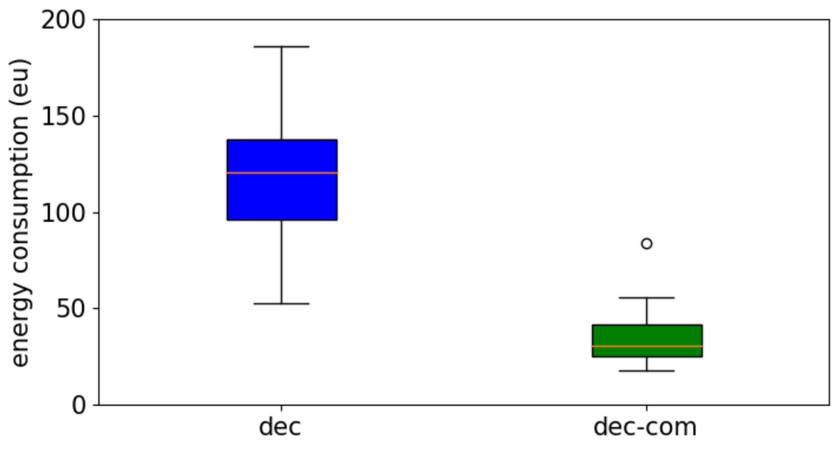
$$P = \frac{E_T}{T}. \tag{3.4}$$

We calculate the performance metrics from the video footage obtained conducting the experiments. Figure 3.9 compares the results of each controller for time taken (Figure 3.9a), energy consumption (Figure 3.9b) and power usage (Figure 3.9c). Table 3.5 summarises the results for each set of trials.

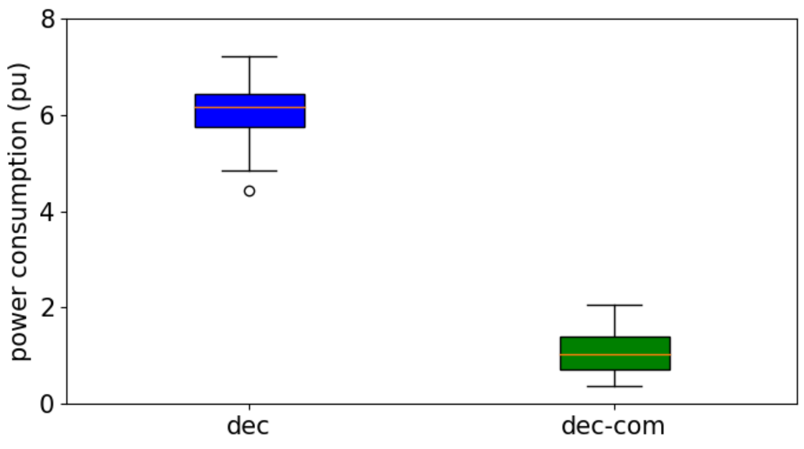
[1]As the modules do not have capabilities that allow them to measure the energy used by a pump, we determine the amount of energy expended form the number of active pumps during the trial. Additionally, we assume that all pumps use the same amount of energy



(a)



(b)



(c)

Fig. 3.9 Time taken, energy consumed and power required for a physical 2×2 MHP robot to move by 60 cm towards the goal using the `dec` and `dec-com` controllers. Successful trials only; the boxes on the left and right represent 27 and 30 trials, respectively.

The dec and dec-com controllers, while having a similar minimum completion time, require an average time of 19.5 s and 38.0 s, respectively, across the successful trials. This difference is significant, and corresponds to an increase in time taken of 94.9% when using the dec-com controller. When the controllers were introduced, it was expected that completion times for both controllers should be identical. The dec-com controller, when compared with the dec controller, saves energy by not actuating pumps that have no impact on the net force acting on the robot. While similar minimum completion times are in agreement with this, the average and maximum completion times are not.

The robot uses significantly less energy at a significantly lower power rate when using the dec-com controller over the dec controller. The successful trials of the dec controller use, on average, 118.9 energy units per trial at a rate of 6.03 power units . The successful trials of the dec-com controller use, on average, 34.8 energy units per trial at a rate of 1.09 power units. This corresponds, on average, to an energy reduction of 70.7% and a power reduction of 81.9% when using the dec-com controller, rather than the dec controller. Note that for the energy comparison, the minimum energy consumed by the dec controller is similar to the maximum energy spent by the dec-com controller, which agrees with what is expected from the controller. However, the average energy reduction is, once more, higher than the maximum expected value (66%).

From the definition of power units, it would be theoretically expected that the reduction in power would be the same as the reduction in energy. However, as the completion times are significantly different, the reduction in power usage is greater than the reduction in energy consumption. This also justifies the differences in range o power units used. We expected the dec controller to range between 4 and 6 power units used while the dec-com controller should range between 2 and 4 power units used. Neither controller performed within the expected ranges (Table 3.5). However, the discrepancy between the results and theory is higher for the dec-com controller.

The differences between the results and the expected outcome will be discussed in the next Section.

3.5.3.2 In-depth investigation

From the differences in the behaviours of controllers illustrated in Figure 3.6, and since the force acting on the robot and the point of application remain the same, the theory suggests that other than a reduction of the number of pumps active, there should be no differences, such

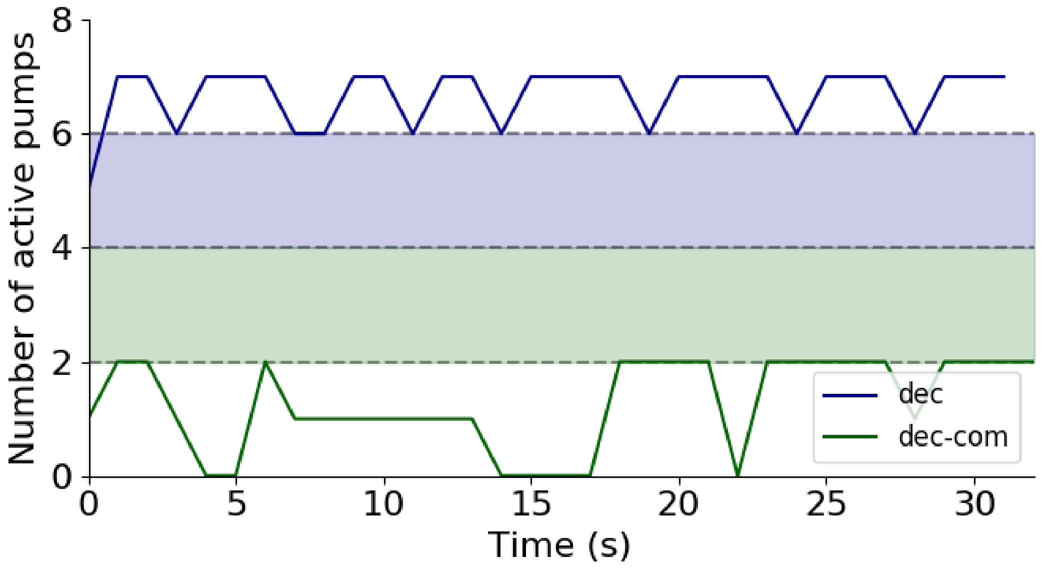


Fig. 3.10 Number of active pumps of a physical 2×2 MHP robot during two successful trials using the `dec` controller and `dec-com` controller, respectively. Shaded areas correspond to the respective theoretically possible range (see Fig. 3.11).

as completion time and trajectory. However, we observed significant differences between the experimental results and the expected results. However, when testing a robotic system in reality some discrepancies are to be expected. This phenomenon is referred to as the reality gap [112]. In order to understand how this reality gap manifests itself in our results, we analyse two successful trials of similar length, one for each controller.

Figure 3.10 compares the number of pumps active throughout each trial. For the `dec` controller between five and seven pump were active at all times. For the `dec-com` controller between zero and two pumps were active. In both cases this represents a departure from how each controller should work in theory. Figure 3.11 shows the possible actuator firing configurations (excluding symmetric ones), under the assumptions of the model presented in [111] . Either four or six pumps should be active for the `dec` controller, while either two or four pumps should be active for the `dec-com` controller.

One explanation for this difference is the detection angle of the goal sensors. The model, presented in [111], on how the robots behave when using the controllers, rests on the assumption that the sensors have a hemispherical field-of-view. In reality, the field of view is smaller than this. For this reason, false-negative readings (situations where even though the robot is not able to detect the goal, the proposed model of the sensors presumes

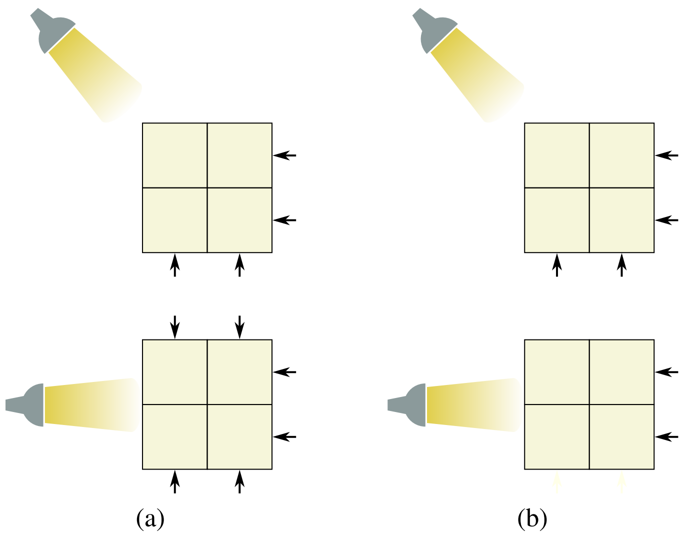


Fig. 3.11 Possible actuator firing configurations for 2×2 MHP robots assuming hemispherical sensors, a single point goal and a fault-free system. (a) Robots running the `dec` controllers. (b) Robots running the `dec-com` controllers. Arrows indicate the forces caused by active pumps. When using the `dec-com` controller, communication prevents paired faces to both activate their pumps.

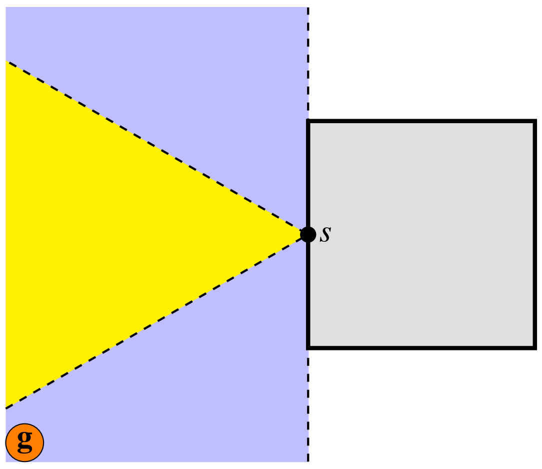


Fig. 3.12 Example of a false negative sensor reading. The real sensing angle is represented in yellow while the theoretical sensing angle is represented in blue. According to the theoretical model, the goal should be detected (it is in the blue area). However, the sensors used are not able to detect it (goal is outside yellow area), leading to a false negative reading.

it is), can occur (see Figure 3.12). From the observation of the trials' footage, while some false positives were identified (due to light reflections on the water tank glass walls), the predominant error observed was false-negative sensor readings.

From the sample trials it is apparent that the presence of false-negative readings affect the two controllers in different ways. When using the `dec` controller, false-negative readings cause an activation of pumps, resulting in a higher than expected number of pumps firing. Conversely, when using the `dec-com` controller, false-negative reading cause a deactivation of pumps, resulting in a lower than expected number of pumps firing. Indeed there are periods of time when no pumps are firing; the robot then moves only due to inertia.

This can help justify the results obtained in Section 3.5.3.1. When using the `dec-com` controller, false-negative readings cause the robot to undergo periods of time when no pumps are firing, slowing its progress. This can be particularly significant if it happens at the start of a trial, where inertial forces are minimal and the robot has to drift slowly into a position in which a sensor detects the goal. Even after detecting the goal for the first time, the moments when no pumps are firing impact the trial completion time greatly, for the `dec-com` controller, as the movement speed of the robot is greatly diminished. The different reactions to false negatives contribute to the energy consumption reduction being higher than expected (based on hemispherical sensors). Not only can a robot using the `dec-com` controller use fewer pumps than expected, a robot using `dec` controller can use more pumps than expected.

Another element that affects the experimental results is that the robot tends to undergo rotational movements, whereas the theory predicts purely translational movement. This can be attributed not only to the aforementioned false-negative readings, but also to noise and inaccuracies in the actuators. When developing the controller, one of the assumptions used was that each active pump produces the same constant force along the face normal. However, the physical pumps produce forces of similar, but not identical, magnitudes, which can also be slightly offset from the face normal. In addition, the internal fluid routing through the robot, which is not considered by the controller, could have an effect. On the other hand, while the presence of noise and inaccuracies in the actuators has an effect on performance, it also helps to show the robustness of the controllers to these elements.

3.5.3.3 Trajectory analysis

As both controllers are expected to produce the same net force, it is expected that both controllers produce similar trajectories. While the trajectory for each trial is dependent on the

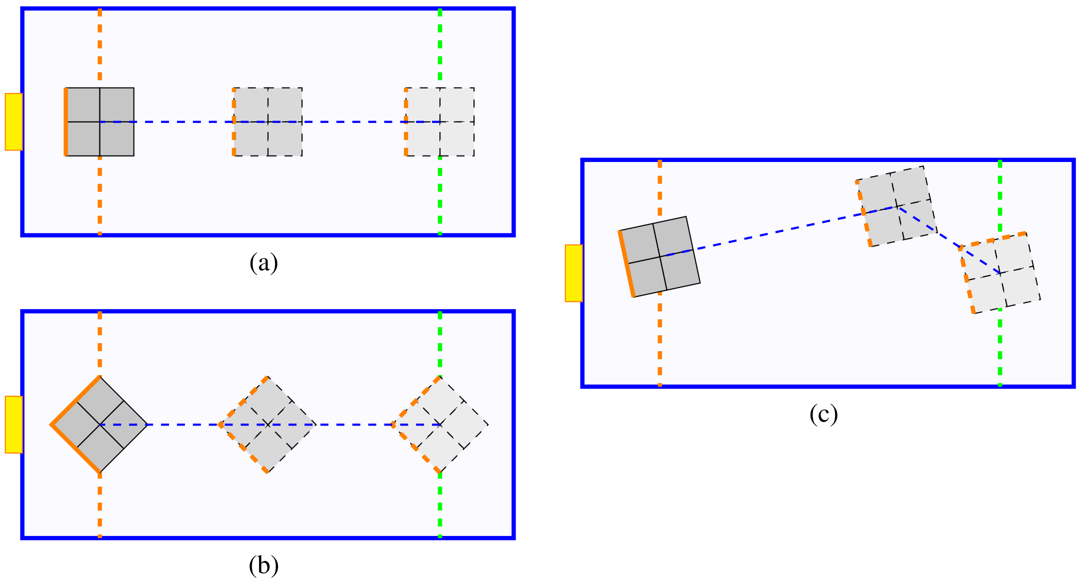


Fig. 3.13 Expected trajectories for a 2×2 MHP robot. The green and orange dashed lines represent the start and finish line, respectively. The blue dashed line shows the robot trajectory for each case. (a) Robot with a single face detecting the goal. (b) Robot at a $45\,^{\circ}$ angle; two faces detecting the goal. (c) Robot at an angle, different from $45\,^{\circ}$; two faces detecting the goal. For (a) and (b) the robot moves in a straight line; for (c) the robot moves fist into a position where a single face detects the goal, and then it moves in a straight line.

starting orientation, there are only two generic forms the trajectories should take, based on the theoretical analysis of the controllers. The two trajectory forms are shown in Figure 3.13, for a 2×2 robot . If in the initial orientation, the modules' face normals are either parallel, perpendicular or at $45\,^{\circ}$ angle to the shortest line connecting the centre of the robot and the goal, the robot travels in a straight line (Figure 3.13a). The net force generated by the robot in this situation is perpendicular to the faces detecting the goal, and does not change—the robot's sensing status does not change. In any other situation the robot trajectory is divided in two segments (Figure 3.13c). The length and angle of the segments is dependent on the initial orientation of the robot. Since two different robot faces detect the target, the net force acting on the robot moves it in the direction of the corner formed by the two faces detecting the goal. The robot will move in this fashion until the sensing status changes, so that a single robot face detects the goal.

Figure 3.14 presents the experimental and expected trajectories of the robot when travelling towards the goal for the `dec` and `dec-com` controllers. The initial position and orientation of the robot, as well as its trajectory, for each trial was obtained by using tracking software (using the OpenCV computer vision library [113]) on the footage of the trial. The expected

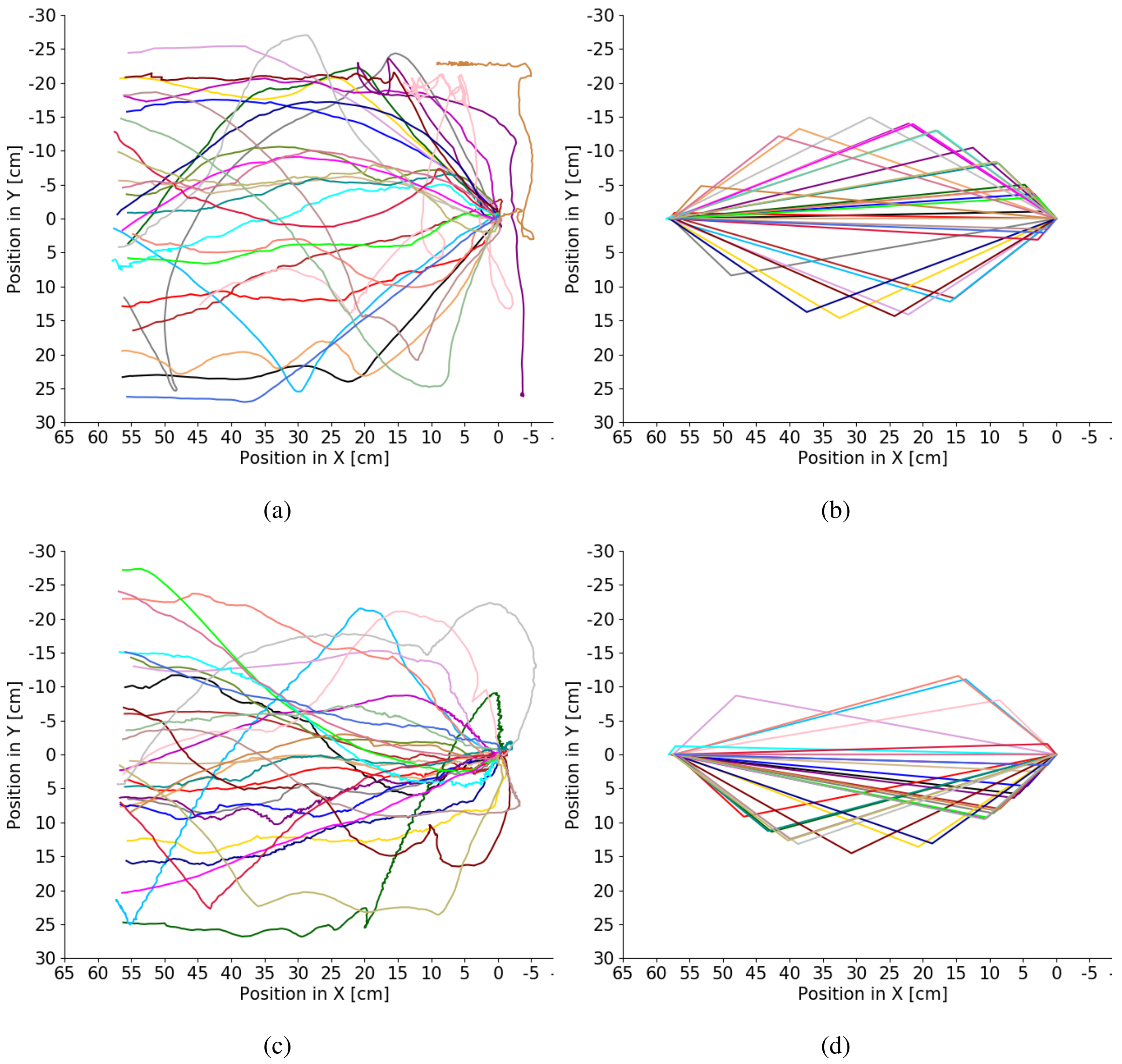


Fig. 3.14 Trajectories of the robot when traveling towards the goal. (a) Experimental trajectories for the `dec` controller. (b) Theoretical trajectories for the `dec` controller. (c) Experimental trajectories for the `dec-com` controller. (d) Theoretical trajectories for the `dec-com` controller. The theoretical trajectories were determined from the robot's initial orientation for each trial.

trajectories are obtained by taking into account the robots initial orientation in each trial. All of the initial positions were corrected so that the robot starts at the origin, for every trial.

From the analysis of Figure 3.14a we can see that the `dec` controller produces a widespread set of trajectories, with the robots frequently reaching the walls of the water tank. On the other hand, the trajectories of the robot, when using the `dec-com` controller (Figure 3.14c), pass more frequently through the center of the water tank. This difference could, once again, be related to the different reactions of each controller to the false-negative sensor readings. As the `dec` controller has a higher number of pumps active, at the beginning of each trial, it is faster at moving out of the expected trajectory. For the `dec-com` controller, as the number of active pumps is low, and the robot is frequently moving in and out of situations where no sensor detects light, the robot has fewer opportunities to leave the centre of the water tank.

While some of the experimental trajectories have similarities with the theoretical trajectories, in terms of expected behaviour, the majority is different. The differences between the theoretical and experimental trajectories can be largely attributed to the same reasons previously stated in Section 3.5.3.2—false-negative sensor readings and actuation noise. As stated before, false-negative readings lead to a different number of active pumps than the theoretical expected. A different number of active pumps changes the net force acting on the robot. The actuation noise adds to this as it causes fluctuations in the net force, even with constant sensor readings. The rotation caused by these two factors also plays a role in changing the trajectory, as changes in the orientation of the robot lead to a different net force.

Additionally, one other factor that contributes to different experimental and theoretical trajectories is actuator failure. Even if the robot is in a state where no false negative occur, if one or more actuators fail, the net force generated will differ.

3.6 Summary

This chapter presented a new implementation of the MHP concept, the MHP 2 robot. An MHP robot consists in a group of modules, physically connected (forming a single structure) that route fluid between each other. Each module is a square/cube, and contains an internal fluid reservoir. The fluid routing is performed by micropumps, that connect ports at the center of each module's face and the internal reservoir. The robot's motive force is generated through the routing of fluid between the environment and the modules' internal reservoirs.

So that it would be possible to test any developed strategies in the real world, a physical implementation of the MHP concept was required. While an initial hardware prototype of the system existed, it presented some limitations that prevented further work with the system. We described limitations of the original system and how to overcome them in the design process. We built six modules, where each module is able to actuate on the horizontal plane, detect a goal (i.e. light source), communicate with connected modules and detect obstacles within a limited range.

While the MHP 2 represents an improvement over the initial hardware prototype, it is not without its limitations. While the amplification of the sensor readings allowed to design the detection range of the modules to the environment where the modules will be used, the values obtained were prone to high frequency noise introduced by the power source. The addition of capacitors to each amplifier was required, to filter the sensor readings. While this proved to be effective, the addition of components to the modules after completion introduces additional failure points. One additional limitation of the system is the number of hardware serial ports available in the ATMega324P. The lack of additional hardware serial ports introduces complexity in the communication between modules. While this does not limit the functionality and applications of the MHP 2, it once more introduces additional points of failure in the system.

We reported on a series of experiments examining the performance of the `dec` and `dec-com` controllers, in order to evaluate how the inclusion of communication in the controller impacts the performance of the robot. The two controllers completed the task in 90% and 100% of the experimental trials. When comparing the performance of the two controllers, the `dec-com` controller saved on average 70.7% energy at the expense of increasing the time taken to reach the target on average by 94.9%.

From the results of the experiments, the addition of communication, for sharing sensing information between modules, has an impact on the performance of the robot in a translation task. However, translation is a single component of the movement a robot can perform. In the next chapter we will be looking into the rotation component of the movement, which will allow a robot to control not only its position, but also its orientation. We will develop rotation controllers and, in a similar fashion to what was performed in this chapter, evaluate the impact information sharing between modules can have on performance.

Chapter 4

Decentralised and Reactive Pose Control of an MHP Robot

This chapter is based and extends on the author's original contributions to Sections 1–4 of the publication [114].

4.1 Introduction

In the previous chapter we tested the performance of an occlusion-based controller in allowing an MHP robot to reach a goal. Additionally, the impact communication has on the robot's performance was tested. Although the ability to reaching a goal can be relevant, some applications require the control of orientation of the robot to successfully complete the task. Potential applications include inspection of underwater environments [82], where robots need to reorient themselves to collect information from different angles, construction of temporary structures [80], where robots are only able to connect to the structure if in the correct orientation, and navigation in narrow passages, where the robot needs to be correctly oriented to prevent collisions. In this chapter we look into minimalistic strategies for controlling the pose of MHP robots. Specifically, we are interested in fully reactive and decentralised control strategies.

Multiple solutions for controlling the orientation of modular robots can be found in the literature. However, these solutions often require either centralised control [79, 84], external sensors [63] or a significant level of computational capabilities from the robot [64]. Such resources, when reducing the modules' size, are impractical, leading the controller design to be minimalistic.

In this chapter, we aim to develop controller solutions that allow for the control of an MHP robot's pose. The control strategies should be decentralised and fully reactive, not requiring run-time memory or arithmetic computations. This will allow them to be implemented in robots with limited computational power. We look into solutions with different levels of complexity, in order to be able to evaluate the trade off between complexity of the solution and performance.

The remainder of this chapter starts with an analysis of the range of unique total forces and net torques that can be produced by a 3D MHP robot of convex shape in Section 4.2. Section 4.3 presents the problem formulation and Section 4.4 details the controller solutions. The controllers are formally analysed in Section 4.5 for convex shaped robots. Section 4.6 extends the analysis for non-convex shaped robots. Section 4.7 concludes the chapter.

4.2 Force and torque analysis of an MHP robot in 3D

We aim to develop controllers that allow for efficient movement. The knowledge of the range of possible actuation states, and respective forces and torques, can give perspective on the possible amount of solutions for this problem. Additionally, this knowledge could prove relevant for the development of control strategies not considered in this work. In this section, we make an analysis on the range of forces and torques an MHP robot can generate. That is to say, we derive the number of actuation states that produce unique total forces and net torques, acting on the robot's center of mass, for an MHP robot. The analysis focuses on the possible actuation states, independently of controller used. This analysis gives an understanding of the type and degree of granularity of movements the robot can perform. The analysis of the range of movement of a 2D convex (rectangular) MHP robot was performed in [115]. Here, we extend the analysis for 3D convex robots.

The robot consists of n connected cubic modules, that form a structure of dimensions $a_1\varepsilon \times a_2\varepsilon \times a_3\varepsilon$, where ε is the module side length, and a_1, a_2 and a_3 are the number of modules in the robot along the X_1, X_2 and X_3 axes, respectively. The vector $\mathbf{x}_e$ describes the position of module e with regard to the robot's centre. For simplicity, we assume all modules share a common reference frame, that is, the reference frame of each module is aligned with the robot's reference frame. For a robot moving in an N-dimensional space, the normals to

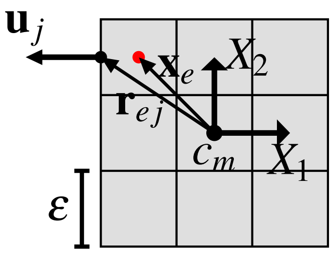


Fig. 4.1 Illustration of an 3×3 robot introduced the notation used in Section 4.2. $\mathbf{x}_e$ represents the vector describing the position of module e (top left corner module) with regard to the robot's center of mass (c_m) . $\mathbf{u}_j$ represents the outward normal vector of face j. $\mathbf{r}_{ej}$ represent the position of actuator p_{ej} of module e in face j.

the modules' faces are given by $\mathbf{u}_j = (-1)^j \mathbf{n}_i$ where $j = 2i - 1$ or $j = 2i$, $\mathbf{n}_i$ is an unitary vector aligned with the X_i axis, and $i = 1, \ldots, N$ corresponds to the world dimension. The actuator p_{ej} of module e, face j is positioned at $\mathbf{r}_{ej} = \mathbf{x}_e + 0.5\varepsilon \mathbf{u}_j$. Figure 4.1 illustrates the relationship between these variables. For the remaining of this this chapter, to simplify the analysis, we set $\varepsilon = 1$.

The actuator force model is based on the following assumptions:

- Only external actuators (actuators that belong to external module faces) can be active,
- An active actuator applies a force f_p along $\mathbf{u}_j$, on the robot at $\mathbf{r}_{ej}$,
- Forces generated by internal fluid flow through the robot are neglected.

In this section, we assume that the world and robot reference frames are identical, and are henceforth referred as cardinal axes. The total force and net torque acting on a robot, due to active actuators, are then given by:

$$\mathbf{f} = -\sum_{e,j} p_{ej} f_p \mathbf{u}_j, \tag{4.1}$$

$$\tau = -\sum_{e,j} p_{ej} f_p (\mathbf{r}_{ej} \times \mathbf{u}_j) \tag{4.2}$$

4.2.1 Range of unique total forces

Theorem 1. *The number of unique total forces that a convex MHP robot, with $a_1, a_2 a_3 \in \mathbb{Z}^+$, can generate is $\eta_f^{total} = (2a_2a_3 + 1)(2a_1a_3 + 1)(2a_1a_2 + 1)$.*

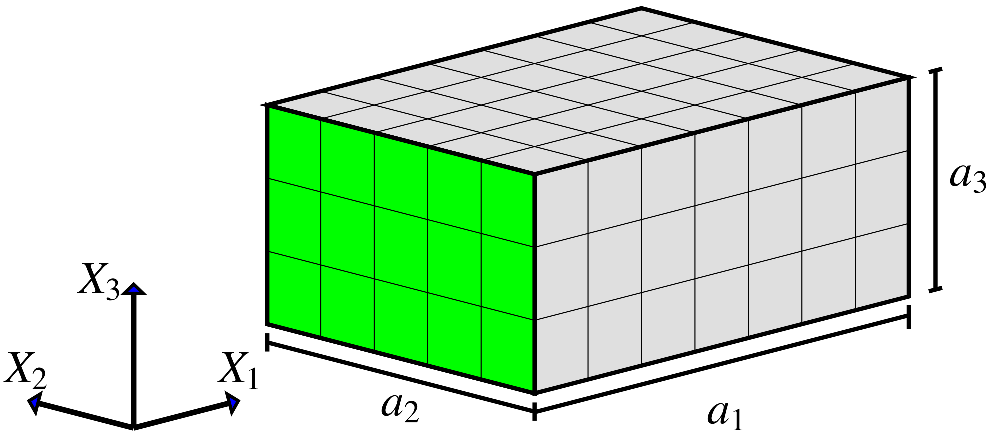


Fig. 4.2 An $a_1 \times a_2 \times a_3$ robot (all side lengths odd) with active actuator configuration that achieves maximum force along X_1. Green faces represent the faces with active actuators. No actuators fire in the non-visible faces.

Proof. Consider a convex MHP robot, with $a_1, a_2, a_3 \in \mathbb{Z}^+$. Along the X_1 axis the minimum and maximum forces occur when a_2a_3 actuators are active (see Figure 4.2);

$$f_{min}^{X_1} = -a_2a_3f_p, \tag{4.3}$$

$$f_{max}^{X_1} = a_2a_3f_p. \tag{4.4}$$

The robot can achieve any force that is an integer multiple of f_p, between $f_{min}^{X_1}$ and $f_{max}^{X_1}$, including zero. As a result, the number of unique total forces the robot can produce along X_1 is:

$$\eta_f^{X_1} = 2a_2a_3 + 1. \tag{4.5}$$

Similarly, the number of unique total forces along the X_2 and X_3 axes are given by:

$$\eta_f^{X_2} = 2a_1a_3 + 1, \tag{4.6}$$

$$\eta_f^{X_3} = 2a_1a_2 + 1. \tag{4.7}$$

As the forces acting along each axis are independent (perpendicular axes), the total number of possible forces that can be produced is :

$$\eta_f^{total} = \eta_f^{X_1}\eta_f^{X_2}\eta_f^{X_3} = (2a_2a_3 + 1)(2a_1a_3 + 1)(2a_1a_2 + 1). \tag{4.8}$$

■

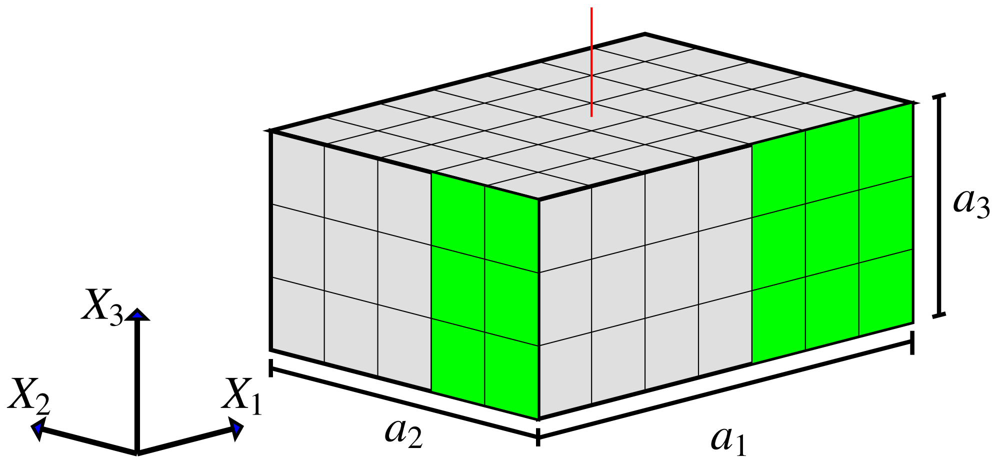


Fig. 4.3 An $a_1 \times a_2 \times a_3$ robot (all side lengths odd) with active actuator configuration that achieves maximum torque along axis X_3. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

4.2.2 Range of unique net torques

To cover all possible cases, we split the analysis of the number of unique torques into three possible situations: torque acting along one, two or three of the axes defining the cardinal axes. In other words, we consider separately the cases where the torque, when described in the cardinal axes, contains one, two or three non-zero components.

Knowing the number of possible torques for each case, the total number of torques a robot will be given by:

$$\eta_\tau^{\text{total}} = \eta_\tau^{\text{single}} + \eta_\tau^{\text{double}} + \eta_\tau^{\text{triple}}, \tag{4.9}$$

where $\eta_\tau^{\text{single}}$, $\eta_\tau^{\text{double}}$ and $\eta_\tau^{\text{triple}}$ are the number of unique torques with one, two or three non-zero components, respectively.

4.2.2.1 Torque with a single non-zero component

Consider a convex robot, with $a_1, a_2, a_3 \in 2\mathbb{Z}^+ - 1$ (all robot side lengths have an odd number of modules). We first treat the case where the torque has a single non-zero component, applied along either X_1, X_2, or X_3.

The maximum torque along X_3 is obtained by:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_3^\top(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise}, \end{cases} \tag{4.10}$$

where $\mathbf{n}_3^\top$ is the transpose of the vector defining axis X_3 and c_{ej} represents the connection status of face j of module e and is defined by:

$$c_{ej} = \begin{cases} 1, & \text{face } j \text{ of module } e \text{ is connected with another module;} \\ 0, & \text{otherwise.} \end{cases} \tag{4.11}$$

Figure 4.3 illustrates the actuator firing configuration described in Equation (4.10). Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_3}(a_1,a_2,a_3) &= 2a_3\left[\left(1+2+\cdots+\frac{a_1-1}{2}\right)+\left(1+2+\cdots+\frac{a_2-1}{2}\right)\right] \\ &= a_3\left(\frac{(a_1-1)(a_1+1)}{4}+\frac{(a_2-1)(a_2+1)}{4}\right)=\frac{a_3(a_1^2+a_2^2-2)}{4}. \end{aligned} \tag{4.12}$$

Similarly, the minimum achievable torque is $\tau_{min}^{X_3}(a_1,a_2,a_3) = -\tau_{max}^{X_3}(a_1,a_2,a_3)$. The same analysis can be performed for axis X_1 and X_2, obtaining:

$$\tau_{max}^{X_1}(a_1,a_2,a_3) = \frac{a_1(a_2^2+a_3^2-2)}{4}, \tag{4.13}$$

$$\tau_{max}^{X_2}(a_1,a_2,a_3) = \frac{a_2(a_1^2+a_3^2-2)}{4}. \tag{4.14}$$

Remark. *To ensure the rotation axis is constant, when torque is applied along X_3, a pair of actuators p_{ej} placed at $\mathbf{r}_{ej} = (\frac{a_1-1}{2},x_2,x_3)$ and $\mathbf{r}_{ej} = (\frac{a_1-1}{2},x_2,-x_3)$, need to have the same activation status, if $x_3 \neq 0$. Similarly, a pair of actuators p_{ej} placed at $\mathbf{r}_{ej} = (x_1,\frac{a_2-1}{2},x_3)$ and $\mathbf{r}_{ej} = (x_1,\frac{a_2-1}{2},-x_3)$ need to have the same activation status.*

Lemma 1. *For the case that $a_1,a_2,a_3 \in 2\mathbb{Z}^+ - 1$, the robot can produce every integer torque $\tau^{X_3} \in \{-\tau_{max}^{X_3}(a_1,a_2,a_3),\ldots,\tau_{max}^{X_3}(a_1,a_2,a_3)\}$, along the X_3 axis.*

Proof. We prove Lemma 1 by induction.

Base case: Consider the cases where $a_3 = 1$. If $a_1 = 1 \wedge a_2 = 1$, the robot consists of a single module; from Equation (4.2), no torque can be produced. If $a_1, a_2 = 1, 3, \ldots, k, \forall k \in 2\mathbb{Z}^+ - 1$, when the non-zero component of the torque is applied along X_3, this corresponds to the 2D case presented in [115] and every integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,1), \ldots, \tau^{X_3}_{max}(a_1,a_2,1)\}$ can be produced along X_3 axis.

Inductive step: We assume that Lemma 1 is correct for $a_3 = 1, 3, \ldots, k, \forall k \in 2\mathbb{Z}^+ - 1$. When $a_1 \times a_2 \times 1$ modules are added along X_3, symmetrically about the robot's center of mass, the position of the existing modules does not change. This means that if the robot was able to produce a given torque for $a_3 = k$, that same torque can be produced for $a_3 = k+2$. For this reason, all integer torque values $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,k), \ldots, \tau^{X_3}_{max}(a_1,a_2,k)\}$ can be produced.

Consider now the situation where the maximum torque $\tau^{X_3}_{max}(a_1,a_2,k+2) = \frac{(k+2)(a_1^2+a_2^2-2)}{4}$ is produced. Each of the two layers of $a_1 \times a_2 \times 1$ added modules is capable of producing every integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,1), \ldots, \tau^{X_3}_{max}(a_1,a_2,1)\}$. So that the rotation axis does not change, the two sets of $a_1 \times a_2 \times 1$ added modules are capable of producing every even torque $\tau^{X_3} \in \{0, 2, \ldots, 2\tau^{X_3}_{max}(a_1,a_2,1) - 2, 2\tau^{X_3}_{max}(a_1,a_2,1)\}$. The odd torque values $\tau^{X_3} \in \{1, 3, \ldots, 2\tau^{X_3}_{max}(a_1,a_2,1) - 1\}$ can be produced by activating/deactivating pumps p_{ej} for which $\mathbf{r}_{ej} = (\frac{a_1-1}{2}, 1, 0)$ or $\mathbf{r}_{ej} = (1, \frac{a_2-1}{2}, 0)$. As such all integer torque values $\tau^{X_3} \in \{0, \ldots, 2\tau^{X_3}_{max}(a_1,a_2,1)\}$ can be produced.

Given that $\tau^{X_3}_{max}(a_1,a_2,k+2) - \tau^{X_3}_{max}(a_1,a_2,k) = 2\frac{a_1^2+a_2^2-2}{4} = 2\tau^{X_3}_{max}(a_1,a_2,1)$, the robot is able to produce every integer torque $\tau^{X^3} \in \{\tau^{X_3}_{max}(a_1,a_2,k+2) - \tau^{X_3}_{max}(a_1,a_2,k), \ldots, \tau^{X_3}_{max}(a_1, a_2,k+2)\}$. By symmetry, the robot is able to produce every integer torque $\tau^{X^3} \in \{-\tau^{X_3}_{max}(a_1,a_2, k+2), \ldots, -\tau^{X_3}_{max}(a_1,a_2,k+2) + \tau^{X_3}_{max}(a_1,a_2,k)\}$. Therefore, the robot can produce every integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,k+2), \ldots, \tau^{X_3}_{max}(a_1,a_2,k+2)\}$ which concludes the inductive step. ■

Theorem 2. *The number of unique net torques that a 3D convex MHP robot with* $a_1, a_2, a_3 \in 2\mathbb{Z}^+ - 1$ *is able to generate, with non-zero component along* X_3 *is* $\eta_{\tau^{X_3}}(a_1,a_2,a_3) = a_3\frac{a_1^2+a_2^2-2}{2} + 1$.

Proof. The minimum and maximum net torques the robot can generate are $-\tau^{X_3}_{max}(a_1,a_2,a_3)$ and $\tau^{X_3}_{max}(a_1,a_2,a_3)$, respectively. From Lemma 1, the robot is able to produce all integer torques between $-\tau^{X_3}_{max}(a_1,a_2,a_3)$ and $\tau^{X_3}_{max}(a_1,a_2,a_3)$. Since each actuator can only contribute with an integer torque, the robot is not able to produce a non integer net torque value.

Table 4.1 Number of unique torques for robots of different combinations of odd ($2\mathbb{Z}^+ - 1$) and even ($2\mathbb{Z}^+$) side lengths.

Robot side lengths			Axis		
$a_1 \in$	$a_2 \in$	$a_3 \in$	X_1	X_2	X_3
$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+$	$a_1(a_2^2 + a_3^2 - 1) + 1$	$a_2(a_1^2 + a_3^2 - 1) + 1$	$a_3\frac{a_1^2+a_2^2-2}{4} + 1$
$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+$	$2\mathbb{Z}^+ - 1$	$a_1(a_2^2 + a_3^2 - 1) + 1$	$a_2\frac{a_1^2+a_3^2-2}{4} + 1$	$a_3(a_1^2 + a_2^2 - 1) + 1$
$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+$	$2\mathbb{Z}^+$	$a_1(a_2^2 + a_3^2) + 1$	$a_2\frac{a_1^2+a_3^2-1}{2} + 1$	$a_3\frac{a_1^2+a_2^2-1}{2} + 1$
$2\mathbb{Z}^+$	$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+ - 1$	$a_1\frac{a_2^2+a_3^2-2}{4} + 1$	$a_2(a_1^2 + a_3^2 - 1) + 1$	$a_3(a_1^2 + a_2^2 - 1) + 1$
$2\mathbb{Z}^+$	$2\mathbb{Z}^+ - 1$	$2\mathbb{Z}^+$	$a_1\frac{a_2^2+a_3^2-1}{2} + 1$	$a_2(a_1^2 + a_3^2) + 1$	$a_3\frac{a_1^2+a_2^2-1}{2} + 1$
$2\mathbb{Z}^+$	$2\mathbb{Z}^+$	$2\mathbb{Z}^+ - 1$	$a_1\frac{a_2^2+a_3^2-1}{2} + 1$	$a_2\frac{a_1^2+a_3^2-1}{2} + 1$	$a_3(a_1^2 + a_2^2) + 1$
$2\mathbb{Z}^+$	$2\mathbb{Z}^+$	$2\mathbb{Z}^+$	$a_1\frac{a_2^2+a_3^2}{2} + 1$	$a_2\frac{a_1^2+a_3^2}{2} + 1$	$a_3\frac{a_1^2+a_2^2}{2} + 1$

As a result, the number of unique net torques the robot is able to produce is given by:

$$\eta_{\tau^{X_3}}(a_1, a_2, a_3) = 2\tau_{max}^{X_3}(a_1, a_2, a_3) + 1 = a_3\frac{a_1^2 + a_2^2 - 2}{2} + 1. \tag{4.15}$$

■

Using the same analysis, when the torque is applied along X_1 or X_2, the number of unique possible torques is given by:

$$\eta_{\tau^{X_1}}(a_1, a_2, a_3) = 2\tau_{max}^{X_1}(a_1, a_2, a_3) + 1 = a_1\frac{a_2^2 + a_3^2 - 2}{2} + 1, \tag{4.16}$$

$$\eta_{\tau^{X_2}}(a_1, a_2, a_3) = 2\tau_{max}^{X_2}(a_1, a_2, a_3) + 1 = a_2\frac{a_1^2 + a_3^2 - 2}{2} + 1. \tag{4.17}$$

The expressions obtained for the number of unique torques are only valid for robots with all side lengths odd. However, robots with all side lengths even, or a combination of even and odd side lengths are possible. Through the use of a similar analysis (presented in Appendix B) we can obtain the number of unique torques possible with a single non-zero component along one of the cardinal axes (the resulting rotation is performed along one of the three cardinal axis). The results are presented in Table 4.1. The total number of unique torques, for a given robot, applied along a single cardinal axis, corresponds to the sum of the number of torques possible along each axis, excluding the zero torque for two of the axes;

$$\eta_{\tau}^{\text{single}} = \eta_{\tau^{X_1}} + \eta_{\tau^{X_2}} + \eta_{\tau^{X_3}} - 2. \tag{4.18}$$

4.2.2.2 Torques with two non-zero components

Due to the complexity of the problem, we restrict the analysis to robots that have $a_1 = a_2 = a_3 = a$. Consider a convex robot with $a_i \in 2\mathbb{Z}^+ - 1$ and $a_1, a_2, a_3 \geq 2$. We start by considering torques with two non-zero components, applied along X_2 and X_3, with the same magnitude and sign. In these conditions, the resulting torque is applied along a new axis $X_{2,3}$, defined by the vector $\mathbf{n}_{2,3} = (0,1,1)$[1]. The maximum torque around this axis is generated when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_{2,3}^{\top}(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{4.19}$$

The actuator firing configuration described in Equation (4.19) is illustrated in Figure 4.4. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_2}(a) &= 2\left(2\times 1 + 4\times 2 + \cdots + (a-1)\times\frac{a-1}{2}\right) + 2a\left(1 + 2 + \cdots + \frac{a-1}{2}\right) = \\ &= \frac{a^3-a}{6} + \frac{a(a^2-1)}{4} = \frac{5(a^3-a)}{12}, \end{aligned} \tag{4.20}$$

$$\begin{aligned} \tau_{max}^{X_3}(a) &= 2\left(2\times 1 + 4\times 2 + \cdots + (a-1)\times\frac{a-1}{2}\right) + 2a\left(1 + 2 + \cdots + \frac{a-1}{2}\right) = \\ &= \frac{a^3-a}{6} + \frac{a(a^2-1)}{4} = \frac{5(a^3-a)}{12}. \end{aligned} \tag{4.21}$$

To keep the $X_{2,3}$ axis constant, the components of the torque along X_2 and X_3 need to be identical, that is, $\tau^{X_2} = \tau^{X_3}$. As such, the number of unique torques along axis $X_{2,3}$ will be same as the total number of torques along X_2, that can also be produced along X_3.[2] From Lemma 1, we know that a robot with odd side lengths can produce all integer torques between $-\tau_{max}^{X_i}(a)$ and $\tau_{max}^{X_i}(a)$. As such, the number of unique torques torques along axis $X_{2,3}$ is

[1] As the torques along each of the components have the same magnitude and sign, the vector defining the axis along which the resulting torque is applied has the same characteristics, that is, same magnitude and sign for components X_2 and X_3. For simplicity we use $\mathbf{n}_{2,3} = (0,1,1)$

[2] As the robot is a cube $\eta_{\tau^{X_2}} = \eta_{\tau^{X_3}}$.

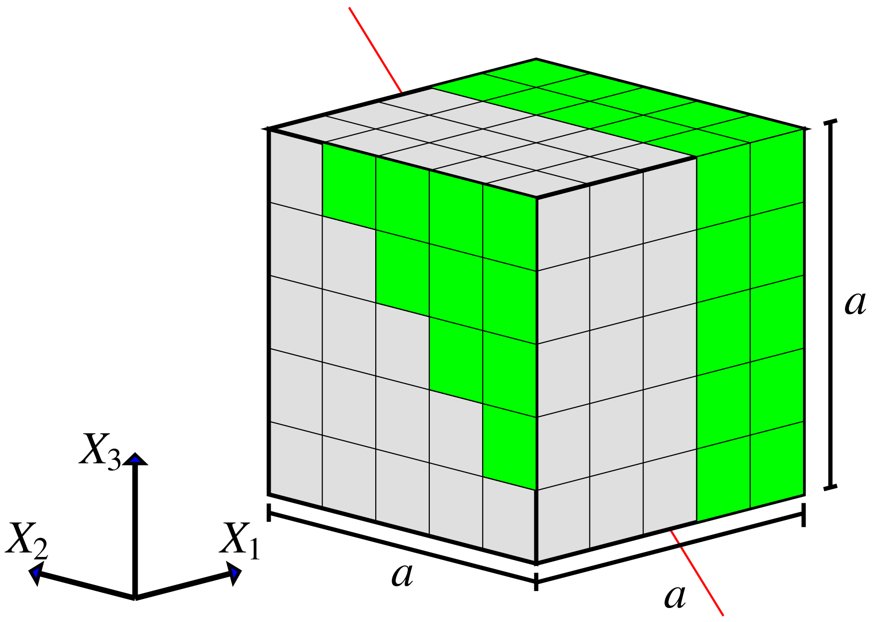


Fig. 4.4 An $a \times a \times a$ robot with active actuators (shown in green) that achieves maximum torque for the X_2 and X_3 components, simultaneously. This effectively generates a new axis, along which the resulting torque is applied. The new axis is represented by the red line, going through two of the robot edges. On the non visible faces the actuators in mirrored positions are firing.

given by:

$$\eta_{\tau^{X_{2,3}}}(a) = 2\tau^{X_2}_{max}(a) + 1 = \frac{5(a^3 - a)}{6} + 1. \tag{4.22}$$

For a robot with all side lengths with an even number of modules, and using Equation (4.19), the maximum torque is given by:

$$\tau^{X_2}_{max}(a) = 2\left(1 \times \frac{1}{2} + 3 \times \frac{3}{2} + \cdots + (a-1) \times \frac{a-1}{2}\right) + 2a\left(\frac{1}{2} + \frac{3}{2} + \cdots + \frac{a-1}{2}\right) =$$
$$= \frac{a^3 - a}{6} + \frac{a^3}{4} = \frac{5a^3 - 2a}{12}. \tag{4.23}$$

As a robot with all side lengths even can produce every half-integer torque for each torque component[3], the number of unique torques is given by:

$$\eta_{\tau^{X_{2,3}}}(a) = 4\tau_{max}^{X_2}(a) + 1 = \frac{5a^3 - 2a}{3} + 1. \tag{4.24}$$

The results obtained for the number of unique torques with two non-zero components in Equations (4.22) and (4.24) only take into account a single case—same magnitude and sign for both torque components. However, for each $\frac{\tau_{X_2}}{\tau_{X_3}}$ value, the resulting torques is applied along a different axis. As such, the number of torques that can be produced for each torque component is independent of the other. For a robot with odd side lengths, the total number of unique torques the robot can produce when the torque has non-zero components along X_2 and X_3 is given by:

$$\eta_{\tau^{X_{2,3}}} = (\eta_{\tau^{X_2}} - 1)(\eta_{\tau^{X_3}} - 1) + 1 = \left(\frac{5(a^3 - a)}{6}\right)^2 + 1, \tag{4.25}$$

where the cases removed correspond to situations where the torque along one of the axis is zero, causing a rotation along a single axis. For a robot with even side lengths, the total number of unique torques the robot can produce with non-zero X_2 and X_3 components is:

$$\eta_{\tau^{X_{2,3}}} = (\eta_{\tau^{X_2}} - 1)(\eta_{\tau^{X_3}} - 1) + 1 = \left(\frac{5a^3 - 2a}{6}\right)^2 + 1. \tag{4.26}$$

As with the case of torques with a single non-zero component, the total number of unique torques a robot can produce with two non-zero components corresponds to the sum of the number of possible net torques for each of the possible components pairs, excluding the zero torques for two of the pairs:

$$\eta_{\tau}^{\text{double}} = \eta_{\tau^{X_{1,2}}} + \eta_{\tau^{X_{1,3}}} + \eta_{\tau^{X_{2,3}}} - 2. \tag{4.27}$$

[3]As the torque has two non-zero components, and each components needs to have the same magnitude and sign, producing half integer torques for each component is possible. If the components were independent of each other, producing half-integer torques for one component would alter the torque for the other component. In this case, we assume only integer torques could be produced.

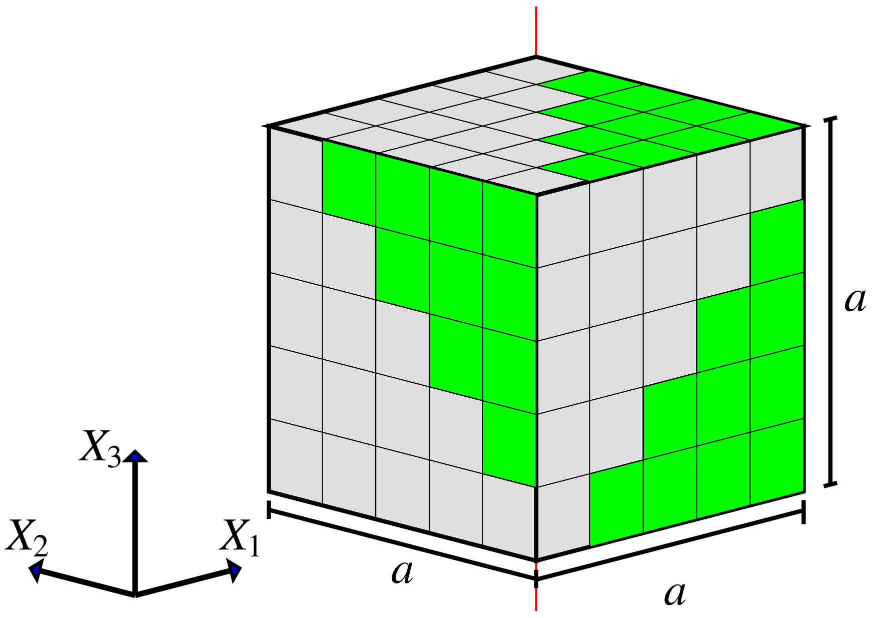


Fig. 4.5 An $a \times a \times a$ robot with active actuators (shown in green) that achieves maximum torque for all components simultaneously. This effectively generates a new axis, along which the resulting torque is applied. The new rotation axis is represented by the red line, going through two of the robot vertex. On the non-visible faces the actuators in mirrored positions are firing.

4.2.2.3 Torque with three non-zero components

Due to the complexity of the problem, we restrict the analysis to robots that have $a_1 = a_2 = a_3 = a$. Consider a convex robot with $a_i \in 2\mathbb{Z}^+ - 1$ and $a_1, a_2, a_3 \geq 2$. We start by considering torques with three non-zero components, applied along X_1, X_2 and X_3, with the same magnitude and sign. In these conditions, the resulting torque is applied along a new axis $X_{1,2,3}$, defined by the vector $\mathbf{n}_{1,2,3} = (1,1,1)$[4]. The maximum torque around this axis is generated when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_{1,2,3}^{\top}(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{4.28}$$

The actuator firing configuration described in Equation (4.28) is illustrated in Figure 4.5. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque (for each

[4]Similarly to Section 4.2.2.2, for simplicity we use $\mathbf{n}_{1,2,3} = (1,1,1)$

of the components) of:

$$\tau_{max}^{X_i}(a) = 4\left(2\times 1 + 4\times 2 + \cdots + (a-1)\times\frac{a-1}{2}\right) = \frac{a^3-a}{3}. \tag{4.29}$$

To keep the $X_{1,2,3}$ axis constant, the components of the torque along X_1, X_2 and X_3 need to be identical, that is, $\tau^{X_1} = \tau^{X_2} = \tau^{X_3}$. As such, the number of unique torques along axis $\mathbf{X}_{1,2,3}$ will be same as the total number of torques along X_1, X_2 or X_3. From Lemma 1, we know that a robot with odd side lengths can produce all integer torques between $-\tau_{max}^{X_i}(a)$ and $-\tau_{max}^{X_i}(a)$. As such, the number of unique torques torques along axis $\mathbf{X}_{1,2,3}$ is given by:

$$\eta_{\tau^{X_{1,2,3}}}(a) = 2\tau_{max}^{X_1}(a) + 1 = \frac{2(a^3-a)}{3} + 1. \tag{4.30}$$

A similar analysis can be made for all side lengths with an even number of modules. The maximum torque for such a robot is given by:

$$\tau_{max}^{X_i}(a) = 4\left(1\times\frac{1}{2} + 3\times\frac{3}{2} + \cdots + (a-1)\times\frac{a-1}{2}\right) = \frac{a^3-a}{3} \tag{4.31}$$

As a robot with all side lengths even can produce every half-integer torque for each torque component[5], the number of unique torques along $\mathbf{X}_{1,2,3}$ is given by:

$$\eta_{\tau^{X_{1,2,3}}}(a) = 4\tau_{max}^{X_1}(a) + 1 = \frac{4(a^3-a)}{3} + 1. \tag{4.32}$$

The results obtained for the number of unique torques with three non-zero components in Equations (4.30) and (4.32) only take into account a single case—same intensity and direction of rotation for torques in each axis. However, for each unique set of $(\frac{\tau_{X_1}}{\tau_{X_2}}, \frac{\tau_{X_1}}{\tau_{X_3}})$ values, the resulting torque is applied along a different axis. As such, the number of torques that can be produced for each torque component is independent of the other. For any cubic robot, the total number of unique torques the robot can produce with non-zero X_1, X_2 and X_3

[5]As the robot is rotating along three axis, and the torques need to be the same for each axis, producing half integer torques is possible. If the torques were independent of each other, producing half-integer torques will affect a torque along a different axis. In this case, we assume only integer torques could be produced.

components is given by:

$$\eta_{\tau^{x_{1,2,3}}} = (\eta_{\tau^{x_1}} - 1)(\eta_{\tau^{x_2}} - 1)(\eta_{\tau^{x_3}} - 1) + 1 = \left(\frac{2(a^3 - a)}{3}\right)^3 + 1, \tag{4.33}$$

where the cases removed correspond to situations where at least one of the torque components is zero, leading to the the cases with one or two non-zero components.

The aforementioned analysis gives us the control solution space for both translation (forces) and rotation (torques) movements for an MHP robot and its relationship with robot size. Having knowledge of the solution space benefits the development of control solutions, where specific requirements, such as minimal energy consumption or movement efficiency are required. In addition, having full knowledge of the solution space facilitates not only the development of the control solutions presented in this work, but also future solutions.

4.3 Problem formulation

In the problem we aim to solve, the robot is required to reach the goal with a given orientation. In this section, we formally define the problem. We describe the robot's properties and characteristics, as well as the environment.

Consider a liquid and unbound environment $\mathcal{E} = \mathbb{R}^N$, $N \in \{2,3\}$ of density ρ, which contains a static goal at $\mathbf{g} \in \mathcal{E}$ and a robot, but is otherwise obstacle-free. The robot's body has a rigid shape, $\mathcal{A} \subset \mathbb{R}^N$. Let $h_t(\mathbf{x}) : \mathbb{R}^N \to \mathcal{E}$ denote a distance-preserving transformation from the robot's local reference frame to the global reference frame at time t. The robot then occupies

$$h_t(\mathcal{A}) = \{h_t(\mathbf{x}) \in \mathcal{E} | \mathbf{x} \in \mathcal{A}\}. \tag{4.34}$$

The robot is made of *modules* that are N-dimensional hyper-cubes of density ρ and unit side length, that is, the modules are neutrally buoyant. Physical connections between modules can be formed in each of the module's $2N$ faces. This allows for robots of different shapes (i.e. N-dimensional lattice configurations) to be built. In the following, we assume the robot's shape to be a hyper-rectangle, that is,

$$\mathcal{A} = [-a_1/2, a_1/2] \times \cdots \times [-a_N/2, a_N/2] \subset \mathbb{R}^N, \tag{4.35}$$

where $a_i \in \mathbb{Z}_+$. In these configurations, the robot comprises $n = \prod_{i=1}^{N} a_i$ modules.

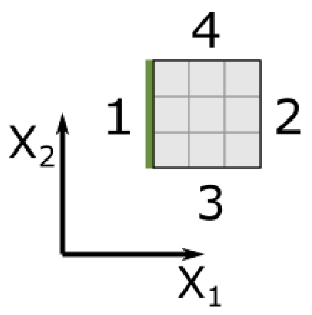


Fig. 4.6 A 3×3 robot, exemplifying the rules to determine the face number in an MHP robot.

We assume that all modules of a robot are oriented in a consistent way, thereby having a common sense of orientation within the robot's own reference frame. Faces $2i-1$ and $2i$, $i = 1, \ldots, N$, have their outward normal vectors anti-parallel and parallel to axis X_i^{local}, respectively (see Figure 4.6).

Remark. *In the previous section, we used two indexes to specify pumps and sensors of a given module, in a given face. However, fro the remainder of this chapter, denoting the module to which a pump or sensor belongs to is unnecessary, as every module uses an identical controller. As such, we will only use the face index, and all definitions of sensors and actuators are common to all modules.*

Each module contains one binary contact sensor per face, which detects whether another module is attached to that face:

$$c_j = \begin{cases} 1, & \text{face } j \text{ is connected with another module;} \\ 0, & \text{otherwise.} \end{cases} \tag{4.36}$$

Modules with $\exists j \in \{1, \ldots, 2N\} : c_j = 0$ are referred to as *boundary* modules. All other modules are referred to as *interior* modules.

Each module contains one binary goal sensor per face. The sensor detects whether the goal is visible, that is, not occluded by the robot. It is mounted in the face center. Let $\mathbf{s}_j(t) \in \mathcal{E}$ denote the position of goal sensor j at time t. Then,

$$d_j = \begin{cases} 1, & h_t(\mathcal{A}) \cap \{\alpha \mathbf{s}_j(t) + (1-\alpha)\mathbf{g} | \alpha \in [0,1)\} = \varnothing; \\ 0, & \text{otherwise.} \end{cases} \tag{4.37}$$

We assume a single goal in the environment, and that the robot cannot detect the goal from faces with symmetrical normals simultaneously.

All modules of a given robot share M binary power lines, that allow information to be shared between modules. Each power line transmits a single bit of information. The state of the k^{th} power line, b_k, is

$$b_k = \begin{cases} 1, & \text{at least one module activates line } k; \\ 0, & \text{otherwise.} \end{cases} \tag{4.38}$$

In other words, $b_k = 0$ if and only if no module actives power line k.

The problem is formulated with an MHP robot in mind. As such, each module contains a reservoir containing the same liquid as the environment. The reservoir is connected to all of the module's faces. Connected modules form a liquid network. Each face of a module contains a binary pump. When turned on ($p_j = 1$) the pump routes liquid from the reservoir through the face into the adjacent module, if present, or the environment, otherwise. When turned off ($p_j = 0$) the liquid can freely move in either direction.

4.3.1 Objective

We assume that all modules are aware of a preferred orientation, $O \in \{1, \ldots, 2N\}$, with respect to the robot's local reference frame, and that $O = 1$.[6] The objective of the robot is to reach the goal, in finite time, with orientation O. In particular, the face of the robot that is oriented towards O (hereafter referred to as the preferred face) needs to make the initial physical contact with the goal. Formally, let $\mathcal{A}_O$ denote the set of points of the modules' external faces that correspond to preferred orientation O (preferred face), $\mathcal{A}_O = \{-a_1/2\} \times [-a_2/2, a_2/2] \times \cdots \times [-a_N/2, a_N/2] \subset \mathcal{A}$. The condition that satisfies the objective is given by:

$$\exists T : (\forall t < T : h_t^{-1}(\mathbf{g}) \notin \mathcal{A}) \wedge (h_T^{-1}(\mathbf{g}) \in \mathcal{A}_O). \tag{4.39}$$

We assume that at time $t = 0$, $|h_0(\mathbf{0}) - \mathbf{g}| > \sqrt{(\frac{a_1}{2})^2 + \cdots + (\frac{a_N}{2})^2}$. In other words, in the initial position, the robot can freely rotate without touching the goal.

[6]The controllers can be adapted to any preference, or, alternatively, be generalized, if preference O is provided as input.

4.4 Controller solutions

The control of the pose of a robot can be divided into two components: position control and orientation control [116]. In the previous chapter, we used occlusion based controllers to move the robot towards a goal; the movement of the robot was dependent on which faces did not detect the goal, rather than the ones that did. When controlling the position component of the robot's pose, in the proposed solutions, we use a similar approach. On the other hand, for the control of the orientation component of the robot's pose, the control is based on the faces that are able to detect the goal.

When possible, we separate the components of the pose control strategies so that they occur sequentially, rather than simultaneously. This allows the robot to first reach the desired orientation, and then translate towards the desired position. By separating the rotation and translation movements, the only requirement for the robot to complete the task is that the robot is able to rotate without touching the goal, in its initial position—the initial distance condition.

The controller solutions we propose are fully decentralised, with every module using an identical controller. Each module sets the state of its pumps according to a control policy, $g_p(\mathbf{c}, \mathbf{d}, \mathbf{b})$. This control policy is dependent on the state of the shared power lines. Each shared power line acts as both an input and output, with each module reading it to determining its actuation states, and/or changing the power line state in accordance with its activation policy, $g_{b,k}(\mathbf{c}, \mathbf{d})$.[7] Both policies are reactive—they do not need run-time memory. While a module can take into account its local connectivity ($\mathbf{c}$), it is unaware of further positional information within the modular robot.

4.4.1 Pose control for robots moving in 2D

The controller solutions we propose, for a robot moving in 2D, allow a robot to control its pose in 3 degrees-of-freedom (DoFs): $\mathbf{X}_1$, $\mathbf{X}_2$ and θ. The differences between the controllers are based on different levels of information shared between modules. We present solutions for modules with access to two, one or no shared power lines. For a robot with no shared power lines we present two control variations, with advantages and disadvantages. For the remaining cases, a single solution is presented.

[7]Note that the activation policy does not take into account the state of actuators or of the power line itself. This is to prevent the power line from serving as a form of memory.

4.4.1.1 2-bits shared power line (`2D-2SP` controller)

We previously stated that the translation is occlusion based. For the robot to reach the goal with the desired orientation, all faces but the preferred should be occluded. As such, ideally, the robot needs to be correctly oriented before it can start translating. This means that, if the rotation and translation components of the movement are done separately, there are three different movements a robot should be able to perform: clockwise rotation, counter-clockwise rotation and translation. To differentiate between these three movements, two bits of information (i.e. $M = 2$), is the minimum amount of information required.

The activation policies of the shared power lines, $g_{b,1}(\mathbf{c}, \mathbf{d})$ and $g_{b,2}(\mathbf{c}, \mathbf{d})$, are given by:

$$b_1 = d_2 \vee d_3, \tag{4.40}$$

$$b_2 = d_4. \tag{4.41}$$

The control policy, $g_p(\mathbf{c}, \mathbf{d}, \mathbf{b})$, is given by

$$p_1 = b_1\bar{b}_2\bar{c}_1\bar{c}_3 \vee b_2\bar{c}_1\bar{c}_4, \tag{4.42}$$

$$p_2 = b_1\bar{b}_2\bar{c}_2\bar{c}_4 \vee b_2\bar{c}_2\bar{c}_3 \vee \bar{b}_1\bar{b}_2\bar{c}_2, \tag{4.43}$$

$$p_3 = b_1\bar{b}_2\bar{c}_3\bar{c}_2 \vee b_2\bar{c}_3\bar{c}_1, \tag{4.44}$$

$$p_4 = b_1\bar{b}_2\bar{c}_4\bar{c}_1 \vee b_2\bar{c}_4\bar{c}_2, \tag{4.45}$$

where p_j corresponds to the pump in face j. The first two terms of p_j are responsible for the rotation of the robot. For each face, one "corner" pump (e.g., $\bar{c}_1\bar{c}_3$) is active. Which corner pump should be active depends on the direction of rotation. As only corner modules activate pumps, and for two parallel faces, the pumps activated are symmetric, no undesired translation occurs. The last term of p_2 is responsible for the translation of the robot. As the modules have full information abut the preferred face, only modules belonging to face with normal anti-parallel to the preferred face's normal, need to activate their pumps. For convex robots, this occlusion-based strategy causes pure translation towards the goal [26, 109].

This means that as long as any face, other than the preferred face, detects the goal, the robot rotates, with the direction of rotation being determined by which faces detect the goal—counter-clockwise rotation for $b_1\bar{b}_2 = 1$, clockwise rotation for $b_2 = 1$, and translation for $\bar{b}_1\bar{b}_2 = 1$. This ensures that the robot undergoes the minimum rotation required to reach

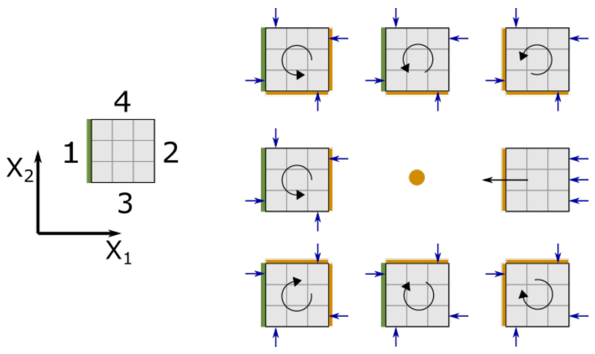


Fig. 4.7 Decentralised pose control with 2-bits of communication (`2D-2SP` controller). Shown are eight scenarios of a robot that needs to touch the goal (orange point) with its preferred face (face 1). The goal is visible from any module face marked with orange. The blue and black arrows correspond to the forces cased by the active actuators, and direction of the movement of the robot, for each scenario. If the robot is not correctly oriented, it rotates, with the direction of rotation dependent on the faces detecting the goal. Only when the robot is correctly oriented does it translate.

the preferred orientation. Figure 4.7 shows all the different possible sensing states that a robot can be placed in, and their respective actuation states, for this controller.

We state that two bits of information is the minimum required to produce the shortest rotation to the preferred orientation. However, by using additional information it might be possible to improve the controller. Table 4.2 compares the behaviours that can be obtained using 4-bit and 2-bit of information shared between the modules. The behaviours obtained, for the two cases are identical. This means that the addition of extra bits of shared information would only increases the complexity of the controller without changing its behaviour.

4.4.1.2 1-bit shared power line (`2D-1SP` controller)

While using two bits of information gives the best possible result—shorter rotation—it is still possible to ensure the task is solved when using one bit of information (i.e. $M = 1$). However, the robot is only able to rotate in one direction. As a result, the modules are able to determine that the robot is in the incorrect orientation, but not what is the rotation direction that ensures the minimum rotation.

The activation policy of the shared power line, $g_{b,1}(\mathbf{c},\mathbf{d})$, is given by:

$$b_1 = d_2 \vee d_3 \vee d_4. \tag{4.46}$$

Table 4.2 Behaviours obtained by using 4 and 2-bit of shared information in the controller. For the 4-bit case, the value of the shared power lines is given by $b_j = d_j$. For the 2-bit case, the value of the shared power lines is given by Equations (4.40) and (4.41).

4-bit shared power line					2-bit shared power line		
b_1	b_2	b_3	b_4	Behaviour	b_1	b_2	Behaviour
0	0	0	1	CW Rot	0	1	CW Rot
0	0	1	0	CCW Rot	1	0	CCW Rot
0	1	0	0	CCW Rot	1	0	CCW Rot
0	1	0	1	CW Rot	1	1	CW Rot
0	1	1	0	CCW Rot	1	0	CCW Rot
1	0	0	0	Trans	0	0	Trans
1	0	0	1	CW Rot	1	1	CW Rot
1	0	1	0	CCW Rot	0	0	CCW Rot

In other words, the power line is active if and only if the goal is perceived by a non-preferred face. The control policy, $g_p(\mathbf{c},\mathbf{d},\mathbf{b})$, is given by:

$$p_1 = b_1\bar{c}_1\bar{c}_3, \tag{4.47}$$

$$p_2 = b_1\bar{c}_2\bar{c}_4 \vee \bar{b}_1\bar{c}_2, \tag{4.48}$$

$$p_3 = b_1\bar{c}_3\bar{c}_2, \tag{4.49}$$

$$p_4 = b_1\bar{c}_4\bar{c}_1. \tag{4.50}$$

The first term of p_j is responsible for the rotation of the robot. The second term f p_2 is responsible for translation. In other words, if the power line is active ($b_1 = 1$), the robot rotates, otherwise it translates. Once more, one corner module per robot face activates its pump to generate the rotation.

The controller allows for only counter-clockwise rotation. As all corner modules contribute to the torque that creates the rotation, no undesired translation is produced. Similarly to the previous controller, during translation, only the modules that belong to the face opposite to the preferred face are active. Figure 4.8 shows all the different possible sensing states that a robot can be placed in, and their respective actuation states, for this controller. A symmetric version of this controller can be created, using clockwise rotation.

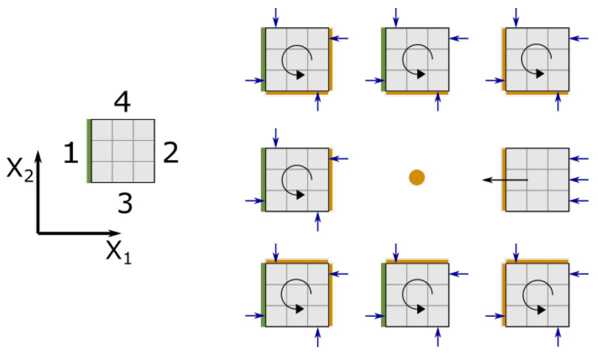


Fig. 4.8 Decentralised pose control with 1 bit of communication (`2D-2SP` controller). Shown are eight scenarios of a robot that needs to touch the goal (orange point) with its preferred face (face 1). The goal is visible from any module face marked with orange. The blue and black arrows correspond to the forces caused by the active actuators, and direction of the movement of the robot, for each scenario. If the robot is not correctly oriented, it rotates, always in the same direction. Only when the robot is correctly oriented does it translate.

4.4.1.3 No shared power line (`2D-0SP-V1` and `2D-0SP-V2` controllers)

While the previous controllers share small amounts of information between modules, some situations could exist where communication between the modules in any form is inviable. With no communication, each module only has local information. As a result, only some of the modules have information about the orientation of the robot. For this reason, it is not possible to preform rotation and translation separately. Therefore, the assumption that the robot can rotate without touching the goal—initial distance condition—is not enough to guarantee the success of these controllers. However, it is possible to obtain controllers that can complete the task, depending on a conjugation of the robots initial orientation and distance to the goal. We propose two solutions, where each module uses only the information it can obtain from the environment (i.e. $M = 0$).

The control policy $g_p(\mathbf{c},\mathbf{d})$ for the first controller solution (`2D-0SP-V1`) is given by:

$$p_1 = \bar{c}_1\bar{d}_1 \vee \bar{c}_1\bar{c}_3 d_3 \vee \bar{c}_1\bar{c}_4 d_4, \tag{4.51}$$

$$p_2 = \bar{c}_2\bar{d}_2 \vee \bar{c}_2\bar{c}_4\bar{d}_4, \tag{4.52}$$

$$p_3 = \bar{c}_3\bar{d}_3 \vee \bar{c}_3\bar{c}_2 d_3, \tag{4.53}$$

$$p_4 = \bar{c}_4\bar{d}_4 \qquad\qquad \vee \bar{c}_4\bar{c}_2 d_4. \tag{4.54}$$

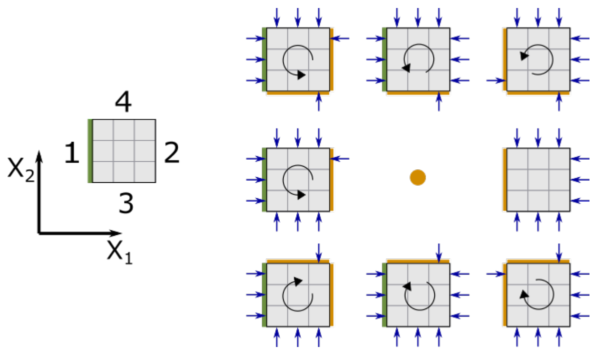


Fig. 4.9 Decentralized pose control without communication (`2D-OSP-V1` controller). Shown are eight scenarios of a robot that needs to touch the goal (orange point) with its preferred face (face 1). The goal is visible from any module face marked with orange. The pumps in all other faces are active, causing the robot to translate towards the goal. Up to two additional pumps are active, causing the robot to rotate.

The terms of the equations are aligned based on the behaviour that they generate in the robot, that is, the first column generates translation, the second column generates counter-clockwise rotation and the last column generates clockwise rotation.

The first term of p_j is $\bar{c}_j \bar{d}_j$. In other words, a pump gets activated if it belongs to an external module face ($\bar{c}_j$) from which the goal is not visible ($\bar{d}_j$), due being occluded by the robot itself. As this behaviour produces pure translation, if the robot is correctly aligned towards the goal (i.e. $d_2 = d_3 = d_4 = 0$), it only translates (see Figure 4.9). Otherwise, rotation is needed. As no communication is permitted, the modules contribute to rotation only if they have sufficient local information about the robot's orientation to guarantee that their actions are in agreement. For this reason, only modules at the corner of a configuration, that can detect the goal, contribute to rotation. Figure 4.9 depicts all eight possible robot orientations (defined by the sensors that are able to detect the goal), and shows for each one the subset of the corner modules that contribute to rotation.

The control policy $g_p(\mathbf{c}, \mathbf{d})$ for the second controller solution (`2D-OSP-V2`) is given by:

$$p_1 = \bar{c}_1 c_2 c_3 c_4 \bar{d}_1 \vee \bar{c}_1 \bar{c}_3 d_3 \quad \vee \bar{c}_1 \bar{c}_4 d_4, \tag{4.55}$$

$$p_2 = c_1 \bar{c}_2 c_3 c_4 \quad \vee \bar{c}_2 \bar{c}_4 d_2 \bar{d}_4, \tag{4.56}$$

$$p_3 = c_1 c_2 \bar{c}_3 c_4 \quad \vee \bar{c}_3 \bar{c}_2 d_3 \quad , \tag{4.57}$$

$$p_4 = c_1 c_2 c_3 \bar{c}_4 \qquad\qquad \vee \bar{c}_4 \bar{c}_2 d_4. \tag{4.58}$$

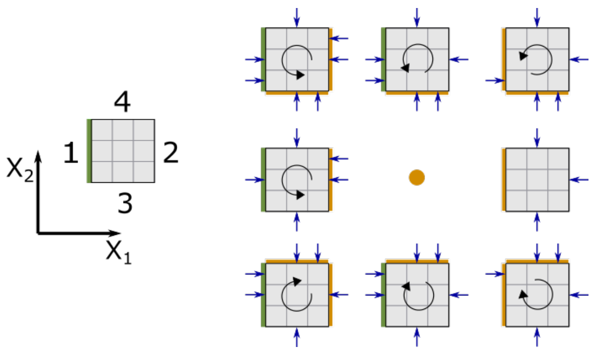


Fig. 4.10 Decentralized pose control without communication (`2D-0SP-V2` controller). Shown are eight scenarios of a robot that needs to touch the goal (orange point) with its preferred face (face 1). The goal is visible from any module face marked with orange. The pumps in all other faces are active, causing the robot to translate towards the goal. Up to two additional pumps are active, causing the robot to rotate.

As before, the terms of the equations are aligned based on the behaviour that they generate in the robot, in the same order.

As with the previous controller, the first term of each equation contributes to the robot's translation. However, for this controller, every face other than the preferred constantly contributes to the translation. As a result, the robot only actively translates if the preferred face detects the target (see Figure 4.10). Another difference, when compared with the `2D-0SP-V1` controller, is that the corners of each face only contribute to the rotation of the robot. In terms of rotation, this controller behaves in the same way as the previous—the rotation terms of Equations (4.55)–(4.58), are the same as of Equations (4.51)–(4.54).

Remark. *It is possible to obtain a controller variant of the* `2D-1SP` *controller, that allows both counter-clockwise and clockwise rotation, by bringing together the terms responsible for the rotation of the* `2D-0SP-V1` *controller and the term responsible for translation of the* `2D-1SP` *controller. For this variation, while there is some translation during rotation, the robot always moves away from the target. As a result, the initial distance condition is enough to guarantee that the task is successful. The control policy and formal analysis of this controller variant is presented in Appendix C.*

4.4.2 Pose control for robots moving in 3D

When changing from movement in 2D space into movement in 3D space, the complexity of the problem increases, due to the increased number of DoFs. The controller solutions we propose, for a robot moving in 3D, allow a robot to control its pose in 5 DoFs. With the increased number of DoFs, the robot will have to choose from multiple axes of rotation. As such, the amount of required information shared between modules increases, in order to complete the task. We propose solutions that use zero, two and five power lines. They are extension of the 2D controllers presented in the previous section.

In 3D, the controllers make use of the robot's edges (opposed to corners in 2D) to generate the rotations. The translation is done by the modules belonging to the robot's faces.

4.4.2.1 No shared power line (`3D-0SP-V1` and `3D-0SP-V1` controllers)

We extend the two controllers, proposed for 2D, that do not share information (i.e. $M = 0$). As in 2D, we are not able to separate translation and rotation, when using no shared information between modules. However, it is still possible to complete the task, depending on the initial position and orientation of the robot.

The control policy $g_p(\mathbf{c}, \mathbf{d})$ for the `3D-0SP-V1` is given by:

$$\begin{aligned}
p_1 &= \bar{c}_1\bar{d}_1 \lor \bar{c}_1\bar{c}_3 d_3 \lor \bar{c}_1\bar{c}_4 d_4 \lor \bar{c}_1\bar{c}_5 d_5 \lor \bar{c}_1\bar{c}_6 d_6, && (4.59)\\
p_2 &= \bar{c}_2\bar{d}_2 \lor \bar{c}_2\bar{c}_4\bar{d}_4, && (4.60)\\
p_3 &= \bar{c}_3\bar{d}_3 \lor \bar{c}_3\bar{c}_2 d_3, && (4.61)\\
p_4 &= \bar{c}_4\bar{d}_4 \qquad\qquad \lor \bar{c}_4\bar{c}_2 d_4, && (4.62)\\
p_5 &= \bar{c}_5\bar{d}_5 \qquad\qquad\qquad\qquad \lor \bar{c}_5\bar{c}_2 d_5, && (4.63)\\
p_6 &= \bar{c}_6\bar{d}_6 \qquad\qquad\qquad\qquad\qquad\qquad \lor \bar{c}_6\bar{c}_2 d_6. && (4.64)
\end{aligned}$$

The policy allows for clockwise and counter-clockwise rotation along two axes—X_2^{local} and X_3^{local}. Rotations along the two axes can occur simultaneously, effectively generating two further axes of rotation. For $d_5 = d_6 = 0$, the policy is identical to `2D-0SP-V1` [see Equations (4.51)–(4.54)]. As before, the alignment of the terms of the equations is based on their contribution for the robot's movement. The first column relates to translation, the second and third columns to counter-clockwise and clockwise rotation along X_3^{local}, respectively, and the last two columns to counter-clockwise and clockwise rotation along X_2^{local}, respectively.

The control policy $g_p(\mathbf{c},\mathbf{d})$ for the `3D-0SP-V2` is given by:

$$
\begin{aligned}
p_1 &= \bar{c}_1 c_2 c_3 c_4 c_5 c_6 \bar{d}_1 \vee \bar{c}_1 \bar{c}_3 d_3 \vee \bar{c}_1 \bar{c}_4 d_4 \vee \bar{c}_1 \bar{c}_5 d_5 \vee \bar{c}_1 \bar{c}_6 d_6, && (4.65)\\
p_2 &= c_1 \bar{c}_2 c_3 c_4 c_5 c_6 \vee \bar{c}_2 \bar{c}_4 \bar{d}_4, && (4.66)\\
p_3 &= c_1 c_2 \bar{c}_3 c_4 c_5 c_6 \vee \bar{c}_3 \bar{c}_2 d_3, && (4.67)\\
p_4 &= c_1 c_2 c_3 \bar{c}_4 c_5 c_6 \vee \bar{c}_4 \bar{c}_2 d_4, && (4.68)\\
p_5 &= c_1 c_2 c_3 c_4 \bar{c}_5 c_6 \vee \bar{c}_5 \bar{c}_2 d_5, && (4.69)\\
p_6 &= c_1 c_2 c_3 c_4 c_5 \bar{c}_6 \vee \bar{c}_6 \bar{c}_2 d_6. && (4.70)
\end{aligned}
$$

This policy controls the robot's rotation in the way as the `3D-0SP-V1` controller. However, translation does not occur until the preferred face detects the target. For $d_5 = d_6 = 0$, the policy is identical to `2D-0SP-V2` [see Equations (4.55)–(4.58)].

4.4.2.2 2-bits shared power line (`3D-2SP` controller)

Due to the multiple rotation axes, a single bit of shared information is not sufficient to guarantee the robot completes the task, with no simultaneous rotation and translation movements. Using two shared power lines (i.e. $M = 2$), it becomes possible to distinguish between the three main axis of rotation (X_1^{local}, X_2^{local} and X_3^{local}) and translation. The activation policies $g_{b,1}(\mathbf{c},\mathbf{d})$ and $g_{b,2}(\mathbf{c},\mathbf{d})$ are given by

$$
\begin{aligned}
b_1 &= d_2 \vee d_3 \vee d_4, && (4.71)\\
b_2 &= d_5 \vee d_6. && (4.72)
\end{aligned}
$$

Note that Equation (4.71) is the same as for `2D-1SP` [Equation (4.46)].

The control policy, $g_p(\mathbf{c},\mathbf{d},\mathbf{b})$ is given by

$$
\begin{aligned}
p_1 &= b_1 \bar{b}_2 \bar{c}_1 \bar{c}_3 \vee \bar{b}_1 b_2 \bar{c}_1 \bar{c}_6, && (4.73)\\
p_2 &= b_1 \bar{b}_2 \bar{c}_2 \bar{c}_4 \vee \bar{b}_1 b_2 \bar{c}_2 \bar{c}_5 \vee \bar{b}_1 \bar{b}_2 \bar{c}_2, && (4.74)\\
p_3 &= b_1 \bar{b}_2 \bar{c}_3 \bar{c}_2 \vee b_1 b_2 \bar{c}_3 \bar{c}_5, && (4.75)\\
p_4 &= b_1 \bar{b}_2 \bar{c}_4 \bar{c}_1 \vee b_1 b_2 \bar{c}_4 \bar{c}_6, && (4.76)\\
p_5 &= \bar{b}_1 b_2 \bar{c}_5 \bar{c}_1 \vee b_1 b_2 \bar{c}_5 \bar{c}_4, && (4.77)\\
p_6 &= \bar{b}_1 b_2 \bar{c}_6 \bar{c}_2 \vee b_1 b_2 \bar{c}_6 \bar{c}_3. && (4.78)
\end{aligned}
$$

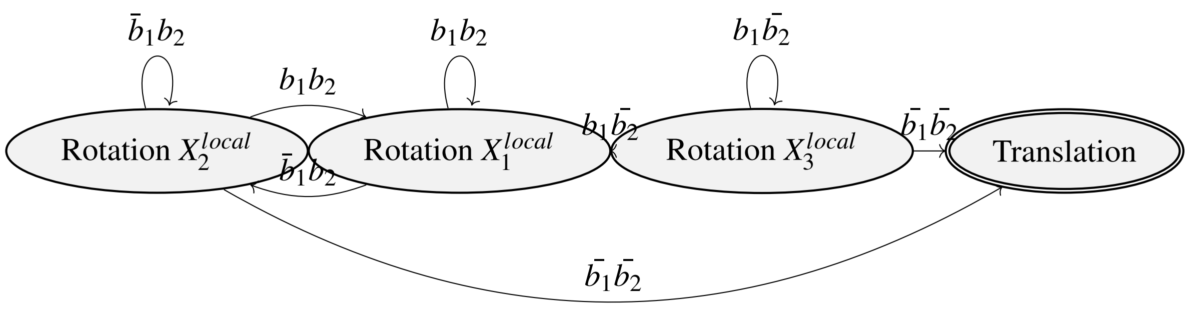


Fig. 4.11 Finite state machine representing the possible transitions between rotations for the `3D-2SP` controller. The state the robot starts in depends on its initial orientation. If it starts in one of the rotations, it will change states, based on the shared power lines values, until it reaches the desired orientation, at which point the robot translates towards the goal. If the robot starts in the preferred orientation, it only translates.

Similarly to the `2D-1SP` controller, this control policy allows only for counter-clockwise rotation. The terms in the first, second and third columns are responsible for rotations around X_3^{local}, X_2^{local} and X_1^{local}, respectively. The single term in the last column is responsible for the robot's translation. For $d_5 = d_6 = 0$ (implying $b_2 = 0$), the policy is identical to `2D-1SP` [see Equations (4.47)–(4.50)]. Different versions of this controller can be obtained by changing the directions of rotation along each of the axis. Including the version where the rotations along all axis are in counter-clockwise, a total of eight similar controllers exist.

Figure 4.11 shows a finite state machine where the transitions between rotations and translations are associated with the sensing status. If the goal is detected by a module's face, other than the preferred face, with normal parallel to X_1^{local} or X_2^{local} only ($b_1\bar{b}_2 = 1$), the robot rotates around X_3^{local}. If the goal is detected by a module's face with normal parallel to X_3^{local} only ($\bar{b}_1 b_2 = 1$), the robot rotates around X_2^{local}. If the goal is detected by faces, other than the preferred face, with normal parallel to X_2^{local} and X_3^{local} or X_1^{local} and X_3^{local}($\bar{b}_1\bar{b}_2 = 1$), the robot rotates around X_1^{local}.

Remark. *As in 2D, it is possible to obtain a controller, using a single shared power line (i.e. $M = 1$), that allows both counter-clockwise and clockwise rotation, around the X_3^{local} and X_2^{local} axis (similarly to* `3D-0SP-V1`*). This control policy uses components of the* `3D-0SP-V1` *and* `3D-2SP` *controllers. It is possible to prove that, while there is some translation during rotation, the robot always moves away from the target. This means we can prove the robot is guaranteed to complete the task, if the initial distance assumption is valid. The control policy and formal analysis of this controller variant is presented in Appendix C.*

4.4.2.3 5-bits shared power line (`3D-5SP-V1` and `3D-5SP-V2` controllers)

In 2D, the minimum amount of shared information required for the robot to perform the shortest rotation to reach the preferred orientation is two bits. When considering the third dimension, a minimum of two more bits of information are required to ensure the minimum rotation. However, if only the robot face with normal anti-parallel to the normal of the preferred face detects the goal, there are four possible directions of rotation that ensure the robot undergoes minimum rotation to reach the preferred orientation. To break this symmetric situation,we use one extra bit of information.[8] As such, this controller requires five shared power lines (i.e. $M = 5$). The activation policy, $g_{b,j}(\mathbf{c},\mathbf{d})$, is given by $b_j = d_j$, though without any change in behaviour, b_1 can be omitted.

We use two different approaches, leading to two different controllers. In the first approach (`3D-5SP-V1` controller), the robot rotates around the three coordinate axes, one axis at the time. In the second approach (`3D-5SP-V2` controller), the robot can rotate around multiple axis simultaneously, requiring only rotations around X_3^{local} and X_2^{local} to reach the preferred orientation.

The activation policy, $g_p(\mathbf{c},\mathbf{d},\mathbf{b})$, for the controller using the first approach (`3D-5SP-V1` controller) is given by:

$$\begin{aligned} p_1 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_4 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_3 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_3 \vee \\ & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_1 \bar{c}_5 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_1 \bar{c}_6 , \end{aligned} \tag{4.79}$$

$$\begin{aligned} p_2 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_3 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_4 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_4 \vee \\ & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_2 \bar{c}_6 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_2 \bar{c}_5 \vee \bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 , \end{aligned} \tag{4.80}$$

$$\begin{aligned} p_3 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_1 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_2 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_2 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_3 \bar{c}_6 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_3 \bar{c}_6 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_3 \bar{c}_6 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_3 \bar{c}_6 , \end{aligned} \tag{4.81}$$

$$\begin{aligned} p_4 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_2 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_1 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_1 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_4 \bar{c}_5 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_4 \bar{c}_5 \vee \bar{b}_4 \bar{b}_5 b_6 \bar{c}_4 \bar{c}_1 b_5 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_4 \bar{c}_5 , \end{aligned} \tag{4.82}$$

$$\begin{aligned} p_5 = {} & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_2 \vee \bar{b}_3 \bar{b}_5 \bar{b}_4 b_6 \bar{c}_5 \bar{c}_1 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_3 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_3 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_5 \bar{c}_3 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_5 \bar{c}_3 , \end{aligned} \tag{4.83}$$

$$\begin{aligned} p_6 = {} & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_1 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_2 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_6 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_6 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_6 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_6 . \end{aligned} \tag{4.84}$$

[8]This will be studied after the introduction of the controllers and the generated behaviours.

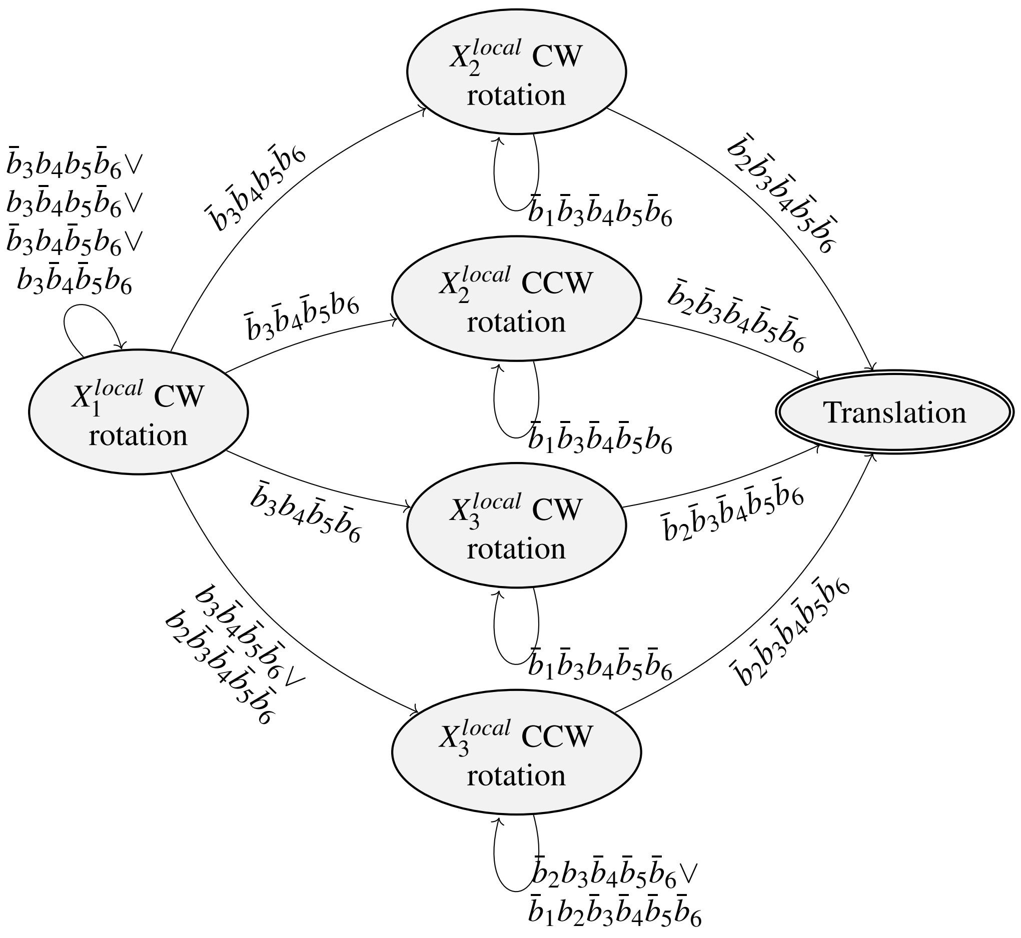

X_2^{local} CW rotation
X_2^{local} CCW rotation
X_1^{local} CW rotation
X_3^{local} CW rotation
X_3^{local} CCW rotation
Translation
$\bar{b}_3 b_4 b_5 \bar{b}_6 \vee$
$b_3 \bar{b}_4 b_5 \bar{b}_6 \vee$
$\bar{b}_3 b_4 \bar{b}_5 b_6 \vee$
$b_3 \bar{b}_4 \bar{b}_5 b_6$
$\bar{b}_3 \bar{b}_4 b_5 \bar{b}_6$
$\bar{b}_1 \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6$
$\bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_3 \bar{b}_4 \bar{b}_5 b_6$
$\bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_1 \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6$
$\bar{b}_3 b_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$
$b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \vee$
$b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_1 \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$
$\bar{b}_2 b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \vee$
$\bar{b}_1 b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6$

Fig. 4.12 Finite state machine representing the possible transitions between rotations for the `3D-5SPV1` controller. The state the robot starts in depends on its initial orientation. If it starts in one of the rotations, it will change states, based on the shared power lines values, until it reaches the desired orientation, at which point the robot translates towards the goal. If the robot starts in the preferred orientation, it only translates.

Due to the length of the control policy, each term is coloured, instead of aligned, based on its impact on the robot movement— clockwise around X_3^{local}, counter-clockwise around X_3^{local}, clockwise around X_2^{local}, counter-clockwise around X_2^{local}, clockwise around X_1^{local}, and translation[9]. The light green term corresponds to the case breaking the symmetry (counter-clockwise rotation around X_3^{local}). Each coloured term corresponds to one of the ten possible distinct sensing status of the robot, when using five shared power lines.

Figure 4.12 shows a finite state machine where the transactions between rotations and translations are associated with the sensing status. If the more than one power line is active

[9]The direction of rotation around X_1^{local} can be changed without change in performance.

(other than b_2), the robot rotates around X_1^{local}, in a clockwise manner. If a single power line (other than b_2) detects the goal, the robot performs the shortest rotation to reach the target, around X_3^{local} ($b_3\bar{b}_4 \vee \bar{b}_3 b_4 = 1$) or X_2^{local} ($b_5\bar{b}_6 \vee \bar{b}_5 b_6 = 1$). In the symmetric case, where $b_2\bar{b}_3\bar{b}_4\bar{b}_5\bar{b}_6 = 1$, the robot rotates around X_3^{local}, in a counter-clockwise manner. A symmetric version of this controller can be created, using a counter-clockwise rotation along X_1^{local}.

The activation policy, $g_p(\mathbf{c}, \mathbf{d}, \mathbf{b})$, for the controller using the second approach (`3D-5SP-V2` controller) is given by:

$$\begin{aligned} p_1 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_4 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_3 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_3 \vee \\ & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_1 \bar{c}_5 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_1 \bar{c}_6 \vee b_3 \bar{b}_4 b_5 \bar{b}_6 (\bar{c}_1 \bar{c}_3 \vee \bar{c}_1 \bar{c}_5) \vee \\ & \bar{b}_3 b_4 b_5 \bar{b}_6 (\bar{c}_1 \bar{c}_4 \vee \bar{c}_1 \bar{c}_5) \vee b_3 \bar{b}_4 \bar{b}_5 b_6 (\bar{c}_1 \bar{c}_3 \vee \bar{c}_1 \bar{c}_6) \vee \bar{b}_3 b_4 \bar{b}_5 b_6 (\bar{c}_1 \bar{c}_4 \vee \bar{c}_1 \bar{c}_6), \end{aligned} \tag{4.85}$$

$$\begin{aligned} p_2 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_3 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_4 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_4 \vee \\ & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_2 \bar{c}_6 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_2 \bar{c}_5 \vee b_3 \bar{b}_4 b_5 \bar{b}_6 (\bar{c}_2 \bar{c}_4 \vee \bar{c}_2 \bar{c}_6) \vee \\ & \bar{b}_3 b_4 b_5 \bar{b}_6 (\bar{c}_2 \bar{c}_3 \vee \bar{c}_2 \bar{c}_6) \vee b_3 \bar{b}_4 \bar{b}_5 b_6 (\bar{c}_2 \bar{c}_4 \vee \bar{c}_2 \bar{c}_5) \vee \bar{b}_3 b_4 \bar{b}_5 b_6 (\bar{c}_2 \bar{c}_3 \vee \bar{c}_2 \bar{c}_5) \vee \\ & \bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2, \end{aligned} \tag{4.86}$$

$$\begin{aligned} p_3 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_1 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_2 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_2 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_3 \bar{c}_2 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_3 \bar{c}_1 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_3 \bar{c}_2 \vee \\ & \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_3 \bar{c}_1, \end{aligned} \tag{4.87}$$

$$\begin{aligned} p_4 = {} & \bar{b}_3 b_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_2 \vee b_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_1 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_1 \vee \\ & b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_4 \bar{c}_1 \vee \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_4 \bar{c}_2 \vee \bar{b}_4 \bar{b}_5 b_6 \bar{c}_4 \bar{c}_1 b_3 \vee \\ & \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_4 \bar{c}_2, \end{aligned} \tag{4.88}$$

$$\begin{aligned} p_5 = {} & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_2 \vee \bar{b}_3 \bar{b}_5 \bar{b}_4 b_6 \bar{c}_5 \bar{c}_1 \vee b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_2 \vee \\ & \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_5 \bar{c}_2 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_5 \bar{c}_1 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_5 \bar{c}_1, \end{aligned} \tag{4.89}$$

$$\begin{aligned} p_6 = {} & \bar{b}_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_1 \vee \bar{b}_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_2 \vee b_3 \bar{b}_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_1 \vee \\ & \bar{b}_3 b_4 b_5 \bar{b}_6 \bar{c}_6 \bar{c}_1 \vee b_3 \bar{b}_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_2 \vee \bar{b}_3 b_4 \bar{b}_5 b_6 \bar{c}_6 \bar{c}_2. \end{aligned} \tag{4.90}$$

As with the previous controller, each term is coloured, based on its impact on the robot movement. The colour labelling is the same for the colours in common; the remaining colours are associated with: clockwise around both X_2^{local} and X_3^{local}, counter-clockwise around both X_2^{local} and X_3^{local}, clockwise around X_3^{local} and counter-clockwise around X_2^{local}, and counter-clockwise around X_3^{local} and clockwise around X_2^{local}. Given the assumptions that there is a single goal in the environment, and the robot cannot detect the goal from faces with symmetrical normals simultaneously ($b_1 b_2 = 0 \wedge b_3 b_4 = 0 \wedge b_5 b_6 = 0$), the control policy can be simplified to:

$$p_1 = \bar{b}_3 b_4 \bar{c}_1 \bar{c}_4 \vee b_3 \bar{b}_4 \bar{c}_1 \bar{c}_3 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_1 \bar{c}_3 \vee b_5 \bar{b}_6 \bar{c}_1 \bar{c}_5 \vee \bar{b}_5 b_6 \bar{c}_1 \bar{c}_6, \tag{4.91}$$

$$p_2 = \bar{b}_3 b_4 \bar{c}_2 \bar{c}_3 \vee b_3 \bar{b}_4 \bar{c}_2 \bar{c}_4 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2 \bar{c}_4 \vee b_5 \bar{b}_6 \bar{c}_2 \bar{c}_6 \vee \bar{b}_5 b_6 \bar{c}_2 \bar{c}_5 \vee \bar{b}_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_2, \tag{4.92}$$

$$p_3 = \bar{b}_3 b_4 \bar{c}_3 \bar{c}_1 \vee b_3 \bar{b}_4 \bar{c}_3 \bar{c}_2 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_3 \bar{c}_2, \tag{4.93}$$

$$p_4 = \bar{b}_3 b_4 \bar{c}_4 \bar{c}_2 \vee b_3 \bar{b}_4 \bar{c}_4 \bar{c}_1 \vee b_2 \bar{b}_3 \bar{b}_4 \bar{b}_5 \bar{b}_6 \bar{c}_4 \bar{c}_1, \tag{4.94}$$

$$p_5 = b_5 \bar{b}_6 \bar{c}_5 \bar{c}_2 \vee \bar{b}_5 b_6 \bar{c}_5 \bar{c}_1, \tag{4.95}$$

$$p_6 = b_5 \bar{b}_6 \bar{c}_6 \bar{c}_1 \vee \bar{b}_5 b_6 \bar{c}_6 \bar{c}_2, \tag{4.96}$$

using the same colour scheme as before. The rotations around two simultaneous rotation axis occur by combining simultaneous rotations around the X_2^{local} and X_3^{local} (e.g. both yellow and blue terms).

We previously stated that to perform the shortest rotation possible, a minimum of five bits of information is required. To rotate both clockwise and counter-clockwise along X_2^{local} and X_3^{local} a minimum of four bits of communication is required. This is a result of the rotations along each of the axis being independent, meaning a minimum of two bit is required for each axis to allow for no rotation, clockwise rotation and counter-clockwise rotation. However, if the goal is detected only by the face with outward normal parallel to X_1^{local}, all four possible rotations considered lead to the shortest rotation. In 2D, this situation is solved by combining the sensing states of two faces in a single shared power line, as there are only two possible rotations. However, in 3D combining it with the sensing of another face into a single shared power line generates a longer rotation than needed for another configuration.

Table 4.3 compares the behaviours generated by using four and five bits of information for the `3D-5SP-V2` controller. It can be seen that by combining the sensing status of the face with outward normal parallel to X_1^{local} with the sensing status of adjacent face changes the behaviour of the controller. Similar results would be obtained if a different adjacent face was used. For the`3D-5SP-V1` controller, the effect would be similar. However, instead of changing from rotating along a single axis to rotating along two axes simultaneously in the affected sensing states, the controller would just change the rotation axis[10]. As it is not

[10]This is shown in Appendix D.

Table 4.3 Behaviours obtained by using 5 and 4-bit of shared information in the `3D-5SP-V1` controller. For the 5-bit case, the value of the shared power lines is given by $b_j = d_j$. For the 5-bit case, the value of the shared power lines is given by $b_3 = d_2 \vee d_3$, $b_4 = d_4$, $b_5 = d_5$, $b_6 = d_6$.

5-bit shared power line						4-bit shared power line				
b_2	b_3	b_4	b_5	b_6	Behaviour	b_3	b_4	b_5	b_6	Behaviour
0	0	0	0	0	T	0	0	0	0	T
0	0	0	0	1	CCW X_2^{local}	0	0	0	1	CCW X_2^{local}
0	0	0	1	0	CW X_2^{local}	0	0	1	0	CW X_2^{local}
0	0	1	0	0	CW X_3^{local}	0	1	0	0	CW X_3^{local}
0	0	1	0	1	CW X_3^{local} CCW X_2^{local}	0	1	0	1	CW X_3^{local} CCW X_2^{local}
0	0	1	1	0	CW X_3^{local} CW X_2^{local}	0	1	1	0	CW X_3^{local} CW X_2^{local}
0	1	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
0	1	0	0	1	CCW X_3^{local} CCW X_2^{local}	1	0	0	1	CCW X_3^{local} CCW X_2^{local}
0	1	0	1	0	CCW X_3^{local} CW X_2^{local}	1	0	1	0	CCW X_3^{local} CW X_2^{local}
1	0	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
1	0	0	0	1	**CCW** X_2^{local}	1	0	0	1	**CCW** X_3^{local} **CCW** X_2^{local}
1	0	0	1	0	**CW** X_2^{local}	1	0	1	0	**CCW** X_3^{local} **CW** X_2^{local}
1	0	1	0	0	CW X_3^{local}	1	1	0	0	CW X_3^{local}
1	0	1	0	1	CW X_3^{local} CCW X_2^{local}	1	1	0	1	CW X_3^{local} CCW X_2^{local}
1	0	1	1	0	CW X_3^{local} CW X_2^{local}	1	1	1	0	CW X_3^{local} CW X_2^{local}
1	1	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
1	1	0	0	1	CCW X_3^{local} CCW X_2^{local}	1	0	0	1	CCW X_3^{local} CCW X_2^{local}
1	1	0	1	0	CCW X_3^{local} CW X_2^{local}	1	0	1	0	CCW X_3^{local} CW X_2^{local}

possible to obtain the best result with four bits of shared information, an extra shared power line is required. As with the 2D case, adding an extra shared power line, would bring no improvement to the controller, while increasing the complexity[10].

4.5 Formal analysis of the proposed controllers

In this section, we provide an analysis of the decentralised controllers proposed in Section 4.4, for convex robots. We make the following assumptions: (i) time t is continuous, (ii) instantaneous information is obtained from sensors ($\mathbf{c}$ and $\mathbf{d}$) and power lines ($\mathbf{b}$)[11], (iii) changes in pump ($\mathbf{p}$) and power line ($\mathbf{b}$) activation take immediate effect and (iv) the robot is subject to translational and rotational drag forces. It accelerates instantaneously, emulating quasi-static motion. It is thus either at rest or moving with terminal velocity.

[11]Sensing information corresponds to information each module can connect independently form the environment, while shared power lines inform each module of information collected by other modules.

Theorem 3. *A goal that has not been reached is detected by all goal sensors of at least one of the robot's faces.*

Proof. We assume that the opposite was true, that is, $\exists t : \mathbf{g} \notin h_t(\mathcal{A}) \wedge (\nexists j : \bar{c}_j \Rightarrow d_j)$. Consider the position of the goal at time t in the local coordinate system of the robot, $\mathbf{g}^{\text{local}} = h_t^{-1}(\mathbf{g}) \in \mathbb{R}^N$. It follows that $\mathbf{g}^{\text{local}} \in \mathbb{R}^N \setminus \mathcal{A} = \mathbb{R}^N \setminus [-a_1/2, a_1/2] \times \cdots \times [-a_N/2, a_N/2] = \cup_i \mathcal{H}_i$, where $\mathcal{H}_i = \{(x_1, \ldots, x_N) \in \mathbb{R}^N | (x_i < -a_i/2) \vee (a_i/2 < x_i)\}$. Therefore, $\exists i : \mathbf{g}^{\text{local}} \in \mathcal{H}_i$. There exists at least one module with goal sensor $2i-1$ at $\mathbf{s}_{2i-1}^{\text{local}}[i] = -a_i/2$ and with $\bar{c}_{2i-1} = 1$, and at least one module with goal sensor $2i$ at $\mathbf{s}_{2i}^{\text{local}}[i] = a_i/2$ and with $\bar{c}_{2i} = 1$. Hence, $\mathcal{A} \cap \{\alpha \mathbf{s}_{2i-1}^{\text{local}} + (1-\alpha)\mathbf{g}^{\text{local}} | \alpha \in [0,1)\} = \varnothing$ or $\mathcal{A} \cap \{\alpha \mathbf{s}_{2i}^{\text{local}} + (1-\alpha)\mathbf{g}^{\text{local}} | \alpha \in [0,1)\} = \varnothing$ must hold true. As h_t is bijective, it follows from Equation (4.37) that either $d_{2i-1} = 1$ or $d_{2i} = 1$, which violates our assumption. ■

4.5.1 Pose control in 2D environments

Consider the controllers presented in Section 4.4.1. When executed on an interior module ($\forall j : c_j = 1$), it follows from the control policies of each controller [Equations (4.42)–(4.45), Equations (4.47)–(4.50), Equations (4.51)–(4.54) and Equations (4.55)–(4.58)] that no pump is activated, that is, $p_1 = p_2 = p_3 = p_4 = 0$; fluid can freely flow in any direction through the module. When executed on a boundary module, pumps on faces connected to other modules are not activated either. Therefore, the following analysis focuses on pumps on the robot's external faces. The robot has $2(a_1 + a_2)$ such pumps.

4.5.1.1 Controllers with shared power lines: `2D-1SP` and `2D-2SP` controllers

We start by performing the analysis for the `2D-2SP` and `2D-1SP` controllers. As the translation and rotation do not occur simultaneously, the analysis of the controller becomes simpler. To prove the controllers complete the task, we need to prove that when translating the robot reaches the target, and when rotating it reaches the desired orientation, in finite time.

Lemma 2. *An $a_1 \times a_2$ robot, using the* `2D-1SP` *controller, that faces the goal with only the preferred face, that is, $\bigvee_{j \neq 1} d_j = 0$, completes the task in finite time.*

Proof. From $\bigvee_{j \neq 1} d_j = 0$ and Equation (4.46), it follows that $b_1 = 0$. Using Equations (4.47)–(4.50), we obtain $p_1 = p_3 = p_4 = 0$ and $p_2 = \bar{c}_2$. Hence all pumps belonging to a face with normal parallel to X_1^{local} are active Each active pump produces a constant thrust force, $f_p > 0$,

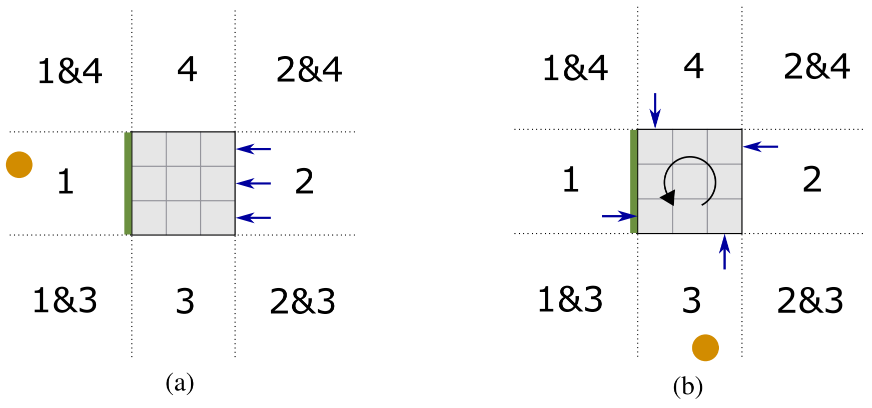


Fig. 4.13 Illustration of a modular robot that is tasked to approach a goal (orange point) with a preferred orientation (green face, using the 2D-1SP controller. In (a), the robot is correctly oriented, and only needs to translate towards the goal. It does so by activating the pumps (blue arrows) on the face that is opposite to the preferred face. In (b), the robot rotates counter-clockwise, by activating four pumps (blue arrows) of its corner modules, until the preferred orientation is reached.

which is anti-parallel to the preferred face's normal. As all active pumps (and hence forces) are arranged symmetrically, no torque is produced. The robot undergoes a pure translation towards the goal (see Figure 4.13a.) The direction of movement is parallel to the two lines that define, respectively, the half-plane sensing regions of goal sensors 3 and 4. As a consequence, $\bigvee_{j\neq 1} d_j = 0$ as long as the robot does not touch the goal. The robot has a_2 active pumps giving a net force of $a_2 f_p$ along the preferred face's normal. This is a positive constant and, from the quasi-static motion assumption, produces a positive constant velocity. As the distance between the robot and goal is finite and decreasing at a constant rate, the goal is reached in finite time. ■

Lemma 2 is also valid for the 2D-2SP controller, as the force acting on the robot when in the preferred orientation ($\bigvee_{j\neq 1} d_j = 0$) is the same; using Equations (4.40)–(4.41), the control policy for the 2D-2SP controller [Equations (4.42)–(4.45)] has the same output.

Theorem 4. *An $a_1 \times a_2$ robot (with $a_i \geq 2$ for some i), using the* 2D-1SP *controller, completes the task in finite time.*

Proof. According to Lemma 2, we only need to consider the case $\bigvee_{j\neq 1} d_j = 1$. From Equation (4.46), it follows that $b_1 = 1$. As the second term ($\bar{b}_1\bar{c}_2$) in Equation (4.48) disappears, Equations (4.47)–(4.50) assume a symmetric form, causing the corner modules of the rectangular configuration to activate four pumps in total (see Figure 4.13b). As the

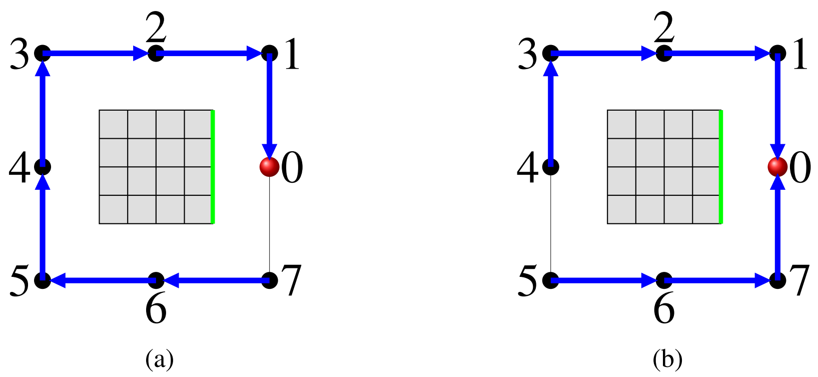


Fig. 4.14 Visualization of the graph representing the 8 global sensing states (nodes) of a convex-shaped 2D robot with respect to the goal. The robot is represented by the grey squares, with the green line representing the preferred face. Each node represents a position of the goal with respect to the robot (corner point, face without edges) with its name corresponding to the robot's feature that contains the closest point to the goal. Transitions between nodes, that is movements of the goal, with respect to the robot (directed edges) are those chosen by the (a) decentralised `2D-1SP` and (b) decentralised `2D-2SP` controllers, which have partial and full information about the global state, respectively. For any given node, the controller follows a unique path that is guaranteed to terminate with the goal in front of the preferred face (node 0) after at most seven (a) and four (b) transitions.

activated pumps face in opposing directions, no translation occurs. The geometric center of the robot coincides with the center of mass. The pumps firing with a direction parallel or anti-parallel to X_1 produce a torque of $2(\frac{a_2}{2} - \frac{1}{2})f_p = (a_2 - 1)f_p$, whereas the pumps firing with a direction parallel or anti-parallel to X_2 produce a torque of $(a_1 - 1)f_p$. The combined torque is $(a_1 + a_2 - 2)f_p \geq f_p$, which is positive constant. As the moment of inertia of the robot is constant, the torque results in a positive, constant angular velocity (assuming quasi-static motion). The robot hence rotates counter-clockwise until $\bigvee_{j\neq 1} d_j = 0$, in which case Lemma 2 applies. Figure 4.14a shows the direction of rotation of the robot, based on the goal position. The time to rotate is bounded by the time for a full revolution; the latter is constant, for a given robot size, as the angular velocity is constant. ■

Theorem 5. *An $a_1 \times a_2$ robot (with $a_i \geq 2$ for some i), using the* `2D-2SP` *controller, completes the task in finite time.*

Proof. As with Theorem 4, we only need to consider the case $\bigvee_{j\neq 1} d_j = 1$. From Equations (4.40) and (4.41), it follows that either $b_1 = 1 \vee b_2 = 1$. As the second term $(\bar{b}_1\bar{c}_2)$ in Equation (4.43) disappears, Equations (4.42)–(4.45) assume a symmetric form, causing the corner modules of the rectangular configuration to activate four pumps in total (see

Table 4.4 Forces and torques generated for each distinct sensing status. Each sensing status corresponds to a node on the graphs shown in Figure 4.14b. Positive and negative forces acting on the robot move it towards and away form the goal, respectively. Positive and negative torques correspond to counter-clockwise and clockwise rotation, respectively.

Node	`2D-OSP-V1` controller		`2D-OSP-V2` controller	
	f	τ	**f**	τ
0	$(0,a_2)f_p$	0	$(0,a_2-2)f_p$	0
1	$(a_1-1,a_2-1)f_p$	$\frac{a_1+a_2-2}{2}f_p$	$(a_2-3,-1)f_p$	$\frac{a_1+a_2-2}{2}f_p$
2	$(a_1-1,0)f_p$	$\frac{a_1-1}{2}f_p$	$(-1,-1)f_p$	$\frac{a_1+a_2-2}{2}f_p$
3	$(a_1-1,a_2-1)f_p$	$\frac{a_1+a_2-2}{2}f_p$	$(0,-1)f_p$	$\frac{a_1+2a_2-3}{2}f_p$
4	$(0,a_2-1)f_p$	$\frac{a_2-1}{2}f_p$	$(-1,0)f_p$	$\frac{a_2-1}{2}f_p$
5	$(a_1-1,a_2-1)f_p$	$-\frac{a_1-1}{2}f_p$	$(0,-1)f_p$	$-\frac{a_1+a_2-2}{2}f_p$
6	$(a_1-1,0)f_p$	$-\frac{a_1-1}{2}f_p$	$(-1,-1)f_p$	$-\frac{a_1+a_2-2}{2}f_p$
7	$(a_1-1,a_2-1)f_p$	$-\frac{a_1+a_2-2}{2}f_p$	$(-1,a_2-3)f_p$	$-\frac{a_1+a_2-2}{2}f_p$

Figure 4.7). As the activated pumps face in opposing directions, no translation occurs. The geometric center of the robot coincides with the center of mass. The pumps firing produce a torque of $(a_1+a_2-2)f_p \geq f_p$ or $-(a_1+a_2-2)f_p \leq -f_p$, which is non-zero constant. As the moment of inertia of the robot is constant, the torque results in a non-zero, constant angular velocity (assuming quasi-static motion). The robot rotates either counter-clockwise or clockwise, until $\bigvee_{j\neq 1} d_j = 0$, in which case Lemma 2 applies. Figure 4.14b shows the robot's direction of rotation, based on the robot's sensing state. The rotation time is bounded by the time of half revolution; the latter is constant, for a given robot size, as the angular velocity is constant. ■

Corollary 1. *A* 1×1 *robot is not guaranteed to complete the task in finite time.*

Proof. As can be seen from the proof of Theorems 4 and 5, a single module is unable to rotate, as the torque is $a_1+a_2-2=0$. The reason for this is the rotation and reflection symmetry of the module, where each pump's force and position vectors are anti-parallel to each other. ■

4.5.1.2 Controllers without shared power lines: `2D-OSP-V1` and `2D-OSP-V2` controllers

Both versions of the controller without shared power lines travel the graph shown in Figure 4.14b. From Theorem 5, we know that, as long as the torque acting on the robot is non-zero, the robot will rotate until reaching the desired orientation, in finite time. From

Lemma 2, once the desired orientation is reached, if the force acting on the robot is a positive constant and no torque is applied, it will reach the goal in finite time. Table 4.4 lists the forces and torques acting on the robot for the two controllers that use no communication, for the different sensing status. We can see that, for the `2D-OSP-V1` controller, as long as $a_k, a_l \geq 2$, for some $k \neq l$, the torque is non zero, unless in the preferred orientation (node zero). The same is true for the `2D-OSP-V2`, when $a_k \geq 2 \wedge a_l > 2$, for some $k \neq l$. When in the preferred orientation, both controllers cause no rotation (torque is zero) and apply a constant positive force to act on the robot. However, since rotation and translation occur simultaneously for these controllers, these two components being bound in terms of time is not sufficient to guarantee completion of the task. If the distance between the goal and the target is not large enough, so that the robot has time to reach the desired orientation before reaching the target, the robot is not able to complete the task using these controllers. As the behaviours of the controllers are different, the distances required for ensuring task completion are different.

The forces acting on the robot are considered positive if the resulting translation moves the robot towards the goal. Negative forces move the robot away from the goal. The `2D-OSP-V1` controller causes a positive force to act on the robot, regardless of the orientation. To successfully complete the task, the robot's translation time needs to be equal or larger than the rotation time.

To obtain a bound for the minimum distance required to complete the task successfully, we assume that the forces and torques acting on the robot are small and the robot moves in a highly viscous fluid environment. In these environments, the drag forces acting on the robot are proportional to the velocity. For this reason, the robot's dynamics can be described by:

$$\mathbf{v} = k\mathbf{f} \tag{4.97}$$

$$\omega = k_\theta \tau, \tag{4.98}$$

where $\mathbf{v}$ and ω are the robot's linear and angular velocities, $\mathbf{f}$ and τ are the force and torque acting on the robot and k and k_θ positive constants. Let Δd be the initial distance between the robot's closest point to the goal, and the goal and $\Delta\theta$ be the angle offset form the preferred orientation, respectively. We have:

$$t_{trans} = \frac{\Delta d}{v} \tag{4.99}$$

$$t_{rot} = \frac{\Delta\theta}{\omega}, \tag{4.100}$$

where t_{trans} and t_{rot} are the time it takes for the robot to reach the goal and desired orientation, respectively. As the robot needs to reach the desired orientation before reaching the goal,

$$t_{trans} \geq t_{rot} \Leftrightarrow \frac{\Delta d}{v} \geq \frac{\Delta\theta}{\omega} \Leftrightarrow \Delta d \geq \frac{\Delta\theta v}{\omega}. \tag{4.101}$$

Replacing v and ω for Equations (4.97) and (4.98), we have

$$\Delta d \geq \frac{\Delta\theta k f}{k_\theta \tau}. \tag{4.102}$$

The maximum angle offset for this controller is $\Delta\theta = \pi$. From Table 4.4, we know that $f \leq f_p\sqrt{(a_1-1)^2+(a_2-1)^2} \leq f_p\sqrt{a_1^2+a_2^2}$ and $\tau \geq \frac{a_2-1}{2}f_p$. Replacing in Equation (4.102),

$$\Delta d_1 \geq 2\pi k_r \frac{\sqrt{a_1^2+a_2^2}}{a_2-1}, \text{with } k_r = \frac{k}{k_\theta}. \tag{4.103}$$

If the condition obtained in Equation (4.103) is not met, the success of the controller is not guaranteed, and the robot can reach the goal with the incorrect orientation.

The `2D-OSP-V2` controller causes a negative force to act on the robot if the preferred face cannot detect the goal. This occurs because, all the modules with a single external face, belonging to the all the faces except the preferred, activate their pumps regardless of their sensing state, while the preferred face runs the occlusion based controller. In this situation, the forces that cause the rotation of the robot, also cause a translation away from the goal. When the preferred face detects the goal, the robot starts translating towards the goal.

Using the same dynamics model as before we can determine the necessary distance between the robot and the goal, to ensure the success of the controller, when the robot starts translating. This corresponds to the robot being in situation represented by nodes 1 and 7 in Table 4.4. The force and torque acting on the robot in those situations are given by $f = f_p\sqrt{(a_2-3)^2+(-1)^2} \leq f_p\sqrt{a_2^2+1}$ and $\tau = \frac{a_1+a_2-2}{2}f_p$ (from Table 4.4). The angle offset for these nodes is $\theta < \frac{\pi}{2}$. The distance condition becomes:

$$\Delta d_2 \geq \pi k_r \frac{\sqrt{a_2^2+1}}{(a_1+a_2-2)}, \text{with } k_r = \frac{k}{k_\theta}. \tag{4.104}$$

If the condition obtained in Equation (4.104) is not met, the success of the controller is not guaranteed, and the robot can orbit around the goal without ever reaching it[12].

Due to starting the active translation only after being closer to the desired orientation, the minimum distance to guarantee the success of the `2D-OSP-V2` controller is smaller than for the `2D-OSP-V1` controller.

4.5.2 Pose control in 3D environments

Consider the controllers presented in Section 4.4.2. As for the 2D pose control analysis, only pumps in the robot's external faces need to be considered. The robot has $2(a_1a_2 + a_1a_3 + a_2a_3)$ such pumps.

4.5.2.1 Controllers with shared power lines: `3D-2SP`, `3D-5SP-V1` and `3D-5SP-V2` controllers

As for the 2D analysis, we start by performing the analysis of the controllers that require the used of shared power lines. As the translation and rotation do not occur simultaneously, the analysis of the controller becomes simpler. We can prove the controllers complete the task if we prove that when translating the robot reaches the target, and when rotating it reaches the desired orientation, in finite time.

Lemma 3. *An $a_1 \times a_2 \times a_3$ robot that faces the goal with only the preferred face, that is, $\bigvee_{i \neq 1} d_i = 0$, completes the task in finite time, independently of the controller used.*

Proof. The proof is omitted, as a straight-forward extension of the proof of Lemma 2. ■

Theorem 6. *An $a_1 \times a_2 \times a_3$ robot (with $a_k, a_l \geq 2$ for some $k \neq l$), using the* `3D-2SP` *controller, completes the task in finite time.*

Proof. According to Lemma 3, we need only to consider the case $\bigvee_{j \neq 1} d_j = 1$. From Equations (4.71) and (4.72), it follows that $b_1 \vee b_2 = 1$. As such, the third term of Equation (4.74) disappears. Depending on the values of b_1 and b_2, two of Equations (4.73)–(4.78) return 0, whereas the remaining four equations assume a symmetric form, causing the modules belonging to the robot's edges that are parallel to X_i^{local} to activate $4a_i$ pumps in total. This

[12]More details about this behaviour will be discussed in Section 5.2.3.3

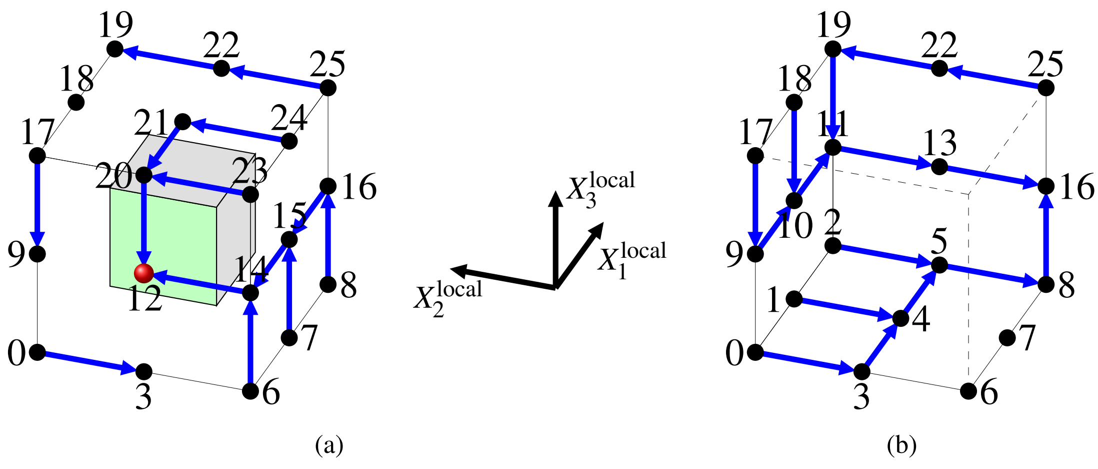


Fig. 4.15 Visualization of the graph representing the 26 global sensing states (nodes) of a convex-shaped 3D robot with respect to the goal. The robot is represented by the grey cube, with the green face representing the preferred face. Each node represents a position of the goal with respect to the robot (corner point, edge without corner points, face without edges) with its name corresponding to the robot's feature that contains the closest point to the goal. Transitions between nodes, that is movements of the goal, with respect to the robot(directed edges) are those chosen by the decentralised `3D-2SP` controller, which has only partial information about the global state. For any given node, the controller follows a unique path that is guaranteed to terminate in the preferred state (node 12) after at most eight transitions. In (a) the front, right and top of the graph are represented. In (b) the back, left and bottom of the graph are represented.

generates a 2D rotation around axis X_i^{local}. By applying the analysis for Theorem 4, one can show that a new situation, with different values b_1 or b_2, is reached in finite time. The robot can be in any of 26 global sensing states [13], which are depicted in Figure 4.15. The robot transitions between these states by using 2D rotations along either of the robot's reference frame axes. Figure 4.15 shows the graph of transitions realized by the controller. The graph is acyclic. Moreover, from any starting orientation (i.e. global sensing state) there is a unique path to reach the desired orientation, which corresponds to global sensing state 12. As any path can be completed by at most eight steps (i.e. 2D rotations), the overall task is completed in finite time. ■

Theorem 7. *An $a_1 \times a_2 \times a_3$ robot (with $a_k, a_l \geq 2$ for some $k \neq l$), using the* `3D-5SP-V1` *controller, completes the task in finite time.*

[13] We refer to these states as global, as the modules do not have access to complete state information.

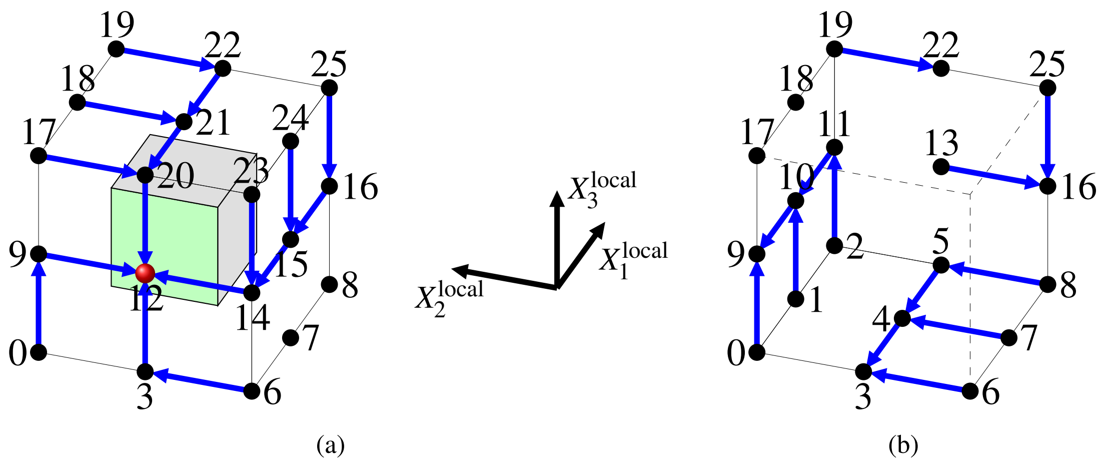


Fig. 4.16 Visualization of the graph representing the 26 global sensing states (nodes) of a convex-shaped 3D robot with respect to the goal. The robot is represented by the grey cube, with the green face representing the preferred face. Each node represents a position of the goal with respect to the robot (corner point, edge without corner points, face without edges) with its name corresponding to the robot's feature that contains the closest point to the goal. Transitions between nodes, that is movements of the goal, with respect to the robot (directed edges) are those chosen by the decentralised `3D-5SP-V1` controller, which has full information about the global state. For any given node, the controller follows a unique path that is guaranteed to terminate in the preferred state (node 12) after at most four transitions. In (a) the front, right and top of the graph are represented. In (b) the back, left and bottom of the graph are represented.

Proof. As before, we need to consider only the case $\bigvee_{j\neq 1} d_j = 1$. From the definition of b_j, it follows that $b_2 \vee b_3 \vee b_4 \vee b_5 \vee b_6 = 1$. As such, the last term of Equation (4.80) disappears. Depending on the values of the power lines, two of Equations (4.79)–(4.84) return 0, whereas the remaining four equations assume a symmetric form, causing the modules belonging to the robot's edges that are parallel to an axis, X_i^{local}, to activate $4a_i$ pumps in total. This generates a 2D rotation around axis X_i^{local} (direction of rotation depend on the power lines active). By applying the analysis of Theorem 5, one can show that a new situation, with different values for the power lines, is reached in finite time. Figure 4.16 shows the graph of transitions, between the 26 possible global sensing states, realized by the controller. The graph is acyclic. Moreover, from any starting orientation (i.e. global sensing state) there is a unique path to reach the desired orientation, which corresponds to global sensing state 12. As any path can be completed by at most four steps (i.e. 2D rotations), the overall task is completed in finite time. ∎

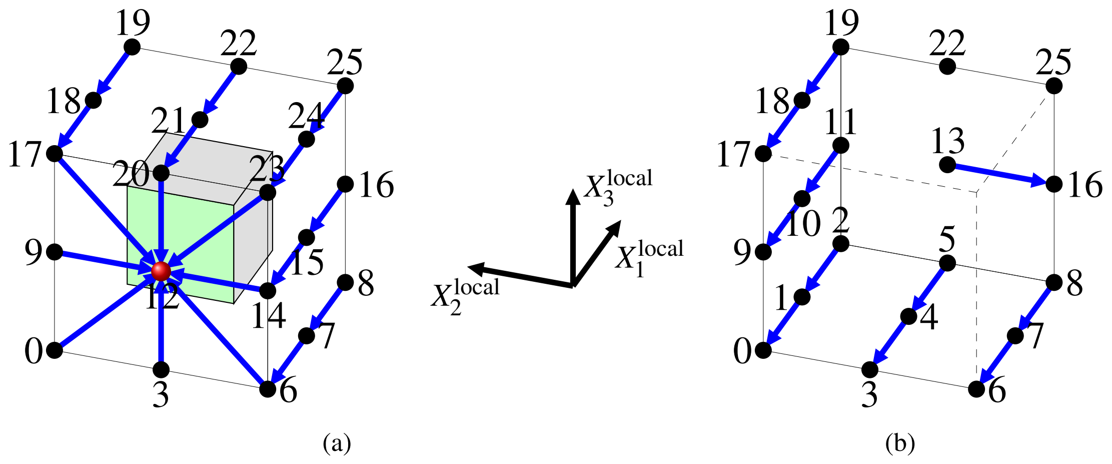


Fig. 4.17 Visualization of the graph representing the 26 global sensing states (nodes) of a convex-shaped 3D robot with respect to the goal. The robot is represented by the grey cube, with the green face representing the preferred face. Each node represents a position of the goal with respect to the robot (corner point, edge without corner points, face without edges) with its name corresponding to the robot's feature that contains the closest point to the goal. Transitions between nodes, that is movements of the goal with respect to the robot(directed edges) are those chosen by the decentralised `3D-5SP-V2` controller, which has full information about the global state. For any given node, the controller follows a unique path that is guaranteed to terminate in the preferred state (node 12) after at most four transitions. In (a) the front, right and top of the graph are represented. In (b) the back, left and bottom of the graph are represented.

Theorem 8. *An $a_1 \times a_2 \times a_3$ robot (with $a_k, a_l \geq 2$ for some $k \neq l$), using the* `3D-5SP-V2` *controller completes the task in finite time.*

Proof. As before, we need to consider only the case $\bigvee_{j \neq 1} d_j = 1$. From the definition of b_j, it follows that $b_2 \vee b_3 \vee b_4 \vee b_5 \vee b_6 = 1$. As such, the last term of Equation (4.80) disappears. If $b_5 b_6 = 0$, two of Equations (4.85)–(4.90) (p_5 and p_6) return 0, whereas the remaining four equations assume a symmetric form, causing the modules belonging to the robot's edges that are parallel to X_3^{local} to activate $4a_3$ pumps in total. If $b_2 b_3 b_4 = 0$, two of Equations (4.85)–(4.90) (p_3 and p_4) return 0, whereas the remaining four equations assume a symmetric form, causing the modules belonging to the robot's edges that are parallel to X_2^{local} to activate $4a_2$ pumps in total. By applying the analysis for Theorem 5, one can show that a new situation, with different values for the power lines, is reached in finite time. If $\overline{b_5 b_6} = 1 \wedge \overline{b_2 b_3 b_4} = 1$, all of Equations (4.85)–(4.90) return 1, where p_1 and p_2 will have a symmetric component to both p_3 and p_4, and p_5 and p_6. This causes the modules belonging to the robot's edges that are parallel to X_2^{local} and X_3^{local} to activate $4a_2$ and $4a_3$ pumps, respectively. This generates a rotation axis that is not one of the major axis. However, as it still is a 2D rotation, the the analysis for Theorem 5 still holds, and a new situation, with different values for the power lines, is reached in finite time. Figure 4.17 shows the graph of transitions, between the 26 possible global sensing states, realized by the controller. The graph is acyclic. Moreover, from any starting orientation (i.e. global sensing state) there is a unique path to reach the desired orientation, which corresponds to global sensing state 12. As any path can be completed by at most four steps (i.e. 2D rotations), the overall task is completed in finite time. ■

4.5.2.2 Controllers without shared power lines: `3D-OSP-V1` and `3D-OSP-V2` controllers

Both versions of the controller without shared power lines travel the graph shown in Figure 4.17. Theorem 8 and Lemma 3 are not sufficient to guarantee the success of the controllers that do not use power lines. By using a similar analysis to the 2D case, we can determine a minimum distance that guarantees the robot completes the task using these controllers.

For the `3D-OSP-V1` controller, the maximum force and minimum torque, respectively, are:

$$f \leq \sqrt{a_2^2(a_3-1)^2 + a_2^2(a_1-1)^2 + a_3^2(a_1-1)^2} < \sqrt{a_2^2a_3^2 + a_2^2a_1^2 + a_3^2a_1^2} \tag{4.105}$$

$$\tau \geq \frac{a_3(a_1-1)}{2} \tag{4.106}$$

This controller allows for rotations along two axes simultaneously. However, the times for the rotations along each of the axis is independent. As a result, to obtain a minimum distance condition we can consider only the longest rotation. The maximum angle offset is $\Delta\theta = \pi$. When replacing in Equation (4.102) we obtain:

$$\Delta d_1 \geq 2\pi k_r \frac{\sqrt{a_1^2a_2^2 + a_1^2a_3^2 + a_2^2a_3^2}}{a_3(a_1-1)}, \text{ where } k_r = \frac{k}{k_\theta} \tag{4.107}$$

For the `3D-OSP-V2` controller, the maximum and minimum possible force and torque, respectively, are:

$$f = \sqrt{a_2^2(a_3-1)^2 + a_2^2 + a_3^2} < \sqrt{a_2^2 a_3^2 + a_2^2 + a_3^2} \tag{4.108}$$

$$\tau = min\left((a_3-1)a_2, \frac{a_3(a_1+a_2-2)}{2} \right) \tag{4.109}$$

The maximum angle offset, for which the robot translates towards the goal, is $\Delta\theta = \frac{\pi}{2}$. When replacing in Equation (4.102) we obtain:

$$\Delta d_2 \geq \begin{cases} \pi k_r \dfrac{\sqrt{a_2^2 a_3^2 + a_2^2 + a_3^2}}{2a_2(a_3-1)}, & \text{for } 3 \leq a_1 \leq \dfrac{a_2 a_3 - 2a_2 + 2a_3}{a_3}; \\ \pi k_r \dfrac{\sqrt{a_2^2 a_3^2 + a_2^2 + a_3^2}}{a_3(a_1+a_2-2)}, & \text{for } a_1 > \dfrac{a_2 a_3 - 2a_2 + 2a_3}{a_3}; \end{cases} \tag{4.110}$$

where, $k_r = \frac{k}{k_\theta}$.[14]

As for the 2D controllers, if the conditions obtained in Equations (4.107) and (4.110) are not met, the controllers are not guaranteed to succeed, and the robot can either reach the goal with the incorrect orientation (`3D-OSP-V1` controller), or orbit around the goal without ever reaching it (`3D-OSP-V2` controller).

As with the 2D case, due to starting the active translation only after being closer to the desired orientation, the minimum distance to guarantee the success of the `3D-OSP-V2` controller is smaller than the one for the `3D-OSP-V1` controller.

4.6 Analysis for robots of generic shape

In all of the previous sections of this chapter, our focus has been on robots of convex shape. However, modular robots, and the MHP robot in particular, can assume any lattice configuration or shape. In this section we expand the analysis of the controllers to concave shape robots. We will start with robots in the 2D space, and then expand to 3D.

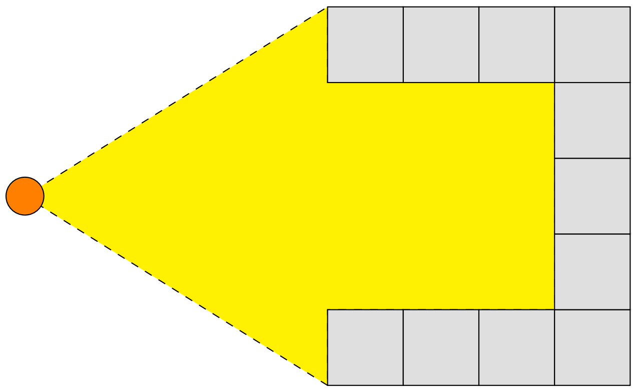


Fig. 4.18 Example of a robot of concave shape, with two symmetric faces detecting the goal. The robot is facing the preferred direction. The orange circle represents the goal; the illuminated faces of the robot can detect the goal.

4.6.1 Limitations of the current solutions

The design of our control solutions was made with the assumption that two faces with anti-parallel normals could not detect the goal simultaneously. However, that possibility exists for robots of generic shape. Figure 4.18 shows an example of a concave robot for which two faces, with anti-parallel normals, detect the goal. For this specific example, assuming the robot is oriented in the preferred orientation, only the controllers using no shared power lines would be able to complete the task. For controllers that use shared power lines, but have limited information about the robot's global sensing status, the robot would rotate indefinitely (there is always, at least, one power line active—see Equations (4.47)–(4.50) and (4.73)–(4.78)). For controllers that use shared power lines, and have full information about the global sensing status, the robot would not move (at least two power lines active, do not allow the robot to translate, and no rotation is generated—see Equations (4.42)–(4.45) and (4.79)–(4.90)). As such, the extension of the analysis to concave robots requires that we adapt the controllers.

However, there is no guarantee that, using only the robot corner/edge modules to rotate the robot, will generate a rotation, for a robot of generic shape. As such, before going through the controller adaptation for generic shapes, we need to assess if a robot of generic shape can rotate when using similar control strategies.

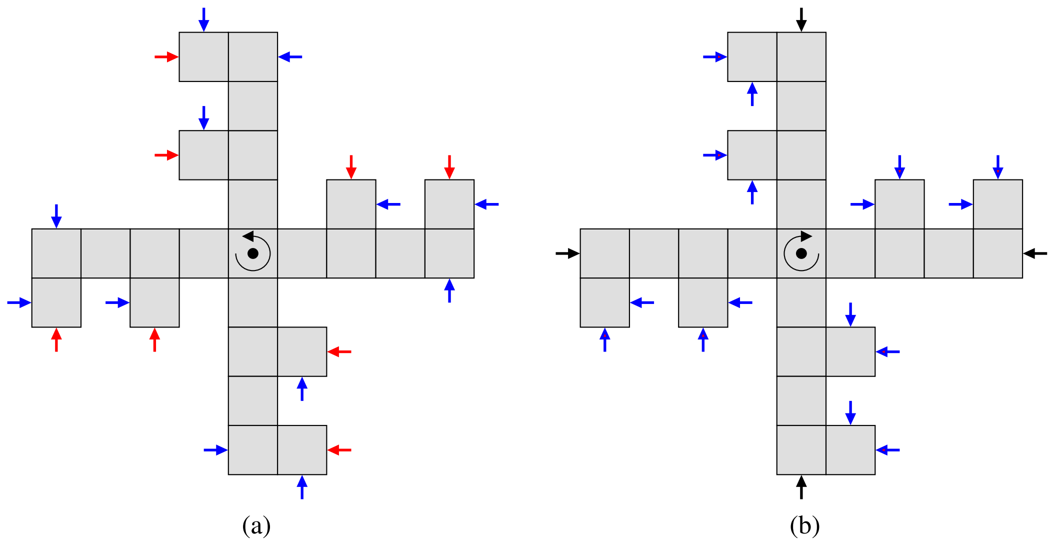


Fig. 4.19 Counter-example for non-convex robot rotation, for a robot with no empty closed spaces. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. The arrows represent the forces that create a torque that goes in favour (blue), against (red) or have no impact (black) on the desired rotation. (a) Counter-clockwise desired rotation. The robot does not rotate as the final torque is zero. (b) Clockwise desired rotation, the robot rotates in the desired rotation.

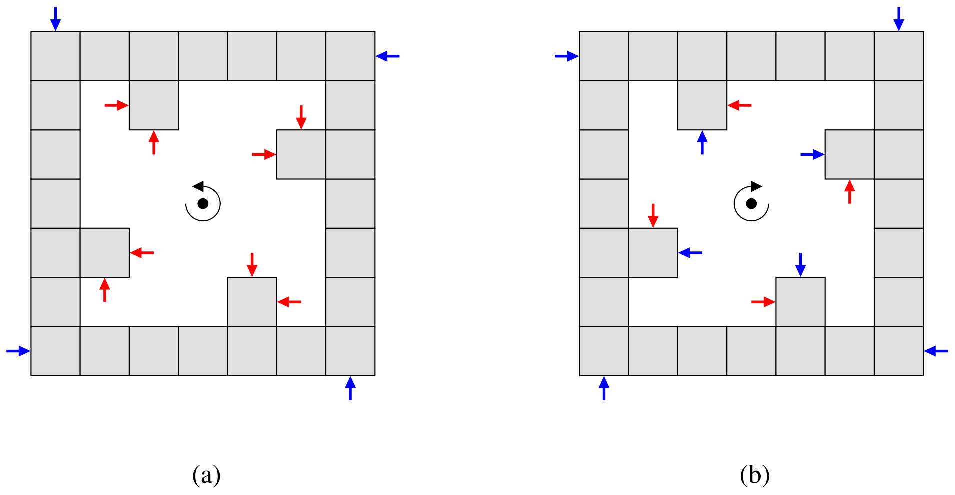


Fig. 4.20 Counter-example for non-convex robot rotation, for a robot with empty closed spaces. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. The arrows represent the forces that create a torque that goes in favour (blue), against (red) or have no impact (black) on the desired rotation. (a) Counter-clockwise desired rotation. The robot does not rotate as the final torque is zero. (b) Clockwise desired rotation, the robot rotates in the desired rotation.

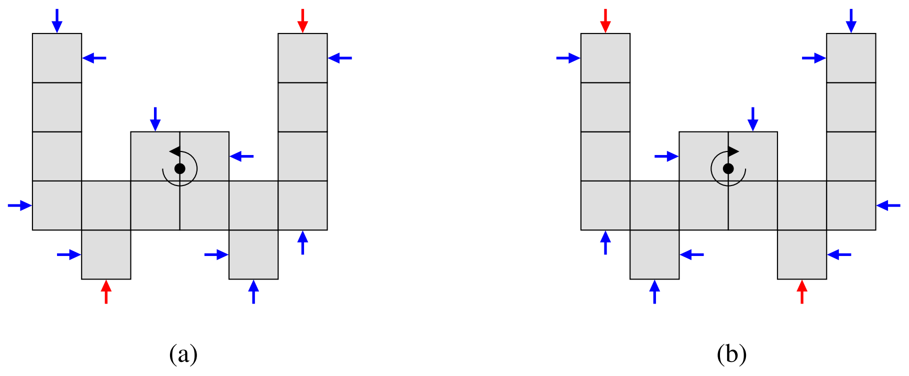


Fig. 4.21 Forces acting on a non-convex robot, symmetric along X_2^{local}, in a counter-clockwise (a) and clockwise (b) rotation. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. The blue and red arrows correspond to the forces acting in favour and against the rotation, respectively. The forces acting on the robot rotating on a clockwise direction are a mirror image to the forces acting on the robot rotating on a counter-clockwise direction. The generated torque is the same for the robot rotating on either direction.

4.6.2 Counter-Examples

In order to guarantee that the robot can rotate, the torque acting on the robot needs to be non-zero. Figures 4.19 and 4.20 show examples of robots, of concave shapes, for which the torque generated in one rotation direction, using only corner modules, is zero. However, for both examples, the torque is only zero for one specific rotation direction (Figures 4.19a and 4.20a); for the opposite rotation direction (Figures 4.19b and 4.20b) the robot is able to rotate.

From the counter-examples given, we know that there are some robot configurations that produce a zero torque when trying to rotate in one direction. As the zero torque occurs only for one rotation direction, the controller adaptation could take that into account to determine the chosen direction of rotation. However, that is only possible if there is no configuration that produces a zero torque when trying to rotate in either direction.

Theorem 9. *Robots that are symmetric along axis X_2^{local} produce the same torque when rotating in a counter-clockwise direction as when rotating in a clockwise direction, that is,* $\tau^{CCW} = -\tau^{CW}$.

[14]Note that for this controller, we need $a_1, a_2, a_3 \geq 3$.

Proof. Consider a non-convex robot that is symmetric along X_2^{local}—Figure 4.21. We can define four corner module types: top-left ($\bar{c}_1\bar{c}_3$), top-right ($\bar{c}_1\bar{c}_4$), bottom-left ($\bar{c}_2\bar{c}_3$) and bottom-right ($\bar{c}_2\bar{c}_4$). Due to the symmetry, every top-left corner module placed at (x_1^{local}, x_2^{local}) has a corresponding top-right corner module placed at ($-x_1^{\text{local}}$, x_2^{local}). Similarly, every bottom-left corner module placed at (x_1^{local}, x_2^{local}) has a corresponding bottom-right corner module placed at ($-x_1^{\text{local}}$, x_2^{local}). This means that the number of corners for $x_1^{\text{local}} > 0$ is that same a s for $x_1^{\text{local}} < 0$.

From the control policies, when rotating counter clockwise, top-left and bottom-right corner modules have forces acting along X_1^{local}, and top-right and bottom-left corner modules have forces acting along X_2^{local}. For clockwise rotation, the opposite occurs; top-left and bottom-right corner modules have forces acting along X_2^{local}, and top-right and bottom-left corner modules have forces acting along X_1^{local}. As the number of corners on each side of the symmetry axis is the same, and each corner module's contribution to the total torque only depends on its position, we can look to a single side of a symmetric robot to determine the total torque acting on it.

From the above and from the symmetry, for $x_1^{\text{local}} > 0$ and for a counter-clockwise rotation, each corner module generates a torque of:[15]

$$\tau_{\bar{c}_1\bar{c}_3}^{CCW} = -x_1^{\text{local}} + x_2^{\text{local}}, \tag{4.111}$$
$$\tau_{\bar{c}_1\bar{c}_4}^{CCW} = x_1^{\text{local}} + x_2^{\text{local}}, \tag{4.112}$$
$$\tau_{\bar{c}_2\bar{c}_3}^{CCW} = -x_1^{\text{local}} - x_2^{\text{local}}, \tag{4.113}$$
$$\tau_{\bar{c}_2\bar{c}_4}^{CCW} = x_1^{\text{local}} - x_2^{\text{local}}, \tag{4.114}$$

where, x_1^{local} and x_2^{local} are the coordinates of the corner module in question.

For a clockwise rotation, the contribution of each corner module pair is given by:

$$\tau_{\bar{c}_1\bar{c}_3}^{CW} = x_1^{\text{local}} - x_2^{\text{local}}, \tag{4.115}$$
$$\tau_{\bar{c}_1\bar{c}_4}^{CW} = -x_1^{\text{local}} - x_2^{\text{local}}, \tag{4.116}$$
$$\tau_{\bar{c}_2\bar{c}_3}^{CW} = x_1^{\text{local}} + x_2^{\text{local}}, \tag{4.117}$$
$$\tau_{\bar{c}_2\bar{c}_4}^{CW} = -x_1^{\text{local}} + x_2^{\text{local}}. \tag{4.118}$$

[15] We assume that a positive torque corresponds to a counter-clockwise rotation and a negative torque corresponds to a clockwise rotation.

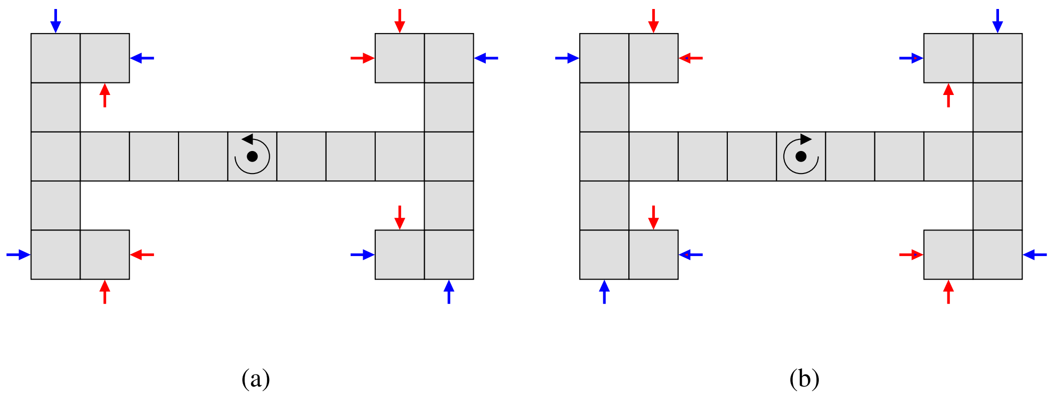


Fig. 4.22 Counter-example for non-convex robot rotation. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. The blue and red arrows represent the forces that contribute in favour or against the direction of rotation. (a) Robot trying to rotate counter-clockwise. (b) Robot trying to rotate clockwise. Both rotation directions produce a zero torque

From Equations (4.111)–(4.118) we have that $\tau^{CW}_{\bar{c}_k\bar{c}_l} = -\tau^{CCW}_{\bar{c}_k\bar{c}_l}$, for $k \in \{1,2\} \wedge l \in \{3,4\}$. When summing over all the corners with $x_1^{\text{local}} > 0$, we obtain:

$$\tau^{CCW} = -\tau^{CW} \tag{4.119}$$

■

A similar analysis can be made for robots that are symmetric along X_1^{local}. Since the premise is valid for robots symmetric along X_1^{local} and X_2^{local} separately, it is also valid for a robot that shows symmetry along both axis. When extended for 3D space, and rotating along one of the three axis of the robot's reference frame, the robot is required to be symmetric along the remaining two, to ensure the same torque is produced, in counter-clockwise and clockwise rotations.

Figure 4.22 shows a robot configuration, that is symmetric along one of the coordinate axis, and produces a zero torque when trying to rotate in either clockwise or counter-clockwise rotation. Therefore, using the local connectivity of the module to determine its contribution to the rotation, robots of generic shape are not guaranteed to succeed.

The counter-examples presented can be extended for 3D, simply by stacking copies of the presented geometries along the third axis. As, for at least one of the rotation axis, the robot produces a zero torque, we cannot guarantee the desired orientation can be reached,

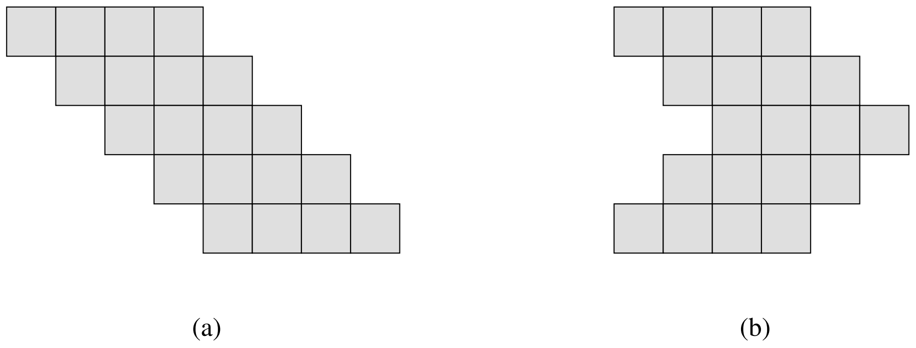


Fig. 4.23 Example of an orthogonally convex robot (a) and non orthogonally convex robot (b). For a orthogonally convex robot, any vertical or horizontal line drawn between two points of the robot is contained by the robot's shape.

when using the `3D-2SP` and `3D-5SP-V1` controllers, as they require rotation along all of the robot's reference frame axis to cover all possible initial orientations. Moreover, when using the `3D-5SP-V2` controller, if the rotation axis along which the robot produces a zero torque is not perpendicular to the desired direction, the desired orientation might not be reached as well.

4.6.3 Orthogonally convex robots

In the previous section, we showed that our current strategy, of using the connectivity of the corner modules to determine their contribution to the rotation, is not viable for robots of generic shapes. However, we can still extend the controllers to specific classes of geometries. In particular we analyse orthogonally convex robots.

Definition 1. *A robot $\mathcal{A}$ is considered orthogonally convex if the line connecting any two points $(x_0, y_0), (x_1, y_1) \in \mathcal{A}$, with $x_0 = x_1 \vee y_0 = y_1$, is contained in $\mathcal{A}$.*

When analysing rectilinear objects, such as the geometries formed by the MHP robot, the geometric property that can play the same role as standard convexity is orthogonal convexity [111]. For an orthogonally convex robot, any vertical or horizontal line can intersect the boundary of the robot a maximum of two times. An example of an orthogonally convex robot is presented in Figure 4.23a. If a vertical or horizontal line intersects the boundary more than two times the robot is not orthogonally convex (Figure 4.23b).

When considering orthogonally convex robots, the proposed controllers can be used, without requiring adaptation, as by definition, an orthogonally convex robot cannot detect the

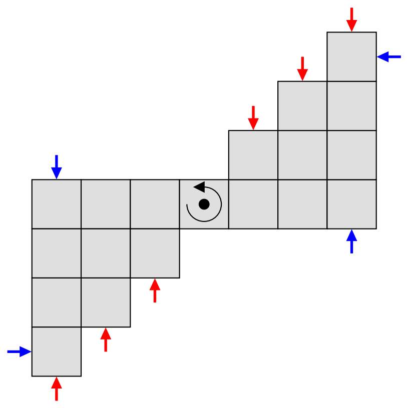


Fig. 4.24 Counter-example for an orthogonally convex robot. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. The blue and red arrows represent the forces that contribute in favour or against the direction of rotation. The torque acting on the robot, when rotating counter-clockwise is zero. As such, the robot is unable to rotate in a counter-clockwise rotation.

goal from two opposing directions, if there is a single goal in the environment. However, it is not possible to ensure that no translation occurs, while rotating, if the robot is not symmetrical along all of its reference frame axes.

As with the case of robots of generic shape, for the controllers to work, we need to guarantee that exists no robot configuration that produces a zero torque. Figure 4.24 shows one such configuration, where the torque generated for a counter-clockwise rotation is zero. Similarly to the robots of generic shape, as the robot in Figure 4.24 is not symmetric along any of the robot's reference frame axis, the torque for the rotation in the opposite direction—in this case, clockwise—is non zero.

Theorem 10. *Robots, of dimension $a_1, a_2 > 1$, that are orthogonally convex and symmetric along X_2^{local} produce a non zero torque when rotating, in any given direction.*

Proof. We assume a counter-clockwise rotation, $x_1^{\text{local}} \geq 0$ (calculations performed for left corners) and that the center of mass of the robot is place at $(0,0)$[16]. We start with a convex rectangular configuration where there are only four corners, one of each type. From the orthogonal convexity and symmetry, the top-left and top-right corners have the same x_2^{local} coordinate, and symmetrical x_1^{local} coordinate. As a result, and from Equation (4.112), we know that $\tau_{top} = max(x_1^{\text{local}}) + max(x_2^{\text{local}})$. Similarly, the torque generated by the bottom-left

[16]For a robot to be orthogonally convex (but not convex) it is required that both dimensions are larger than zero, that is $max(x_1^{\text{local}}) - min(x_1^{\text{local}}) > 0 \wedge max(x_2^{\text{local}}) - min(x_2^{\text{local}}) > 0$. Additionally, due to symmetry $max(x_1^{\text{local}}) = -min(x_1^{\text{local}})$.

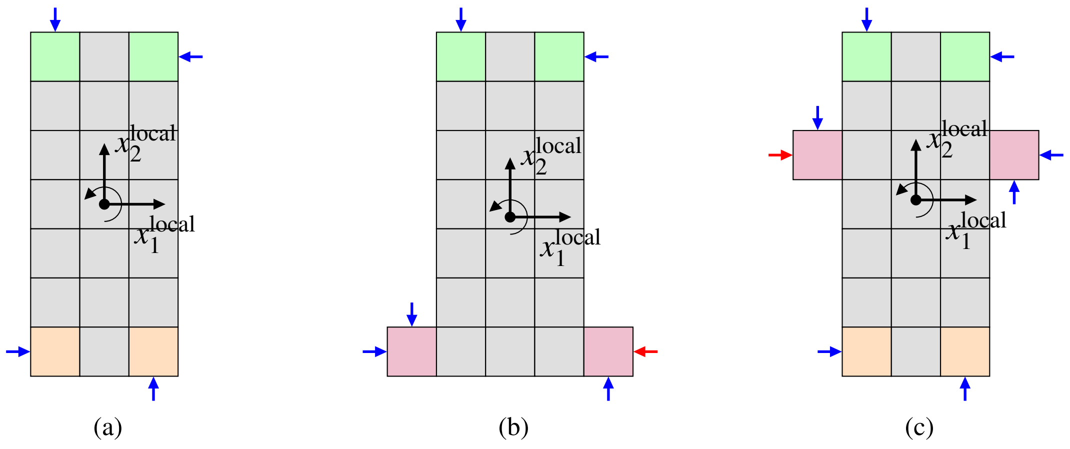


Fig. 4.25 Example of a convex robot (a) and symmetric orthogonally convex robots obtained by the addition of top (b) and both top and bottom (c) corners to the convex robot. The center of mass and desired rotation direction are represented by the point and arrow in the center of the robot. Light green modules act as top corners, light orange modules act as bottom corners and light purple corners act as top and bottom corners simultaneously.

and bottom-right corners is given by $\tau_{bottom} = max(x_1^{\text{local}}) - min(x_2^{\text{local}})$ (Equation (4.114)). Both torques are positive (contribute to a counter clockwise rotation) as τ_{top} occurs for $x_2^{\text{local}} > 0$ and τ_{bottom} occurs for $x_2^{\text{local}} < 0$ (Figure 4.25a).

For the remaining corners, from the orthogonal convexity, three possibilities exist: addition of top corners, addition of bottom corners, and addition of both[17]. If only top corners are added two cases are possible. If $x_1^{\text{local}} \geq -x_2^{\text{local}}$, from Equation (4.112), the contribution of the top corner is positive. If $x_1^{\text{local}} < -x_2^{\text{local}}$, the contribution of the added corners is negative. In the worst case possible, where two modules[18] are added on each side of the robot at $min(x_2^{\text{local}})$ (e.g. Figure 4.25b), the torque acting on the robot goes from $\tau = 2max(x_1^{\text{local}}) + max(x_2^{\text{local}}) - max(x_2^{\text{local}})$ to $\tau = 3max(x_1^{\text{local}}) - 1 + max(x_2^{\text{local}})$. Since $max(x_1^{\text{local}}) > 0 \wedge max(x_2^{\text{local}}) > 0$, the torque acting on the robot is still positive.

As more top corners are added, to keep the symmetry and orthogonal convexity, $max(x_2^{\text{local}})$ decreases, but remains $max(x_2^{\text{local}}) > 0$, while $max(x_1^{\text{local}})$ will increase for each corner added. As a result, the negative impact of every extra top corner decreases, up to the point where $x_2^{\text{local}} > 0$ for new corners, guaranteeing the the torque is always positive.

Adding only bottom corners will lead to a symmetrical situation (rotation of 180° of the previous case).

[17] As the configuration is symmetrical, adding top-right corners requires the addition of top-left corners, in order to not break symmetry. The same applies for bottom corners.

[18] We assume that the size of the modules is 1.

The last possibility is the addition of adding both top and bottom corners. If $max(x_1^{\text{local}})$ happens for $x_2^{\text{local}} = 0$, the torques produced by the additional corners are positive if $x_2^{\text{local}} > 0$ for top corners and $x_2^{\text{local}} < 0$ for bottom corners. If $max(x_1^{\text{local}})$ happens for $x_2^{\text{local}} > 0$ (Figure 4.25c, all top corners added contribute with a positive torque. In this case, bottom corners can contribute with a negative torque, however, due to the orthogonal convexity, the absolute value of the torque generated by bottom corners is smaller of equal to the contribution of top corners. A similar result is obtained in situations in which $max(x_1^{\text{local}})$ happens for $x_2^{\text{local}} < 0$. As a result, the addition of both top and bottom corners to a convex robot can only increase the total torque acting on the robot, which leads to $\tau > 0$, for any orthogonally convex robot, symmetric along X_2^{local}. From the result of Theorem 9, since $\tau = 0$ is not possible for counter clockwise rotation, it is also not possible for clockwise rotation.

$\blacksquare$

A similar analysis can be made for robots that are symmetric along X_1^{local}. Since the premise is valid for robots symmetric along X_1^{local} and X_2^{local} separately, it is also valid for a robot that shows symmetry along both axes. When extended for 3D space, and rotating along one of the three axes of the robot's reference frame, the robot is required to be symmetric along one of the remaining two, to ensure a non zero torque is produced, in counter-clockwise and clockwise rotations.

From the analysis performed in this section, the control strategies proposed in this chapter are guaranteed to complete the task for orthogonally convex robots, symmetric along one of the axes perpendicular to the axis of rotation. Moreover, no controller adaptation is required to extend the solutions to these shapes.

4.7 Summary

In this chapter, we begin with an analysis of the movement capabilities of an MHP robot, when moving in 3D, both in terms of translation and rotation. The analysis of the rotation was completed for rotations along a single or multiple axis of rotation. The high number of unique total forces and net torques a robot can produce enables it to control its translation and rotation velocities with precision. From the expressions obtained, if the modular resolution of the robots is increased, while keeping the robot size constant, the number of unique torques increases. This allows for increased precision in controlling the robot's movement.

We proposed a set of fully reactive and decentralised controllers for modular reconfigurable robots that perform pose control in liquid environments. The control strategies enable a convex-shaped robot to reach the goal with a preferred face. The robot uses simple binary sensors and pumps. Additionally, the strategy allows the modules to access up to 5 bits of information through shared power lines.

We formally proved that the controllers that allow the modules to share information through the use of shared power lines are guaranteed to succeed when moving in both 2D and 3D environments. For the controllers that do not use shared power lines, we showed that they require a minimum initial distance from the goal in order to be able to complete the task, regardless of initial orientation. We extended the analysis of the controllers to generic (non-convex) configurations. We presented a set of counter-examples showing that the control strategy—using the local connectivity of each module to determine its contribution to the rotation—would need to be changed in order to allow these robots to complete the task. We were, however, able to extend the use of the controllers for orthogonally convex robots, that present one or more symmetry axis. For this class of geometries, the controllers do not require any adaptation. However, depending on the geometry of the robot, some translation may be observed during the rotation phase of the movement.

All the results presented in this chapter are theoretical. As a result, the behaviours expected from the controllers are based on assumptions that might not prove correct when considering real-world environments. For this reason validation of the proposed controllers is required to ascertain if the controllers produce the expected behaviour, and if increases in the amount of information accessible to the modules through shared power lines translates to better performance.

Chapter 5

MHP Pose Control Validation

This chapter is derived from author's original contributions to Sections 5 and 6 of the publication [114].

5.1 Introduction

In the previous chapter we presented multiple reactive and decentralised strategies for controlling the pose of an MHP robot moving in 2D or 3D. The strategies differed based on the amount of information shared between the modules, through shared power lines. While the controllers were shown to be able to complete the task, the analysis relies on strict assumptions, such as quasi-static motion, that might not be applicable when used in real-world applications. For this reason, the validation of the controllers is required.

To validate the controllers we first simulate the movement of MHP robots in 3D when using the proposed controllers. The robots are required to reach a goal point with a specified orientation in an unbound environment. To better understand the advantages and limitation of each controller, we study their performance in different environments and conditions. Additionally, we perform physical experiments, using the hardware developed in Chapter 3, to test the performance of the controllers in 2D.

The remainder of this chapter starts by presenting the details regarding the simulation studies in Section 5.2. This includes details of the simulator and simulation setup in Section 5.2.1. Section 5.2.2 describes the implementation of the multiple controllers and Section 5.2.3 presents an analysis of the robot's movement for each controller and the results for the multiple studies performed. The simulations are followed by physical experiments in Section 5.3. Details of the experiments set up and controller implementation are described in Section 5.3.1 and 5.3.2. The results are presented and discussed in Section 5.3.3. Finally, Section 5.4 concludes the chapter.

5.2 Simulation Studies

In this section we present the details regarding the simulation studies performed not only to validate the controllers, but also to evaluate the trade-offs between performance and complexity, in a variety of scenarios.

5.2.1 Simulation setup

In this section, we describe the simulator, as well as setup used for the simulations. The simulator used was built using the open-source Open Dynamics Engine (ODE) library [117], and is an adaptation of the simulator presented in [115]. Each module is simulated as a solid cube, with side length 1 cm and mass $m = 1\,\mathrm{g}$. The modules are neutrally buoyant, that is, the densities of the fluid and of the modules is equal. The module's faces are numbered from one to six, based on the face's normal directions—odd numbered faces ($j = 2i - 1$) have normal vectors anti-parallel to X_i^{local} and even numbered faces ($j = 2i$) have normal vectors parallel to X_i^{local}. Placed at the centre of each module's face is an actuator, a connection sensor and a light sensor. By default, each active actuator exerts a force $f_p = 5.25\,\mu\mathrm{N}$, applied in the centre of the module's face, in an anti-parallel direction to the face's normal.We chose this value for the force exerted by each actuator, so that in water like conditions the robot is able to travel at a speed of 1 cm/s . Internal fluid movement and forces associated with it, are not simulated.

We simulate a 3D, continuous and unbounded liquid environment. The environment consists of a liquid with constant density $\rho = 1\,\mathrm{g/cm^3}$ that contains a single modular robot and a goal point, shown in Figure 5.1. By default, the simulations are conducted using a $10 \times 10 \times 10$ robot, that is, a cubic robot of 10 cm body length. All the modules in the robot are oriented consistently, that is, the modules and the robot share the orientation of the reference frame. The goal point consist of a rigid sphere of diameter $d = 1\,\mathrm{cm}$. The origin of the world reference frame coincides with the starting position of the robot. By default, the

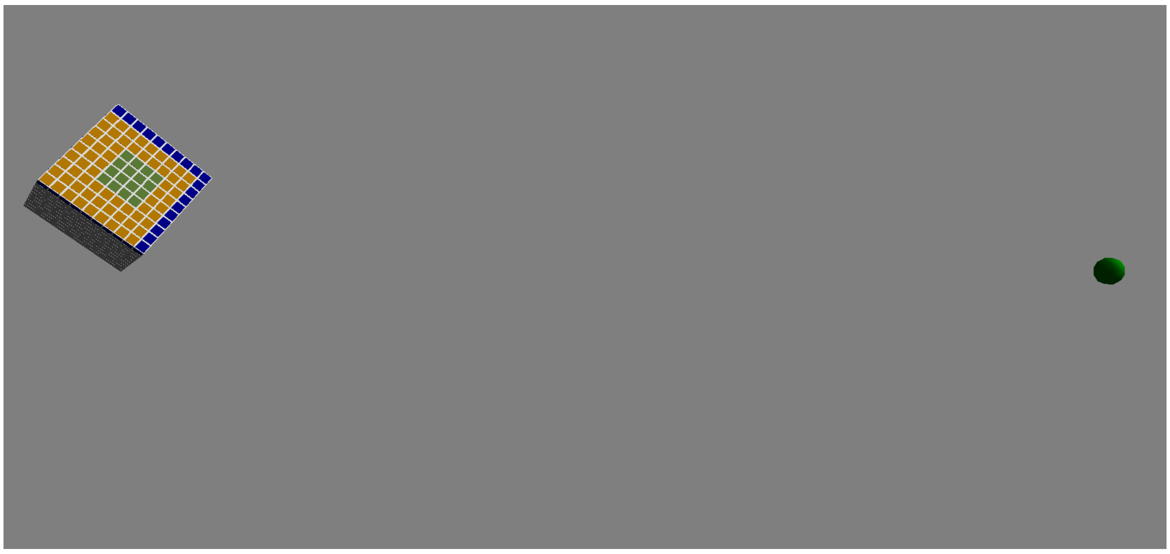


Fig. 5.1 Image from the simulator showing the default simulation environment, consisting of a $10 \times 10 \times 10$ robot and a goal point. The goal is represented by the green sphere. The robot is placed 85.5 cm away from the goal. In yellow are represented the module faces that can detect the target. The green faces are used to mark the face with wich the robot needs to reach the target. In blue are represented the active pumps.

robot is placed 85.5 cm away from the goal[1]. To reach the goal, it has to travel a distance of approximately 8 body lengths[2]. The robot's initial orientation is chosen at random from a uniform distribution. Unless otherwise specified, the time limit for each trial is 1200 s. A simulation trial terminates if the robot collides with the goal or the time limit is reached. A trial is deemed successful if the point of collision between the robot and the goal belongs to the preferred face of the robot (face 1). The point of collision between the robot and the goal is determined using the collision detection functionalities of ODE. This is used to determine if the robot reached the goal with the correct orientation.

The only forces acting on the robot are the forces generated by the modules and the forces associated with fluid drag. As the computational cost of a full fluid dynamics treatment would be significant [71], drag forces are determined using a quadratic approximation. The drag force is calculated for each external module face, and then summed over all external

[1]The initial distance between the goal and the robot is measured from the center of the robot to the center of the goal, and is determined by the number of robot body lengths has to travel, plus half of the robot size plus half of the goal size.

[2]If the robot does not travel in a straight line towards the goal, the travel distance can be more than 8 body lengths. In a similar fashion, if the robot collides with the goal at an angle, the travel distance can be less than 8 body lengths

faces. The drag force $\mathbf{f}_D$ acting on an external module face is determined by:

$$\mathbf{f}_D = \begin{cases} -\frac{1}{2}\rho C_D \varepsilon^2 \dot{\mathbf{q}} \cdot \dot{\mathbf{q}}\mathbf{u}, & \dot{\mathbf{q}} \cdot \mathbf{u} > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

where ρ is the fluid density, ε the module side length, C_D the drag coefficient value, $\dot{\mathbf{q}}$ the velocity of the center point of the module's face and $\mathbf{u}$ the face normal. Using $C_D = 1.05$ (drag coefficient value of a cube translating in a direction perpendicular to a face), the robot can reach a terminal velocity of 1 cm/s when translating.

We test the decentralised controllers presented in Section 4.4.2. We evaluate the impact that changes in fluid drag, initial distance between the robot and the goal, modular resolution, and actuator and sensor noise have each controller's performance. Additionally, we compare the decentralised controllers results against a centralised controller from the literature [27].

The centralised controller used is a proportional-integral-derivative (PID) controller, and requires the position and orientation of the robot, as well as the positions of its thrusters, as inputs. The errors in position and orientation of the robot are used to determine the thruster output of the robot. In the original controller, each thruster is given an output from within the working range of the thruster [27]. In our implementation, the centralised controller uses the same set of binary pumps as the decentralised controllers.

We calibrated the proportional, integral and derivative parameters of the controller via grid search, for the default simulation setup. Figure 5.2 shows the completion times for the sets of parameters tested, with 100% success rate. The values obtained through the calibration ($P_T = 1$, $I_T = 0$, $D_T = 1$, $P_R = 1000$, $I_R = 0$, $D_R = 1000$, PID set index $= 80$) are used for all setups used in the multiple studies. However, this means that, for other than the default setup, the performance of this controller can be lower than expected.

5.2.2 Controller implementation

In this section, we describe the implementation of the controllers used for the simulation studies. In Sections 4.3 and 4.4 we assume that a robot moving in 3D has access to $M \in \{0,2,5\}$ shared power lines, that are used to share information between the modules. These shared power lines need to be implemented in the simulator, to obtains the behaviour produced by the decentralised controllers.

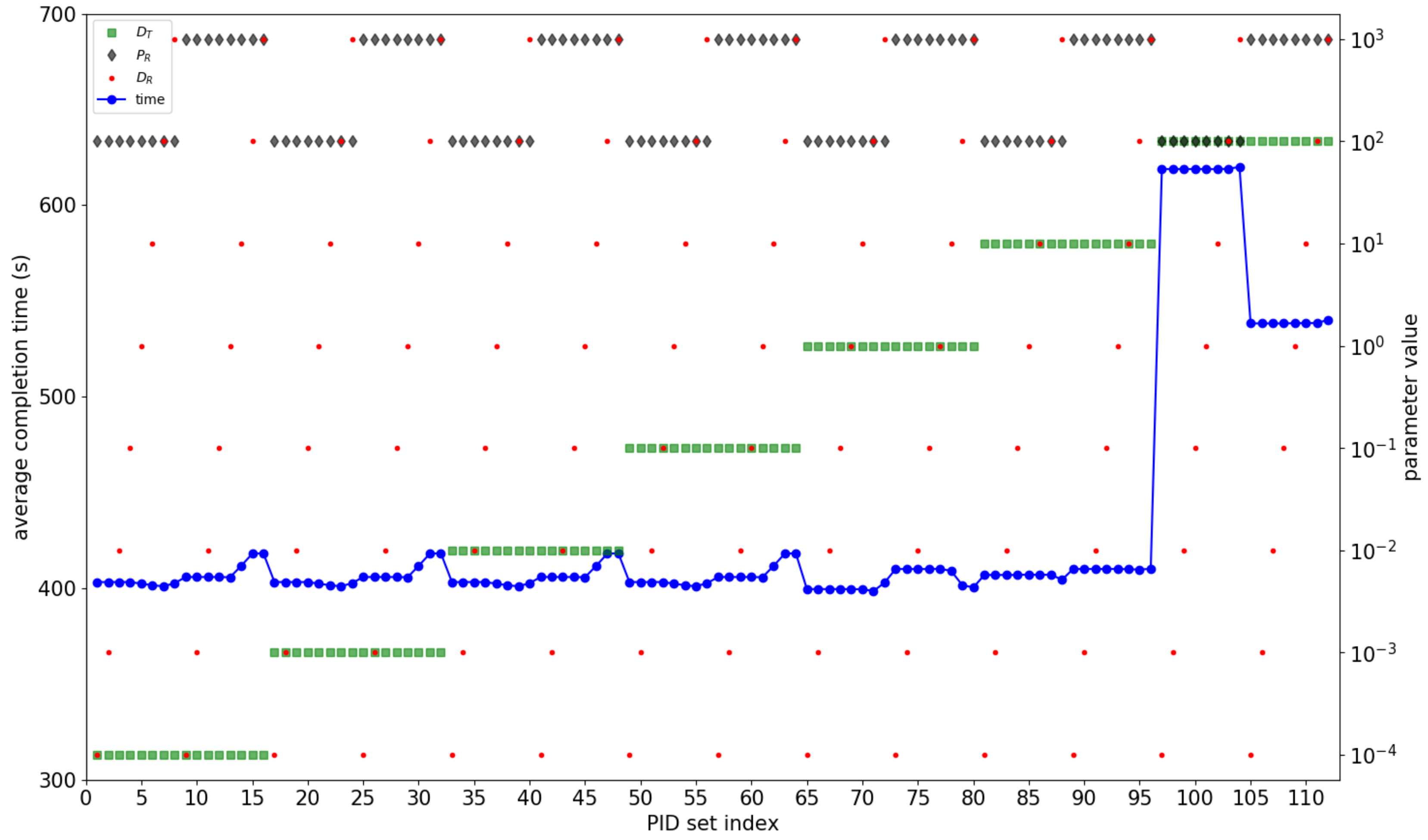


Fig. 5.2 Average completion times for sets of PID parameters producing full success rate. By default, proportional parameter for translation $P_T = 1$, integral parameter for translation $I_T = 0$, integral parameter for rotation $I_R = 0$. The values of the remaining parameters (D_T, P_R and D_R) for each set of parameters used are given in the secondary axis.

Algorithm 3 Simulator generic time step

```
1: Let $n$ be the number of modules in the robot, $t$ and $T$ be the current and maximum
   simulation time and $M \in \{0,2,5\}$ be the number of shared power lines.
2: **while** $t < T$ **do**
3:     **if** $decentralised$ **then**                              ▷ If using a decentralised controller.
4:         **if** $M \geq 2$ **then**                               ▷ If the controller uses shared power lines.
5:             **for** $i \leftarrow 1,n$ **do**                    ▷ Go through all modules.
6:                 **for** $j \leftarrow 1,6$ **do**                ▷ Go through all faces.
7:                     $d_j \leftarrow$ Goal sensor reading
8:                     $\mathbf{b} \leftarrow f_b(\mathbf{c}, d_j)$        ▷ Evaluate state of power lines.
9:                 **end for**
10:            **end for**
11:        **end if**
12:        **for** $i \leftarrow 1,n$ **do**
13:            **for** $j \leftarrow 1,6$ **do**
14:                $p_j \leftarrow f_p(\mathbf{c}, \mathbf{d}, \mathbf{b})$   ▷ Determine pump activation state, from controller.
15:            **end for**
16:        **end for**
17:    **else**                                                  ▷ If using the centralised controller.
18:        $\mathbf{q} \leftarrow robot\_position$                  ▷ Obtain the robot's position.
19:        $\theta \leftarrow robot\_orientation$                   ▷ Obtain the robot's orientation.
20:        **for** $i \leftarrow 1,n$ **do**
21:            **for** $j \leftarrow 1,6$ **do**
22:                $p_j \leftarrow PID(\mathbf{q}, \theta)$          ▷ Get pump activation, from PID controller.
23:            **end for**
24:        **end for**
25:    **end if**
26:    Do physics time-step                                      ▷ Determine current pose.
27:    **if** $collision$ **then**
28:        **break**                        ▷ If a collision is detected, terminate the simulation
29:    **end if**
30:    $t \leftarrow t + \Delta t$                                  ▷ Increase current time by $\Delta t$
31: **end while**
```

Algorithm 3 describes the steps of a generic time step for the simulator. Each simulation is run until either the time limit is reached (i.e. 1200 s), or a collision between the robot and goal has been detected. When using one of the decentralised controllers that use shared power lines, it is required to go through the faces of the modules twice: the first time to evaluate the state of the shared power lines according the functions $g_{b,k}(\mathbf{c},\mathbf{c})$ presented in Section 4.4.2 (lines 3–11), and the second to have the actuation state of the pump in each face match the state obtained by using the functions $g_p(\mathbf{c},\mathbf{c},\mathbf{b})$ presented in Section 4.4.2 (lines 12–16). As the state of each shared power line is dependent on the sensing states of all modules, it represents the minimum amount of information each module requires from a set of sensors of all the modules. In a physical robot, if the sensor readings of a particular module indicate that the k-th inactive shared power line ($b_k = 0$) should be active ($b_k = 1$), the module is capable of changing the shared power line state, while simultaneously updating its actuation state. Similarly, when a module detects a change in a shared power line it can update the actuation state of each pump. As a result, there is no significant difference between the updating of the shared power lines and actuation. However, for each module to be able to react to changes in the shared power lines in the simulator, it is required that the state of the shared power lines is known. To determine this, the simulator first needs to go through all module faces. When the decentralised controller does not use shared power lines, only one cycle though all module faces is required, as the activation state of each pump is based on the sensor readings of each face. If using the centralised controller, the position and orientation of the robot are obtained and stored (lines 18 and 19). The robot for which the centralised controller was developed had access to an IMU and GPS sensors, allowing for this information to be easily obtained. The position and orientation of the robot are used to determine the error between the current and preferred position and orientation. These errors, together with the positions of each pump, are used to determine the activation state of each pump (lines 20–24) Once the pump activation states are determined, the physics of the system are evaluated to determine the positions and velocities of the robot and each module (line 26). After completion, a test for collision between the robot and the goal is performed. If no collision is detected, the simulation moves into the next time step. If a collision is detected, the simulation terminates (lines 27–31).

5.2.3 Results

In this section, we present the results for five simulation studies performed. In each study, the impact of a different variable of the default setup is evaluated. We perform a total of 100

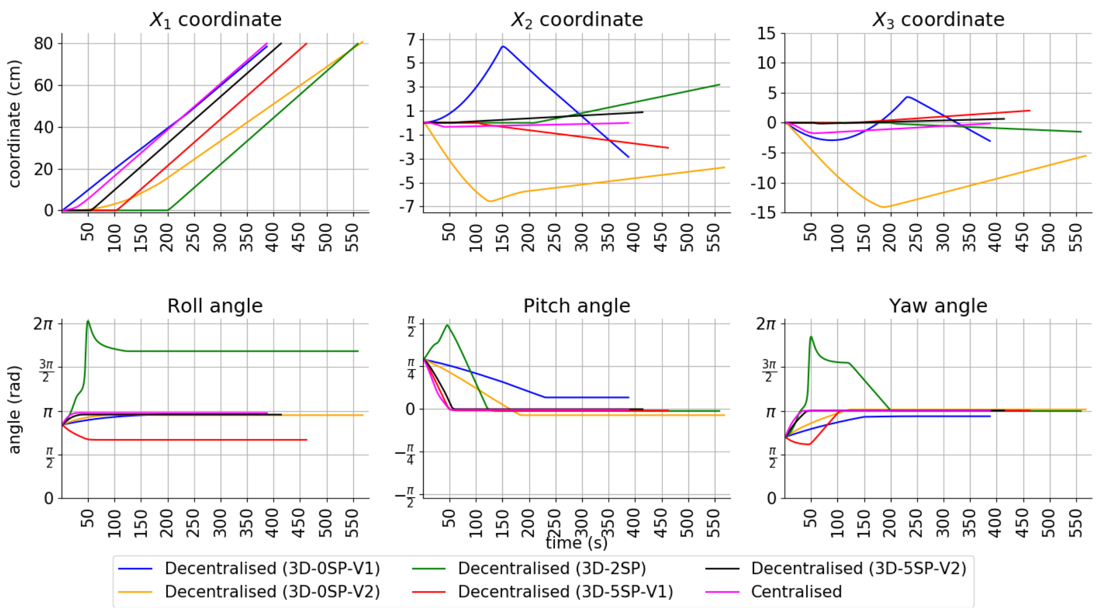


Fig. 5.3 Position and orientation of the robot over time, for a successful trial. The different lines correspond to the `3D-0SP-V1` (blue), `3D-0SP-V2` (orange), `3D-2SP` (green), `3D-5SP-V1` (red), `3D-5SP-V2` (black) and centralised (pink) controllers.

simulation trials per variable value. For each study, a set of fixed 100 random orientations is used for all variable values and controllers. Each study uses a different set of random orientations. Additionally, we perform a comparison of the behaviours and trajectories of the robot, for all controllers.

5.2.3.1 Analysis of the robot's movement for each controller

In this section, we compare the changes in position and orientation of the robot, as well as obtained trajectories, for a successful trial. The trial was chosen in such a way that rotations along all axis occur, for the `3D-2SP` controller. The robot's initial position and orientation is the same for all controllers and corresponds to $(X_1, X_2, X_3) = (0, 0, 0)$ and $(\psi, \theta, \phi) = (0.7\pi, 0.29\pi, 0.84\pi)$, with ψ, θ and ϕ corresponding to the yaw, pitch and roll angles, respectively. This orientation corresponds to the global sensing state represented by node 0 in Figures 4.15–4.17. The goal is positioned at $(X_1, X_2, X_3) = (85.5, 0, 0)$ and the preferred orientation is $(\psi, \theta, \phi) = (\pi, 0, \phi)$. The requirement for a successful trial is that the robot collides with the preferred face. As the preferred face is perpendicular to the X_1 axis, rotations along this axis do not change the face colliding with the goal. As a result, the preferred orientation can be fully defined by the yaw and pitch angles.

Figure 5.3 shows the changes in position and orientation of the robot over time, for each controller, which were obtained from the simulator[3]. For all controllers, the positions and angles present two distinct phases: rotation into preferred orientation (with or without translation) and pure translation towards the goal. During the rotation period, the robot's position only changes if using the `3D-0SP-V1`, `3D-0SP-V2` or centralised controller. The `3D-0SP-V1`, `3D-0SP-V2` cause larger changes in the X_2 and X_3 position of the robot during rotation, due to the disparity between the number of pumps contributing to rotation and translation. On the other hand, the centralised controller, can increase or decrease the number of pumps contributing to each movement. As a result, the robot shows smaller changes in its X_2 and X_3 position while rotating. In terms of the rotation movement, all controllers except the `3D-2SP` and `3D-5SP-V1` controllers reach the preferred orientation in a single rotation. While the rotation movement is the same, the time taken to reach the preferred orientation is different, due to the differences in the number of pumps contributing to the rotation. As a result, the centralised controller is the fastest, as it is able to use more than the robot edges in the rotation, and the `3D-0SP-V1` is the slowest, as it uses the smallest subset of edges for the rotation. When using the `3D-0SP-V1` controller, the robot stops rotating before reaching the preferred orientation $(\psi, \theta, \phi) = (\pi, 0, \phi)$. As the robot moves in the X_2X_3 plane during rotation, the orientation of the robot where it faces the goal with the preferred face does not match the preferred orientation. Both the `3D-2SP` and `3D-5SP-V1` controllers perform more than one rotation movement, causing them to be the slowest in reaching the preferred orientation. In the pure translation phase, all controllers generate a linear motion, where the robot directly translates towards the goal. The differences in final position of the robot are caused by differences in the over-shoot that occurs for each controller. The over-shoot also justifies small differences between the final and preferred orientation.

Figure 5.4 shows the trajectories of the robot, when using the different controllers, projected in the X_1X_2 and X_1X_3 planes. For the controllers that use shared power lines, the trajectories are straight lines towards the goal, as the robot only starts moving after reaching the preferred orientation. For the `3D-0SP-V1`, `3D-0SP-V2` and centralised controllers, the trajectory shows some curvature, while the robot is rotation, followed by a straight line towards the goal, once the preferred orientation is reached. While all of the controllers allow the robot to reach the goal, the contact points for each controller are different. The differences observed can be attributed to small overshoots that occur when the robot is de-accelerating its rotation.

[3]The robots are not capable of obtaining their absolute orientation or position. The data presented is obtained from the records of each physics time step of the simulator.

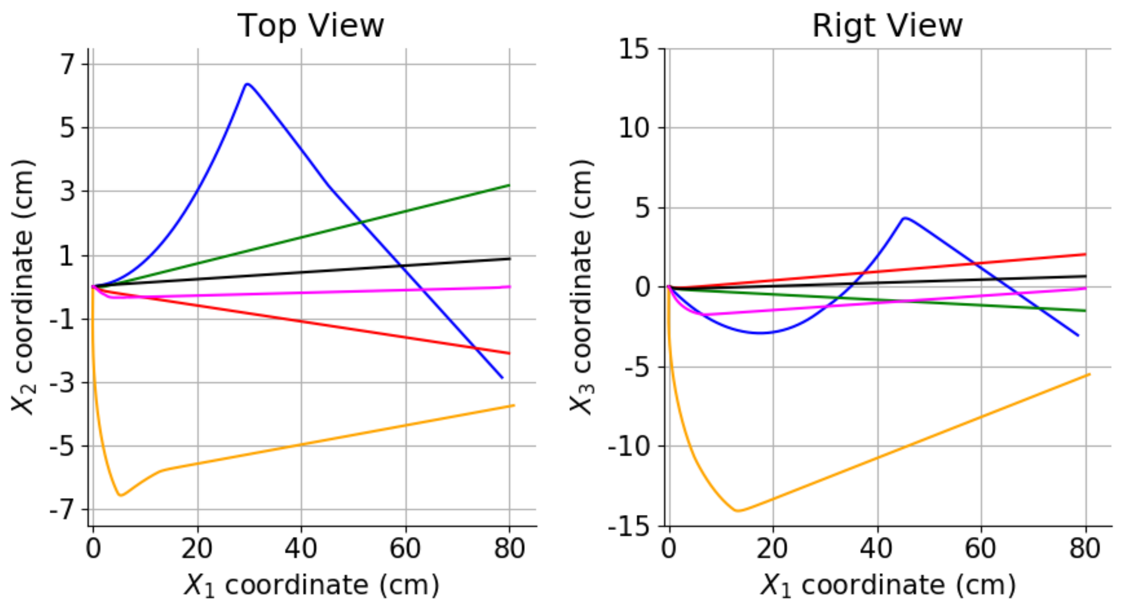


Fig. 5.4 Trajectory of the robot, viewed from two perspectives, for a successful trial. The different lines correspond to the 3D-0SP-V1 (blue), 3D-0SP-V2 (orange), 3D-2SP (green), 3D-5SP-V1 (red), 3D-5SP-V2 (black) and centralised (pink) controllers.

5.2.3.2 Study on the impact of drag coefficient on the performance of the controllers

In this section, we evaluate the performance of each controller under different drag conditions. We tested drag coefficients values of $C_D = 1.05 \times K$, where $K = 1.2^k, k \in \{1, 2, \dots, 30\}$. This allows for the evaluation of the performance of the controllers when the robot is immersed in different fluids. For each K, 100 trials were conducted for each controller.

Figure 5.5 shows the percentage of successful trials for all controllers. From Equation (5.1) we know that $v \propto \frac{1}{\sqrt{C_D}}$, where v corresponds to the robot's terminal velocity. Assuming the robot travels at terminal velocity for the full duration of the trial, in order to be able to reach the goal within the trial time limit, the robot needs to reach a terminal velocity of $v = 0.06(6)\,\mathrm{cm/s}$. As such, in general, for drag coefficients with $K > 225$, the goal cannot be reached, even when travelling at terminal velocity for the entire trial duration.

The 3D-0SP-V1 controller succeeds in about 70% of the trials for all but the highest drag coefficients ($K > 165$). In the unsuccessful trials the robot reaches the goal with a non-preferred orientation. As this controller assigns the smallest subset of the robot edges to contribute to rotation, when compared to other controllers, the toques acting on the robot—and by extension its angular velocity—is smaller. As the controller allows for simultaneous

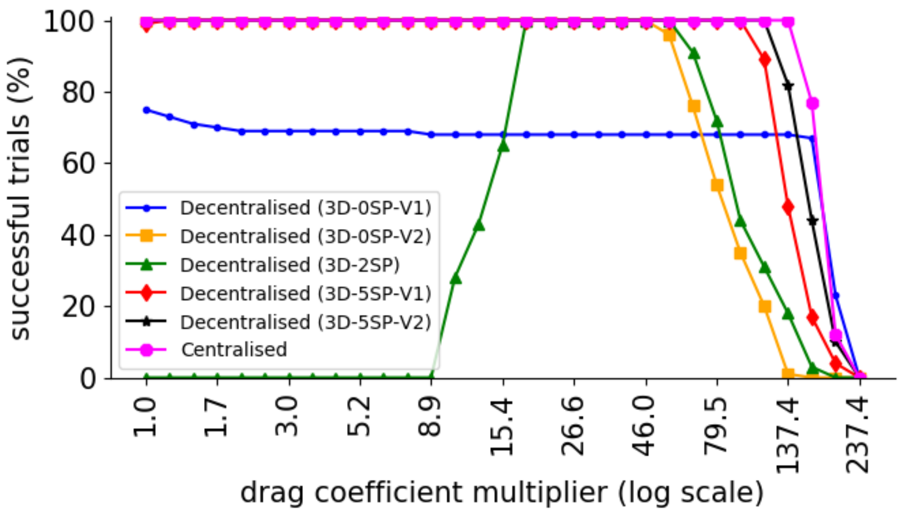


Fig. 5.5 Performance of a $10 \times 10 \times 10$ robot in environments with different drag coefficient (100 observations per setting). A trial is successful if the robot in its preferred orientation collided with the goal within a fixed time period.

translation and rotation, the angular velocity is not always sufficient to correctly align the robot prior to reaching the goal. By contrast, the `3D-0SP-V2` controller is able to successfully complete all trials for drag coefficients that have $K \leq 46$, but its performance drops with further increases in drag coefficient. For drag coefficient values with $K > 137$, the robot is unable to complete the task when using this controller. While the `3D-0SP-V2` controller also allows for simultaneous translation and rotation, there are two differentiating factors when compared to the `3D-0SP-V1` controller: (i) the translation towards the goal (henceforth denominated as active translation) only starts if $|\Delta\theta| < \frac{\pi}{2} \vee |\Delta\psi| < \frac{\pi}{2}$, where $\Delta\theta$ and $\Delta\psi$ correspond to the differences between the robot's current and preferred orientation, with the robot translating away from the goal (henceforth denominated as passive translation) if $|\Delta\theta| > \frac{\pi}{2} \wedge |\Delta\psi| > \frac{\pi}{2}$, and (ii) a larger subset of edges is is used for rotation—the edges only contribute to rotation and are not used for translation—leading to a faster rotation but slower translation. The reduced translation speed, joined together with the robot being able to translating away from the goal, leads to an earlier drop in success rate for higher drag coefficient value, when compared to the `3D-0SP-V1` controller. On the other hand, the faster rotation allows the `3D-0SP-V2` controller to perform better at lower drag coefficient values, when compared to the `3D-0SP-V1` controller.

The `3D-2SP` controller performs flawlessly for drag coefficients in the range $18.5 \leq K \leq 55.2$. For $K > 55.2$, both the linear and angular velocity are reduced to the point where the robot does not have enough time to rotate, and then translate to reach the goal. For lower drag coefficients, the drag is insufficient to counter inertia forces, causing over-shoot during rotation. For $1 \leq K \leq 8.9$ even a small rotation will cause the robot to rotate more than the required. For $8.9 < K < 18.5$ small rotations do not cause significant over-shoot, allowing trials where the differences between initial and preferred orientation are small to succeed.

The `3D-5SP-V1`, `3D-5SP-V2` and centralised controllers perform flawlessly for all but the highest drag coefficients. For high K values, the drops in performance occur first for the `3D-5SP-V1` controller, followed by the `3D-5SP-V2` controller, followed by the centralised controller. Simultaneous rotation and translation ensure the centralised controller is the fastest of the three. Due to performing shorter rotations than the `3D-5SP-V1` controller, the `3D-5SP-V2` consumes less time to reach the preferred orientation. This allows a larger portion of the trial time to be used for translation. For low K values, as the `3D-5SP-V2` and centralized controllers can rotate counter-clockwise and clockwise, along all considered axis of rotation, the over-shoot can be corrected. The `3D-5SP-V1` controller can rotate counter-clockwise and clockwise along two of the axis, but only clockwise along the third. As such, only the over-shoot along axis X_2^{local} and X_3^{local} can be corrected. The over-shoot along the X_1^{local} cannot be corrected. However, due to the sequences of rotations (see Figure 4.16), and the fact that the rotations along the two other axis can correct the over-shoot, the robot will always reach the preferred orientation, if given enough time.

The results presented show that, in general, as the amount of information the modules can share increases (increased number of shared power lines), the value of drag coefficient at which the performance of the controller starts to drop increases as well. However, for very high drag coefficient values, the `3D-0SP-V1` controller is able to slightly outperform all controllers using shared power lines. At high drag coefficient values, the controllers that use share power lines are unable to complete the task due to the limitations on robot velocity caused by the high drag. On the other hand, the `3D-0SP-V1` controller, by generating a rotation movement, while generating a translation at maximum velocity towards the goal, is capable of keeping its performance for higher drags as all other controllers start being unable to complete the task.

For the remainder of this chapter, we use a default value for the drag coefficient of $C_D = 1.05 \times 20$. This value is chosen to allow all controllers to perform their best in the default setup.

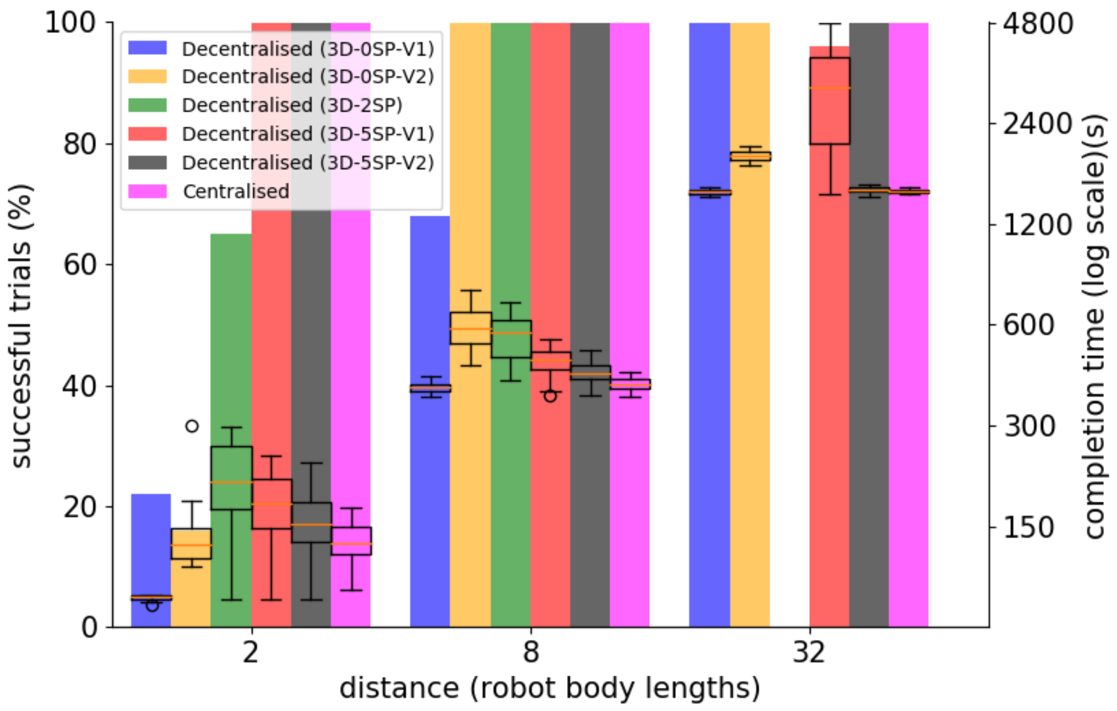


Fig. 5.6 Performance of a $10 \times 10 \times 10$ robot when the goal is initially a short, medium, or large distance away from the robot (100 observations per setting). The controllers are (from the left to the right): the decentralised controllers, `3D-0SP-V1`, `3D-0SP-V2`, `3D-2SP`, `3D-5SP-V1`, `3D-5SP-V2`, and the centralised controller. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

5.2.3.3 Study on the impact of goal distance on the performance of the controllers

From the formal analysis of the controllers done in Section 4.5.2, the performance of the controllers with no communication is dependent on the initial distance to the goal. For the remaining controllers, the distance is only expected to impact the success rate if there is not enough simulation time to rotate and reach the goal. In this study, we evaluate the controller's performance in controlling the pose of the robot over short (2 body lengths), medium (8 body lengths–default setup), and long (32 body lengths) distances. To account for the longer distances, the trial duration is adjusted accordingly to 300 s, 1200 s and 4800 s.

Figure 5.6 shows the percentage of successful trials and time it taken to reach the goal, for each distance considered. The `3D-0SP-V1` and `3D-0SP-V2` controllers exhibit improvements in success rate as the initial distance between the robot and goal increases. For long distances, both controllers complete all tasks. At short distance, however, the performance is poor. Due to simultaneous rotation and translation, shorter distances do not allow the robot to reach

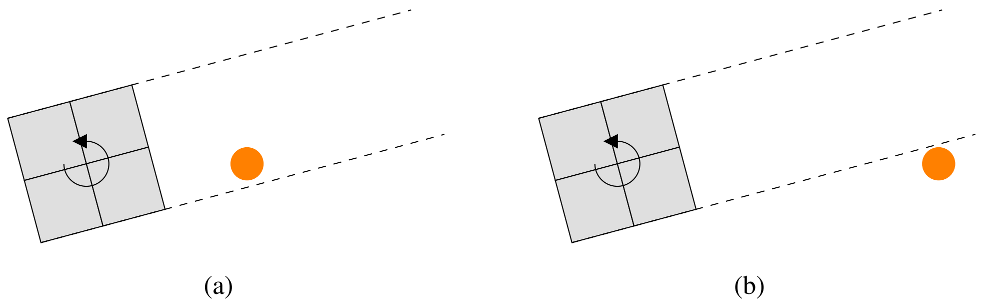


Fig. 5.7 Two robots, with the same orientation, after finishing a counter-clockwise rotation. All of the conditions are the same for both scenarios, except for the distance to the goal. The over-shoot for both case is the same. (a) While the robot over-shoot its rotation, the goal point is still within the boundaries of what is detected only by the preferred face. (b) With the same over-shoot as (a) the goal can be detected by a face other than the preferred, leading the robot to keep rotating.

the preferred orientation before reaching the goal. This effect is amplified by the fact that fewer modules contribute to rotation than for the other controllers. It is worth noting that for short and medium distances the controller that performs better, of these two controllers, is different. This situation will be analysed in more detail in Section 5.2.3.3.

The `3D-2SP` controller fails to complete some of the trials when the goal is at short distance. For medium distances, this controller is able to complete all trials. For long distances, the controller fails to complete the task for all trials. Using this controller, the robot first rotates, and only after reaching the preferred orientation does it translate. For a distance of 2 body lengths, the drag coefficient value limits the rotation and translation velocities, preventing the robot from reaching the goal within the time limit, large differences between the initial and preferred orientation. On the other hand, for a distance of 32 body lengths, the drag coefficient is not large enough to correct the rotation over-shoot, which leads to the robot being unable to reach the preferred orientation. This effect is demonstrated in Figure 5.7. As the distance between the robot and the goal increases, the maximum over-shoot for which the completion of the trial is not compromised decreases. The changes in performance of this controller with distance suggest that the liquids in which the controller is able to perform, are depended on the initial distance between the robot and the goal. To investigate this, the performance of the controller was tested for multiple drag values, for the three distances considered. Figure 5.8 shows the results. The optimal performance for the controller, in terms of drag coefficient, is shown to be dependent on the distance. As the distance between the robot and the goal increases, the curve moves to the left, meaning the controller requires a higher drag coefficient to guarantee completion of the task.

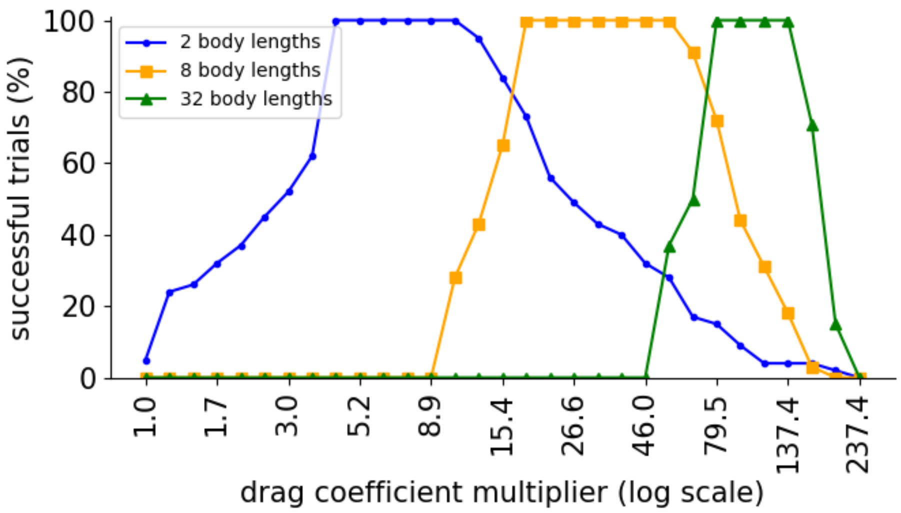


Fig. 5.8 Performance of a $10 \times 10 \times 10$ robot, using the 3D-2SP controller in environments with different drag coefficient (100 observations per setting). Three different initial distances are tested.

The 3D-5SP-V1 controller is able to always complete the task for short and medium distances. For long distances, the success rate drops to 96%. This performance drop is associated with the rotation over-shoot, when rotating along X_1^{local}. As mentioned before, the over-shoot, by itself, does not represents a problem for the controller, in terms of reaching the preferred orientation (in contrast with the 3D-2SP controller, that rotates indefinitely in this situation). However, it leads to increased rotation time. In extreme cases, the robot will be able to reach the preferred orientation, but will not have time to reach the goal.

For all travel distances, the 3D-5SP-V2 and centralised controllers are always able to complete the task. The 3D-5SP-V2 controller, due to shorter rotations, does not suffer from the high drags for short distances. For long distances, as it can rotate counter-clockwise and clockwise, it can correct any possible over-shoot. The centralised controller, due too being able to control both its rotation and translation velocities, is also able to overcome the high drag forces for short distances and eliminate any over-shoot for long distances.

In terms of completion time, the 3D-0SP-V1 controller has the best completion times, as it is always translating at maximum speed. The centralised controller is the second faster, being almost identical for medium and long distances, due to its simultaneous rotation and translation. The 3D-2SP and 3D-5SP-V1 controllers are the slowest, as the rotation and translation are separate. Of the two, the 3D-2SP is the slowest, due to a longer rotation path. The 3D-5SP-V2 controller presents completion times that are in between the centralised

Table 5.1 Average completion times for each controller and distance in seconds.

Controller	Distance 2	Distance 8	Distance 32
`3D-0SP-V1`	91.86	391.34	1496.89
`3D-0SP-V2`	144.95	589.62	1917.40
`3D-2SP`	208.95	556.51	–
`3D-5SP-V1`	173.83	465.76	3075.11
`3D-5SP-V2`	153.99	431.47	1512.17
Centralised	133.99	397.77	1499.06

and `3D-5SP-V1` controllers. Table 5.1 presents the average completion times for each controller, for each distance. As the distance increases, the ratio of average completion time between controllers gets closer to 1. As for longer distances the majority of the trial is spent translating, differences in rotation time have a smaller impact. The `3D-5SP-V1` controller in long distances is an exception to this, as it spends significant time rotating.

As before, increased sharing of information between the modules was expected to translate into performance. However, that is shown to be the case only for short distances. As the distance increases, the performance of the controllers that require no shared power lines improves, with the `3D-0SP-V2` controller surpassing the `3D-2SP` controller in medium and long distances, and the `3D-0SP-V1` controller performing better than all other controllers, including centralised (although the difference is minimal), in long distances. The fact that the `3D-0SP-V1` controller allows the robot to rotate in all directions, overcoming the overshoot present in the `3D-2SP` and `3D-5SP-V1` controllers, and translates towards the goal at maximum speed from the start, allow it to be a better solution for situations where there is enough time (distance) to complete the rotation. On the other hand, its translation becomes the limiting factor in short distances, where while still being the faster controller to complete the task, its ability to complete the task is limited to small offsets of orientation from the desired orientation.

Controllers with no shared power lines (`3D-0SP-V1` and `3D-0SP-V2`): comparison and use cases

We previously observed that for short and medium distances, between the `3D-0SP-V1` and `3D-0SP-V2` controllers, the one with the higher success rate was not the same. Additionally, from the formal analysis performed in Section 4.5.2.2, it was expected that the `3D-0SP-V2` controller would perform better for shorter distances. The results presented in Figure 5.6 show the opposite. To investigate this, the performance of these two controllers was tested

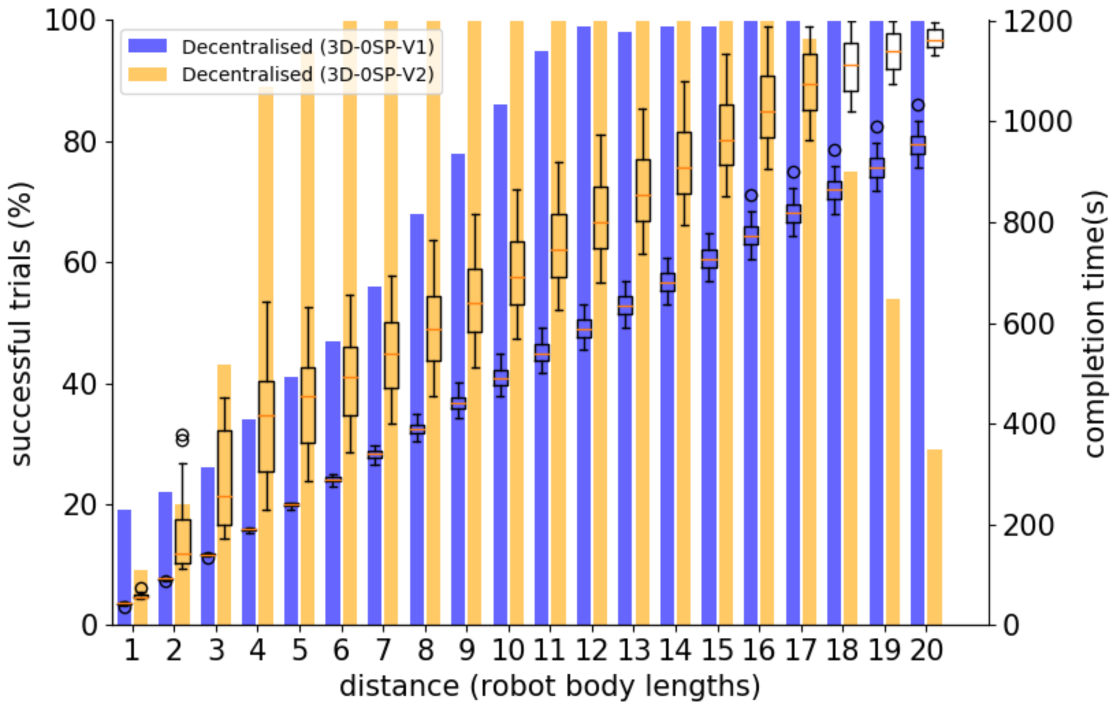


Fig. 5.9 Performance of a $10 \times 10 \times 10$ robot for multiple goal distances (100 observations per setting). The controllers are the `3D-OSP-V1` (blue) and `3D-OSP-V2` (yellow) controllers. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

for initial distances between the robot and the goal of 1 to 20 robot body lengths. For this set of distances, the time was not adjusted, that is, the time limit for each trial is 1200 s for all distances.

Figure 5.9 shows the percentage of successful trials, and the time taken to reach the goal for each of the distances considered, for the two controllers. The `3D-OSP-V1` controller presents a success rate of close to 20% for a distance of one robot body length. As the distance increases the controller's performance increases, obtaining full success rate for distances higher than 15 robot body lengths. The `3D-OSP-V2` controller starts with a success rate of close to 10%, which increases rapidly, reaching 100% success rate at a distance of six robot body lengths. For distances above 16 robot body lengths, the success rate starts dropping, due to the lack of time (in Figure 5.6 this controller is still able to complete all trials for long distances due to the adjusted time limit).

The results obtained for distances equal or larger than three robot body lengths are in line with what was expected from the theory: as the distance required to guarantee the success of

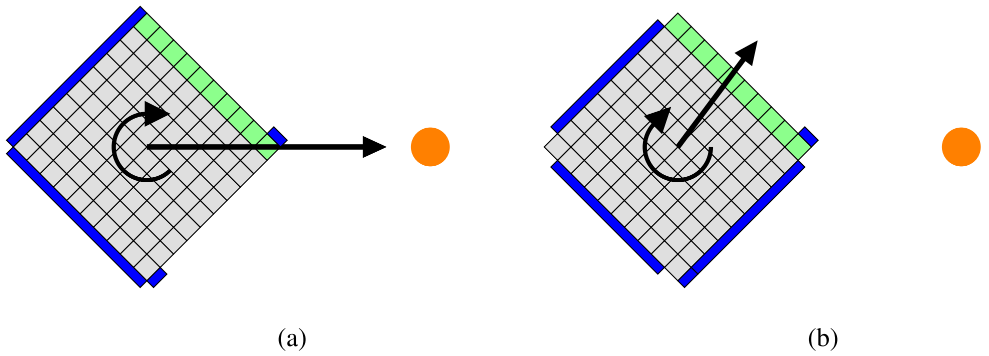


Fig. 5.10 Comparison of the net force, and resulting rotation and translation direction acting on a robot at a short distance from the goal (in orange), with a $\frac{\pi}{4}$ orientation offset between the initial and preferred orientations, using the (a) `3D-OSP-V1` and the (b) `3D-OSP-V2` controllers. The preferred face is represented by the green modules. In blue are represented the pumps that are firing for each controller. While using the `3D-OSP-V1` controller leads to a movement that will reach the goal the goal, the `3D-OSP-V2` controller generates an orbiting movement around the goal, that does not guarantee reaching it.

the controller is smaller for the `3D-OSP-V2` controller, it performs better in shorter distances than the `3D-OSP-V1` controller. As the distance increases, the performance gap gets smaller, until both controllers are always able to complete the task (if time is not considered a factor for the success of the controllers). However, for very short distances (one and two robot body lengths) the `3D-OSP-V1` controller outperforms the `3D-OSP-V2` controller.

For short initial distances between the robot and the goal, the success of these controllers is strongly tied with the difference between initial and preferred orientation[4]. For large orientation differences, neither controller is able to complete the task. For short orientation differences, both controller should be able to complete the task. However, there is a set of initial orientations, for which the difference with the preferred orientation is small[5], that do not allow the `3D-OSP-V2` controller to complete the task. The the `3D-OSP-V1` and `3D-OSP-V2` controllers generate a different translational movement in the same situation. Figure 5.10 show the direction net force acting on a robot using the `3D-OSP-V1` and the `3D-OSP-V2` controllers. The `3D-OSP-V2` controller generates a net force that is not directed towards the goal. For short distances, this leads to the robot orbiting around the goal. On the other hand, the `3D-OSP-V1` controller, will reach the goal with the preferred orientation

[4]A comparison between the "types of failure" each of these controllers produces is made in Appendix E.

[5]This set of orientations is defined as having a small difference to the preferred orientation because the `3D-OSP-V1` controller is able to complete the task successfully.

when placed in a similar situation. As a result, the 3D-0SP-V1 controller presents a higher success rate for initial distances under 3 robot body lengths.

While the 3D-0SP-V2 performs better than the 3D-0SP-V1 for short–medium distances, that only takes into consideration the success rate of each controller. As a consequence of its smaller translation velocity and being able to move the robot away from the goal, the 3D-0SP-V2 controller always takes longer to complete the trials. Additionally, the increased success rate comes at the cost of increased energy consumption.[6] As a result, while the 3D-0SP-V2 controller can be a better option for short–medium distances, for long distances, where the success rate is no longer in question, the 3D-0SP-V1 performs better.

5.2.3.4 Study on the impact of modular resolution on the performance of the controllers

From the analysis performed in Section 4.2 we know that increasing the robot's resolution increases the amount of unique net torques that can be applied to the robot. If the overall size of the robot is kept the same, and the pump's output is adjusted for the modules reduced size, the maximum torque that can be produced does not change significantly. However, due to the increased number of modules, the granularity of the movement undergoes a substantial increase. In this study, we evaluate the ability of the controllers to cope with modular robots of different resolution. While the overall robot size remains the same, the number and size of modules varies. For each module size considered, the thrust force of the pumps is re-calibrated to ensure that the net translational force, and by consequence the terminal translational velocity, remains the same.

Figure 5.11 shows the success rate of each controller for the module sizes considered. In general, the smaller the modules, the lower the torque produced by each module. This means that the terminal angular velocity decreases with the increased modular resolution. As result, the performance of the 3D-0SP-V1 and 3D-0SP-V2 controllers decreases substantially with increased modular resolution. As there is no change in the translation velocity, and the angular velocity is reduced, the robot reaches the goal before it can finish rotating. The performance of the 3D-2SP controller is more robust, but also decays for very small module sizes (ca. 0.16 cm width). This is due to the robot not enough time to reach the goal, and is caused by the longer rotation times. The 3D-5SP-V1 and 3D-5SP-V2 controllers perform

[6] A comparison between the energy consumed by each controller is performed in Appendix E.

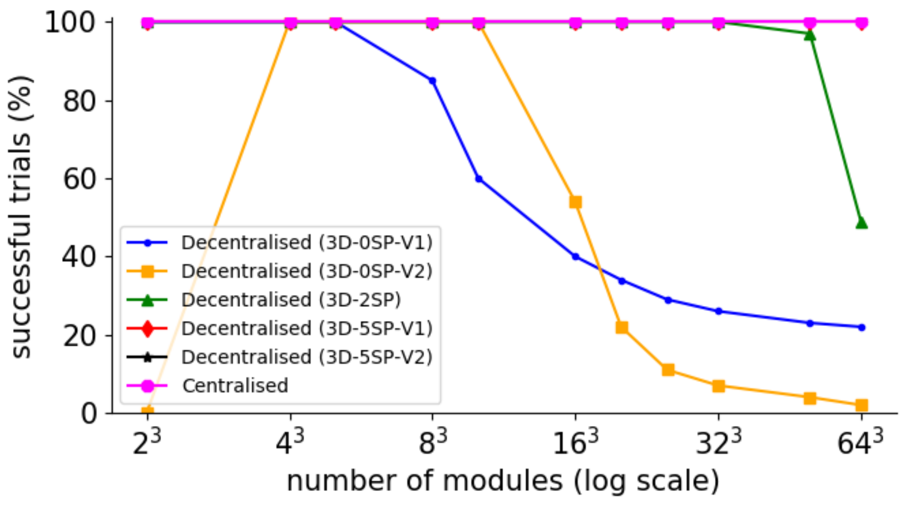


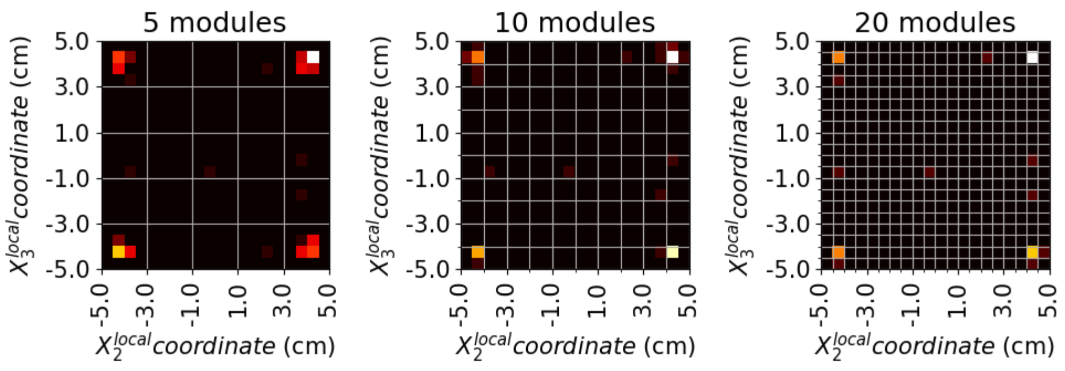
Fig. 5.11 Performance for robots of different modular resolution (100 observations per setting). While the number of modules and their size changes, the robot's overall dimensions remains the same. For details, see text.

flawlessly for the increased modular resolution, as the rotation is shorter. However, we predict that it will eventually degenerate at higher resolutions.

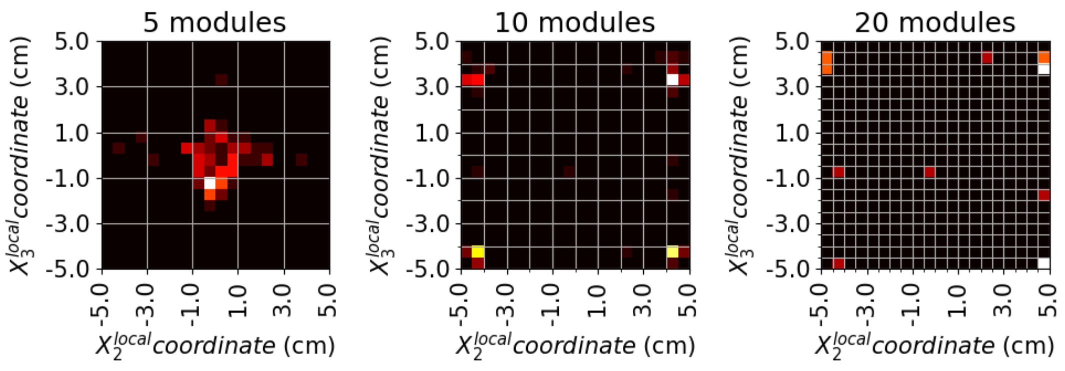
For robots with decreased modular resolution, all controllers, except the `3D-0SP-V2` controllers perform flawlessly. The `3D-0SP-V2` controller is unable too complete any trial for a robot of size 2^3. This is a limitation of the controller, as it uses the module faces that do not belong to the edges to translate. As in this configuration those do not exist, the robot is able to get to the preferred orientation, but cannot actively translate to the goal. If the robot size were to be increased, it is expected that the performance of other controllers, particularly the `3D-2SP` controller, would decrease for low modular resolution. As the torques generated by the edge modules would increase, it would simulate a situation similar to low drag coefficients.

The centralised controller performs well irrespective of the modular resolution. This controller uses additional information (relative positions of all modules), and can activate half of externally-facing pumps of a robot to support a rotation. As such, even if the modular resolution were to be increased even further, no drops in performance are expected.

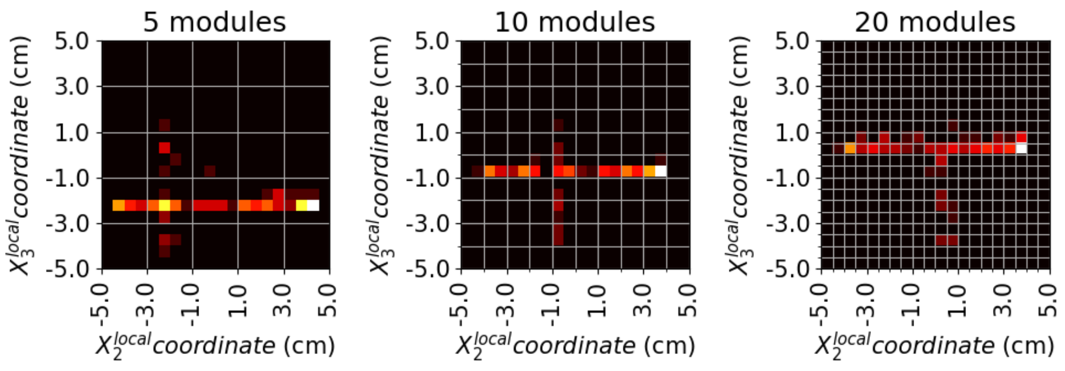
While the performance of the controllers decreases, this is only due to the time limit imposed on the simulations. From Section 4.2, we know that increasing modular resolution increases the granularity of the movement control. This can be advantageous if particularly precise movements are required. Figures 5.12 and 5.13 show, for each controller, how the



(a)

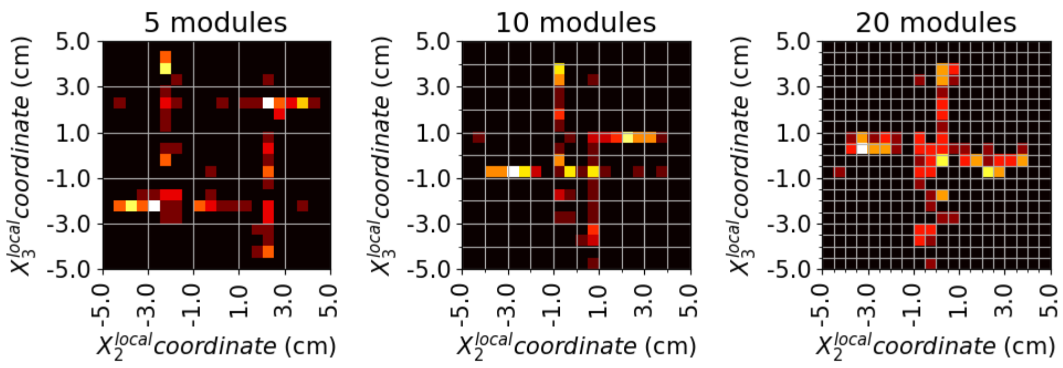


(b)

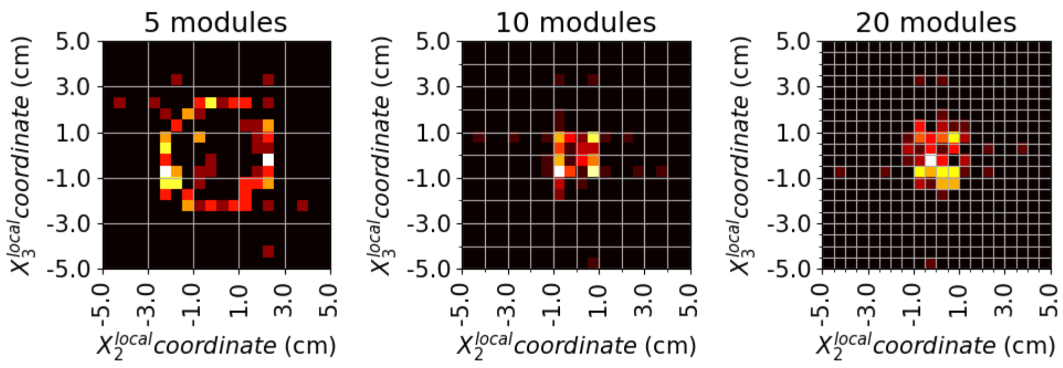


(c)

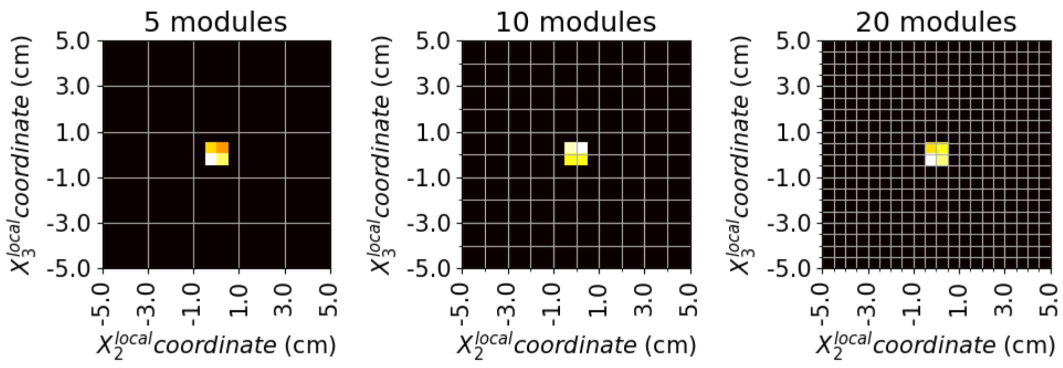
Fig. 5.12 Heatmaps showing the distribution of the contact points between the robot and the goal for successful trials (contact point in the preferred face) when using the (a) `3D-0SP-V1` controller, (b) `3D-0SP-V2` controller and (c) `3D-2SP` controller. Three robot resolutions are show, from left to right: a 5^3 robot, a 10^3 robot and a 20^3 robot.



(a)



(b)



(c)

Fig. 5.13 Heatmaps showing the distribution of the contact points between the robot and the goal for successful trials (contact point in the preferred face) when using the (a) `3D-5SP-V1` controller, (b) `3D-5SP-V2` controller and (c) centralised controller. Three robot resolutions are show, from left to right: a 5^3 robot, a 10^3 robot and a 20^3 robot.

contact points between the robot and goal are distributed along the preferred face. Three modular resolutions are considered: 5^3, 10^3 and 20^3. In general, as the resolution increases, the distribution of the contact points gets less disperse. The positions that the contact points tend to converge to depends on the controller used. The `3D-0SP-V1` and `3D-0SP-V2` controllers can rotate along two axis simultaneously. However, due to the smaller torques applied, a robot using this controller stops rotation faster. For these reasons, the contact points tend to get closer to the corners of the face (Figure 5.12a and 5.12b). For the `3D-2SP` controller, the majority of initial orientations will lead to reach the preferred orientation with a rotation along X_3^{local} (of the 25 global sensing states that lead to a rotation, in only four will the robot reach the preferred orientation with a rotation along X_2^{local}—Figure 4.15). As such, the contact points tend to a line perpendicular to the rotation axis (Figure 5.12c). The `3D-0SP-V1` controller, for similar reasons to the `3D-2SP` controller, tends to form four lines, each corresponding to one of the possible rotation axis and rotation direction(Figure 5.13a). With increased modular resolution, those lines tends to get closer to the center, for both controllers. For the `3D-5SP-V2` controller, as the robot is able to rotate along two axis simultaneously, the points distribution tends to be equidistant to the center, forming a shape similar to a circle (Figure 5.13b). Increased modular resolution decreases the dispersion of the points, bringing them closer to the center. The centralised controller is able to assign more than the edge modules to the rotation. This leads to a more accurate control of the rotation. As a result, when using the centralised controller, the same distribution of contact points is obtained, regardless of resolution.

5.2.3.5 Study on the impact of actuation noise and failure on the performance of the controllers

In all of the previous studies, all of the modules performed flawlessly. However, there are situations where such a performance is not achievable. Additionally, one of the potential advantages of modular robots is their robustness to failure and redundancy [8]. Incorrectly behaving actuators can drastically affect the robot's movement. In this study, we evaluate the impact of modular robots being exposed to actuation noise or actuation failure. Each pump chooses with probability P a uniformly random activation value, or zero, and uses the value obtained by the controller, otherwise.

Figure 5.14 shows the results for robots with modules subject to actuation noise. All controllers succeed in rejecting small disturbances. Apart from the `3D-0SP-V2` and `3D-2SP` controller, the performance is not substantially affected up to a noise level of 40%. Thereafter

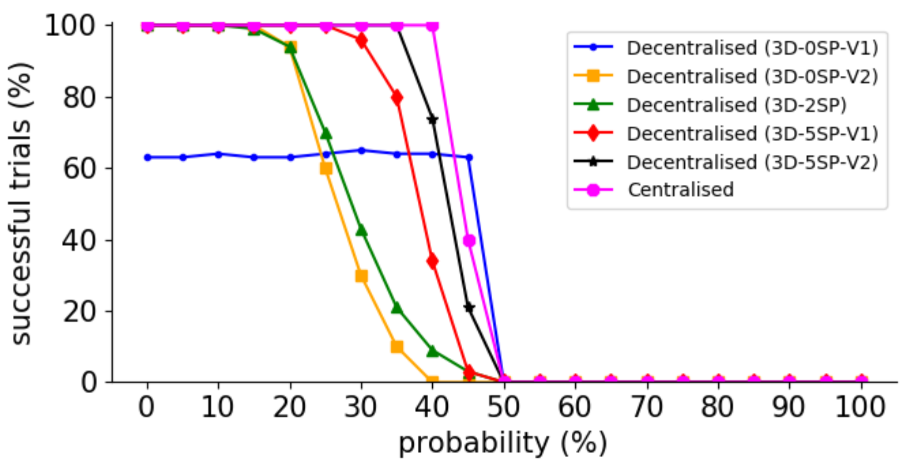


Fig. 5.14 Performance of a $10 \times 10 \times 10$ robot when subjected to different levels of actuation noise (100 observations per setting).

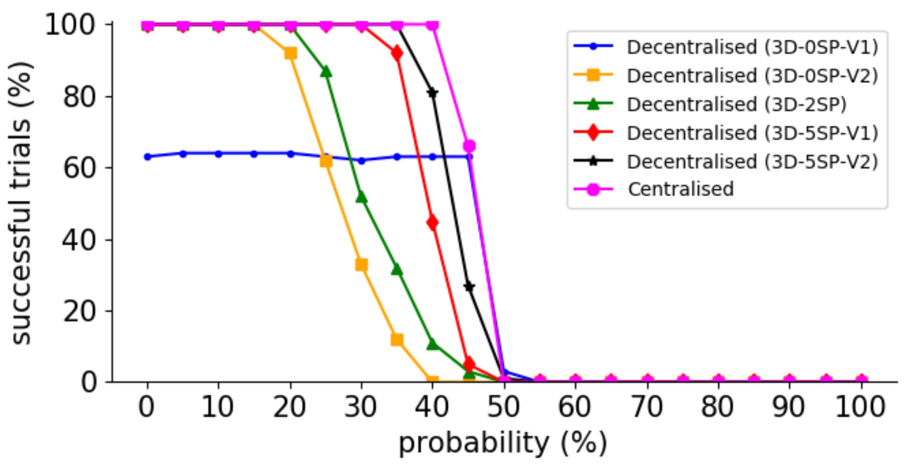


Fig. 5.15 Performance of a $10 \times 10 \times 10$ robot when subjected to different levels of actuation failure (100 observations per setting).

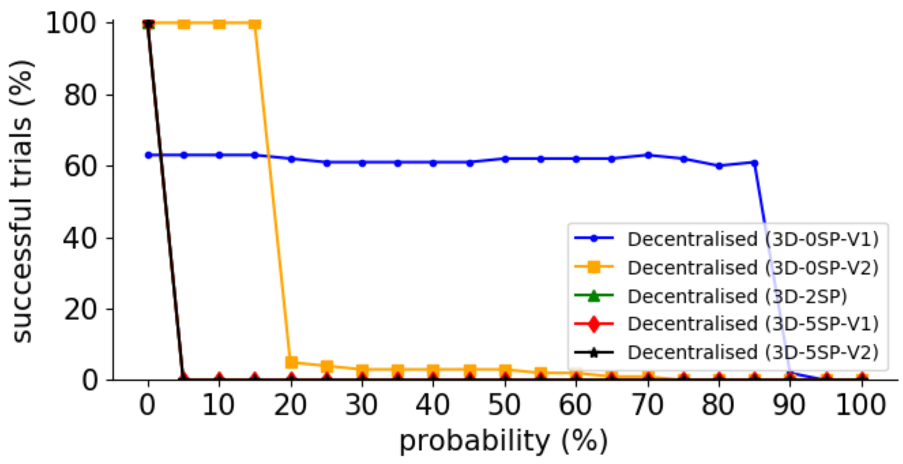


Fig. 5.16 Performance of a $10 \times 10 \times 10$ robot when subjected to different levels of sensory noise (100 observations per setting).

the performance decreases rapidly. These controllers are susceptible to lower noise levels due to their slower translation and rotation, respectively. For noise levels of 50% and beyond the robot is unable to reach the goal within the time limit. Figure 5.15 shows the results for robots with modules subject to actuation failure. The shape of the success rate curves is very similar to the ones from actuation noise. However, the success rates are slightly higher. This is due to the fact that there are no actuators acting against the movement.

5.2.3.6 Study on the impact of sensor noise and failure on the performance of the controllers

The proposed decentralised control solutions are entirely dependent on the sensing of each module to determine the actuator states. As a result, sensors are another component that can highly impact the robot's performance. In this study, we evaluate the impact of modular robots being exposed to sensing noise or failure. Each sensor reports with probability P a uniformly random value, or zero, and uses the original value, otherwise. The centralized controller is not included here, as it doesn't use sensors and its performance would not be affected.

Figure 5.16 shows the results for robots with modules subject to sensor noise. The controllers that use shared power lines are highly susceptible to sensor noise. As long as one sensor belonging to a face other than the preferred direction gets a false positive reading,

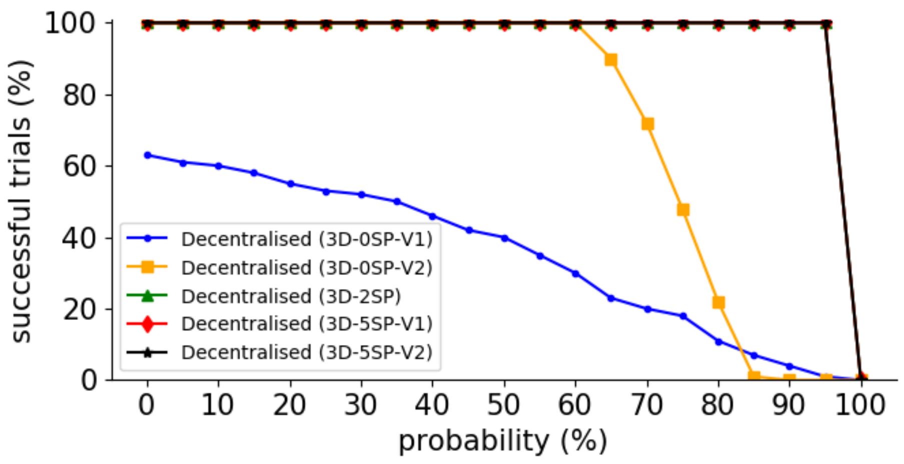


Fig. 5.17 Performance of a $10 \times 10 \times 10$ robot when subjected to different levels of sensory failure (100 observations per setting).

the corresponding power line is pulled up, causing the robot to rotate. This means that a single sensor malfunction can stop the robot from completing the task. On the other hand, the controllers that do not use shared power lines are less susceptible to sensor noise. As the modules do not share any information, false readings cannot propagate. The `3D-0SP-V1` controller performs robustly with up to 85% noise. Above this value of sensor noise, the robot becomes unable to reach the goal. The `3D-0SP-V2` controller is able to perform flawlessly for sensor noise values up to 15%. While the rotation component of the movement should suffer the same as the `3D-0SP-V1` controller, the active translation behaviour is dependent on a single face detecting the goal. As pumps belonging to all faces other than the preferred face are active regardless of sensor readings, false positive readings in the preferred face cause the robot to actively translate towards the wrong direction.

Figure 5.17 shows the results for robots with modules subject to sensor failure. In contrast with the sensor noise, the controllers that use shared power lines are very robust to sensor failure. These controllers effectively only need a total five sensors, one per face other than the preferred, to operate correctly. As a result, they can easily deal with sensor failure. The `3D-0SP-V1` controller, on the other hand, drops in performance as the the sensor failure level increases. This comes from the fact that sensor failure impedes both the robot translation and rotation, meaning that it takes longer to both reach the target and the preferred orientation. The `3D-0SP-V2` suffers from the same effect, although to a smaller scale. For this reason, it

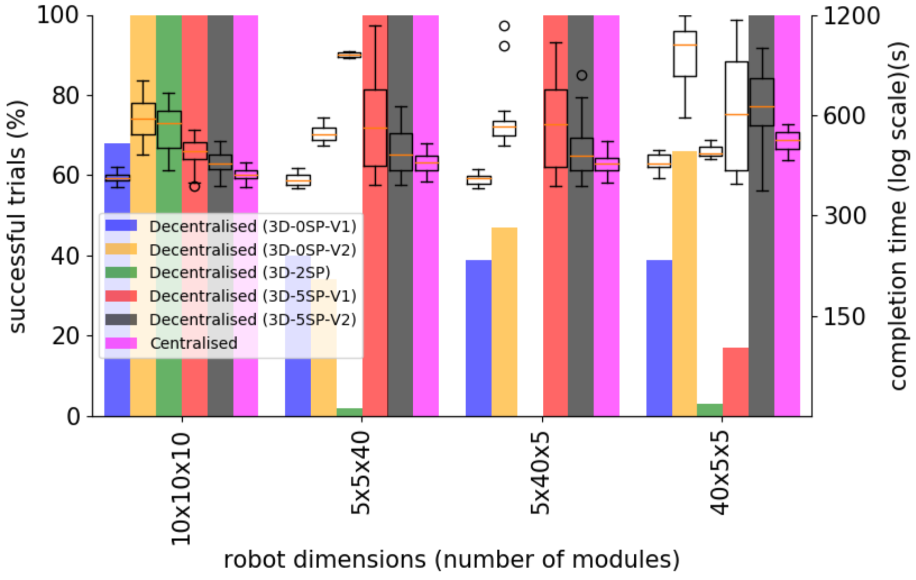


Fig. 5.18 Comparison of the performance of a cubic robot (10 × 10 × 10) with a thin and long robot (5 × 5 × 40). Simulations were performed with the long component along all three axis. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

is able to perform flawlessly up to sensor failure values of 60%, at which point it becomes unable to complete all trials.

5.2.3.7 Study on the impact of robot sizes on the performance of the controllers

In Section 5.2.3.4, we studied the effect of modular resolution on the performance of the controllers. However, while the size of the modules was reduced, the overall size of the robot remained the same. In this section, we study the impact that different robot dimensions can have on the performance of the controllers, using cuboid robots, in order to maintain convexity.

We consider three different sets of robot dimensions: 5 × 5 × 40, 5 × 10 × 20 and 5 × 14 × 14, and compare the performance obtained using the proposed controllers, with the performance of the 10 × 10 × 10. For first two sets of dimensions, the number of modules is the same as the robot used in the previous studies. The last set uses less modules, of a total of 980 (98 % of the volume of the 10 × 10 × 10 robot). The aim is to have robots with two identical dimensions and a larger/smaller third dimension. All possible combinations of the considered dimensions, over the three axes are tested.

Figure 5.18 presents the results for the different arrangements of the set of dimensions $5 \times 5 \times 40$, and compares them with the results obtained for the $10 \times 10 \times 10$ robot. When using this set of dimensions, the torque produced along the axis of longer dimension is approximately 1.8 times larger than for a $10 \times 10 \times 10$ robot, with the torques in the other directions 1.2 times larger. This leads to faster rotations and larger overshoot for robots using the `3D-2SP` controller, which justifies its poor performance for all combinations of the dimensions considered. The `3D-5SP-V1` controller suffers from a similar problem, when the longer dimension is along the X_1^{local} axis, as it can overshoot while rotating along this axis, justifying its poor performance for robots of dimensions $40 \times 5 \times 5$.

When considering translation, if the longer direction is along the X_1^{local} axis, the terminal velocity drops to half of what it is for a $10 \times 10 \times 10$ robot, while it is increases by 40 % for the other two cases. This justifies the increased success rate for the `3D-0SP-V2` controller, as it has additional time to complete the rotation.

The performance of the `3D-0SP-V1` controller remains largely unchanged over the different dimension arrangements, as the changes in rotation velocities balance the changes in translation velocity. As the controller does not generate rotations along the X_1^{local} axis, when this is the larger dimension, there is only a small increase in rotation velocity, and a reduction of half of the translation velocity. If the larger dimension is along any other axis, even though the robot benefits from an significant increase in rotation velocity, it is only along one of the rotation axes, while its translation velocity is increased reducing the time it has to complete the rotation.

The `3D-5SP-V2` controller, demonstrates similar performance for all configurations, only showing an increase of completion time for the $40 \times 5 \times 5$, largely due to its decreased translation velocity.

Figure 5.19 presents the results for the different arrangements of the set of dimensions $5 \times 10 \times 20$, and compares them with the results obtained for a robot of dimensions $10 \times 10 \times 10$. When using the set of dimensions $5 \times 10 \times 20$, the torque produced along the axis of longer dimension is approximately 1.4 times larger than for a $10 \times 10 \times 10$ robot, with the torque along the axis of the second longest direction being approximately 1.3 times larger, and 0.8 times smaller along the remaining axis. This leads to faster rotations along two axis and slower along the last, generating both larger and smaller overshoots depending on the axis of rotation for robots using the `3D-2SP` controller. As this controller likely to perform multiple rotation rotations along the X_1^{local} in a single trial, and the dimension of the robot along this axis increases, its performance drops, due to increased overshoot. The `3D-5SP-V1`

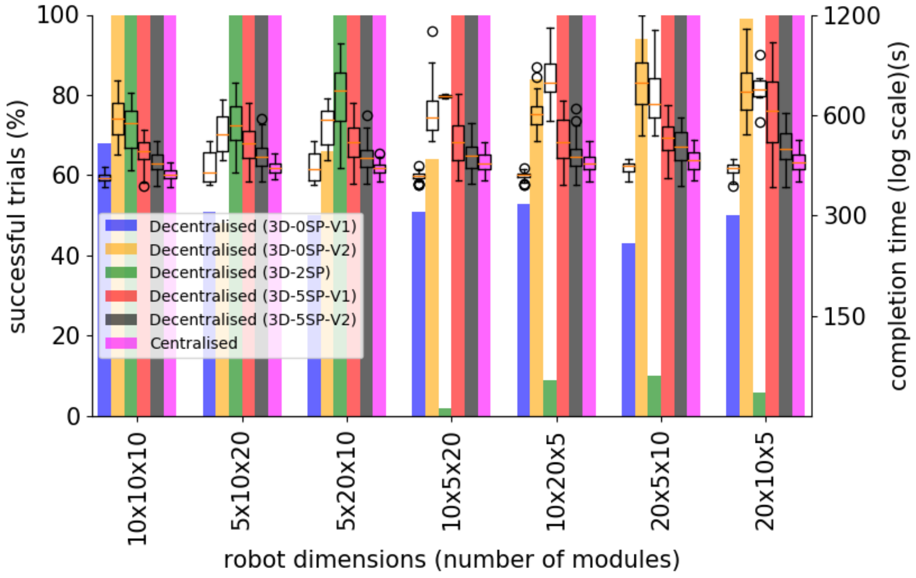


Fig. 5.19 Comparison of the performance of a cubic robot ($10 \times 10 \times 10$) with a robot with three different dimensions ($5 \times 10 \times 20$). Simulations were performed with all possible combinations of dimensions along all three axis. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

controller suffers from a similar problem, when the longer dimension is along the X_1^{local} axis, as it can overshoot while rotating along this axis, justifying its increased completion time for robots with a larger dimension along the X_1^{local} axis.

When considering translation, the terminal velocity 0.7, 1 and 1.4 times of what it is for a $10 \times 10 \times 10$ robot, as the robot dimension along the X_1^{local} axis increases. As with the previous case, this justifies the increased success rate for the `3D-0SP-V2` controller, as it has additional time to complete the rotation.

The performance of the `3D-0SP-V1` controller remains constant for all cases other that configurations with the larger dimension along X_1^{local}. In this case, even though its translation is slower, giving it more time to complete the rotation, its rotations are also slower.

The `3D-5SP-V2` controller demonstrates similar performance for all configurations, only showing an increase of completion time for the configurations with the longest dimension along the X_1^{local} axis, largely due to its decreased translation velocity.

Figure 5.20 shows the results for robots of dimensions $5 \times 14 \times 14$, which follow a similar trend to the results previously presented for all controllers but the `3D-0SP-V1`. This controller shows increased performance with the increases in robot size along the X_1^{local} axis. For this

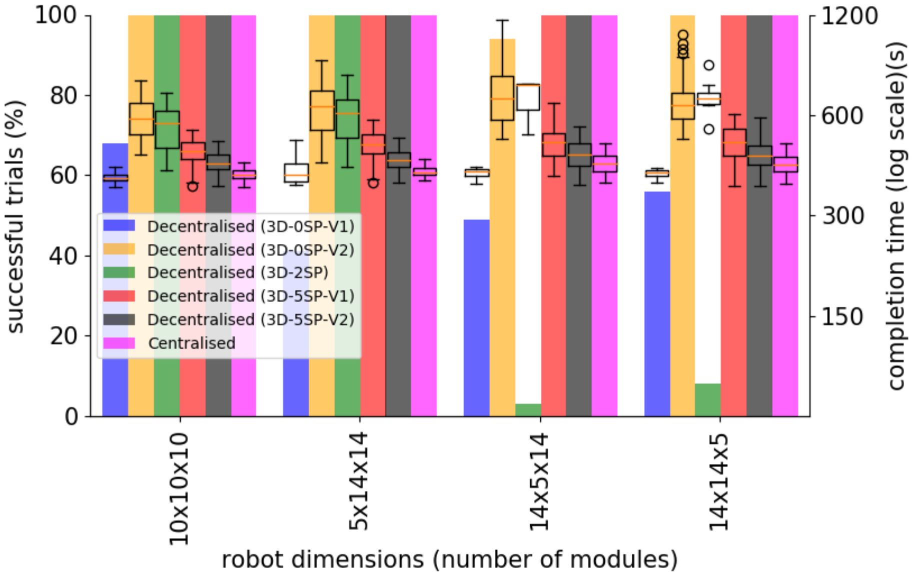


Fig. 5.20 Comparison of the performance of a cubic robot ($10 \times 10 \times 10$) with a short and wide robot ($5 \times 14 \times 14$). Simulations were performed with all possible combinations of dimensions along all three axis. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

particular robot configuration, the increases in rotation velocity are higher than for the cases presented above, while presenting similar translation velocity reductions. Additionally, as the mass of the robots is smaller (and by extension, its inertia), it accelerates faster. As a result, this controller is capable of completing more trials in this configuration.

From the analysis of the results presented in this section it is possible to see that controllers with zero or five shared power lines are capable of good performance, for robots of different sizes, the 3D-2SP controller deals poorly with changes in the robot dimensions. Additionally, it is shown that the controllers used with robots using the same set of dimensions, but with a different order along the axis, can have (minor) differences in performances.

5.2.3.8 Study on performance of the controllers for randomly shaped robots

In Section 4.6, it is shown that while there are some non-convex shapes for which the controllers are able to complete the task, that is not the case for all non-convex shapes. In this chapter we test the performance of the proposed controllers for robots of random shape. These robots are initialized with a single module, with additional modules being added one at a time. A new module is added to a face that is chosen randomly from all available external

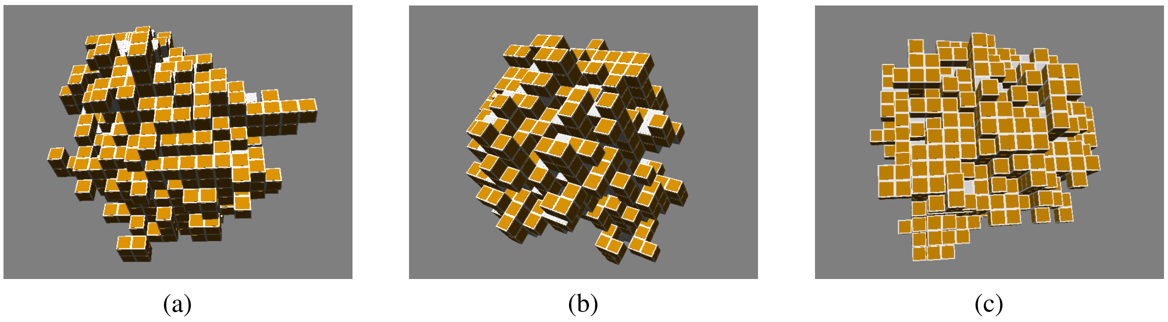

(a) (b) (c)

Fig. 5.21 Examples of robots with random shapes generated with 1000 modules.

module faces on the robot. This repeats until the robot consists of the desired number of modules. So that the mass of the robot does not change, robots of random shape use 1000 modules. Examples of robots with random shapes are presented in Figure 5.21.

Figure 5.22 shows the performance of the controller using robots of random morphologies, and compares it with the performance of a cubic robot of the same mass/volume. From the results presented, it is possible to see that some controllers are unable to complete the task. The `3D-0SP-V2` controller requires at least three modules side by side to be able to perform translations. By the nature of random morphologies, it is not possible to guarantee that this requirement is met. As a result, even though it is possible for a robot to reach the desired orientation, it is unlikely that a robot of random morphology is able to move towards the target using this controller.

In Section 4.6, we mention that a robot in a random configuration might be able to detect the goal from two opposite directions simultaneously. As a result, both the `3D-2SP` and `3D-5SP-V1` controller are unable to complete the task, as detecting the goal with multiple faces leads to continuous rotation along one of the axes.

From this, the only controllers capable of completing the task are the `3D-0SP-V1` and the `3D-5SP-V2` controllers. When compare with a $10 \times 10 \times 10$ robot, the `3D-0SP-V1` controller is capable of completing all trials, although with an increased completion time. Due to the random morphology, the number of corner modules in the robot increases, which leads to more modules contributing to rotation, and less to translation. As a result, not only is the rotation faster, there is more time to complete it, as the translation is slower.

On the other hand, the `3D-5SP-V2` controller performs worse, when compared with an equivalent cubic robot. This controller required small modifications to allow it to translate when detecting the goal from two opposing sides. This small modification made it that the robot could reach the goal. However, due to its random morphology, in situations where the

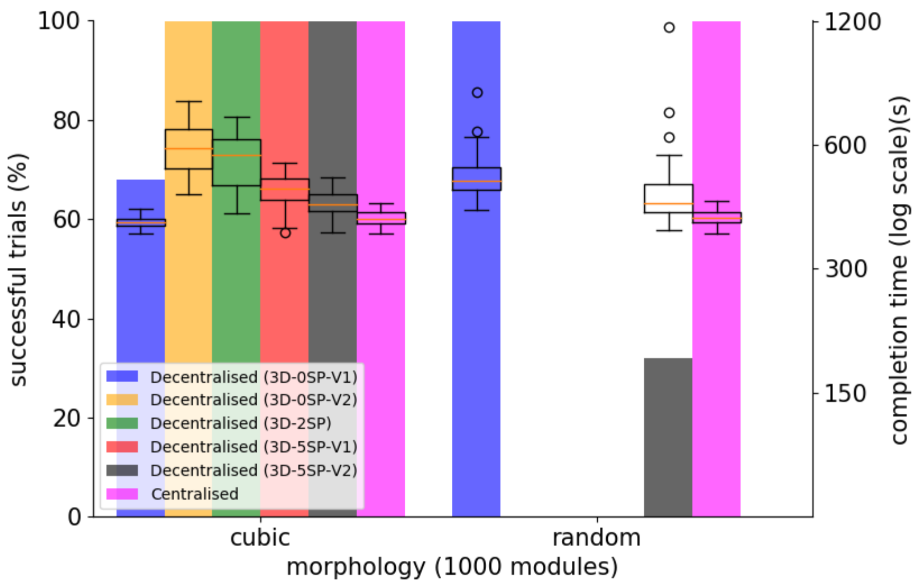


Fig. 5.22 Performance of a $10 \times 10 \times 10$ robot and randomly shaped robots with 1000 modules (100 observations per setting). Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

robot is very close to the goal, without colliding, one of the two opposing faces that detected the goal can be occluded by the robot itself. As a result the robot rotates, it can collide with the goal with the incorrect face, leading to a failed trial. This leads to the controller, although being able to complete the task, having a poor performance. Using a different criteria for ending the simulation and face detection could resolve this issue. For example, if the trial was assumed as finished as soon as the goal enters the smallest axis aligned bounding box that can contain the robot, and the face of contact was determined from this bounding box, we expect that the performance of this controller would improve.

5.2.3.9 Study on the impact of sensing angle on the performance of the controllers

All the simulation studies presented in this chapter assume that the sensors used to detect the goal have a half-plane field of view. However, as discussed in detail in Section 3.5.3, this is not representative of the hardware. In Chapter 3 we propose a hardware implementation of the MHP robot. Upon testing, it was determined that the sensors used in the hardware have a field of view of 64.1 ° (tested at the maximums detection distance used in physical experiments). In this section we study the performance of the controllers, using the field of view of the physical hardware.

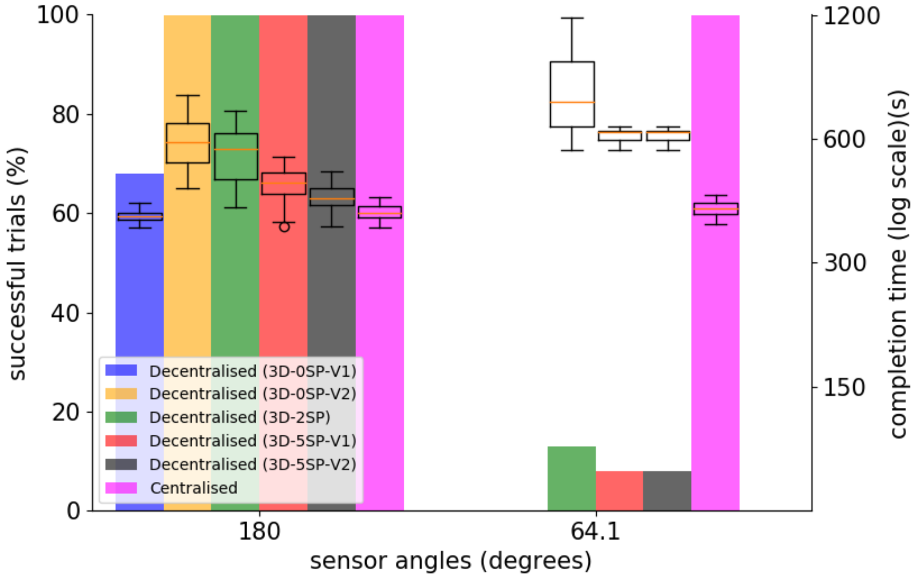


Fig. 5.23 Performance of a $10 \times 10 \times 10$ robot using a sensor field of view of 180 ° (half-plane) and 64.1 ° (measured frm real hardware). Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

Figure 5.23 shows the performance of all controllers for the two fields of view considered. The controllers that use no shared power lines are completely unable to complete the task when using a reduced field of view. When the robot is sufficiently far away from the goal, both controllers are able to detect it. However, as the robot gets close to the goal it can enter a dead sensing zone, i.e. a situation where no face can detect the target. In this situation, both controllers are unable to generate movement, as all faces are pushing the robot. As a result, in these situations the robot moves only due to the inertia generated by the previous movement preformed, until a face is able to detect the goal. This leads to very slow progress, not allowing the robot to reach the goal within the time limit for the simulation.

The controllers that use shared power lines perform slightly better, although not satisfactory, allowing the robot to reach the goal with the correct face in a couple of trials. When using these controllers, if the robot is in a dead sensing zone it will translate in the direction of the preferred face (even if it is not in the correct orientation). As a result, the robots are able to leave dead sensing zones more easily, and reach the goal. When using the half-plane field of view the robot only has to perform a single rotation using these controllers, whereas the reduced field of view, the robot goes through multiple rotations, intercalated with translations. As a result, even when getting close to the goal the robot might not be in the correct orientation, and reaches the goal with the incorrect face.

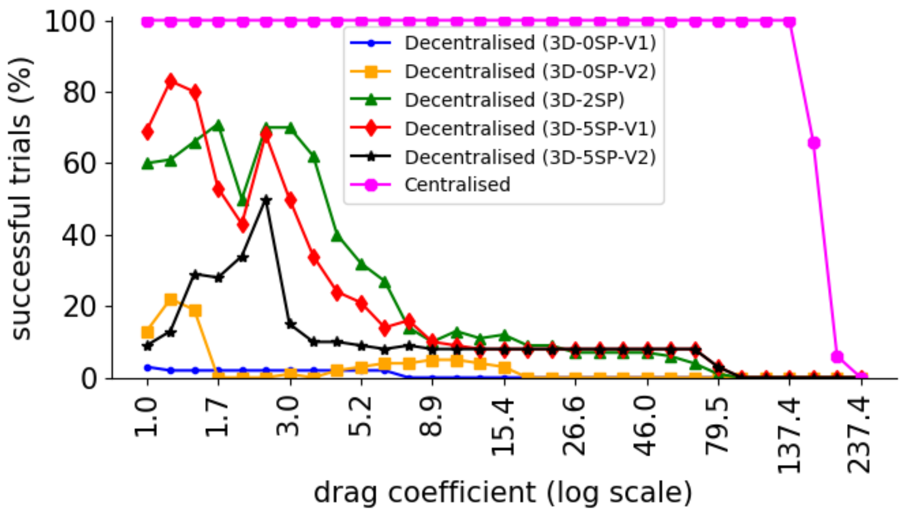


Fig. 5.24 Performance of a $10 \times 10 \times 10$ robot using a sensor field of view of $64.1\,^{\circ}$ (measured from real hardware) in environments with different drag coefficients (100 settings per value). Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

While the performance of the controllers is very poor with a reduced field of view, a lower drag coefficient value could allow it to increase. As when in a dead sensing zone the robot moves only through inertia, a high drag coefficient makes these movements decrease in a short period of time. On the other hand, a lower drag coefficient would allow these movements too continue for longer, reducing the time a robot is in a dead sensing zone. While this, previously defined as overshoot, was a negative factor when using a half-plane field of view, with a reduced field of view, it can be beneficial to the performance of the controllers. Figure 5.24 shows the success rate for the controllers, when using a reduced sensing field of view, for multiple drag coefficient values.

As expected, the performance of the controllers improves with the reduced drag coefficient values. The controllers that present the most significant improvement are the `3D-2SP` and `3D-5SP-V1` controllers. While these controllers are usually the worst performing, as they can rotate in a direction for at least a single axis, in this case this limitation is what leads them to perform better. While in all other studies presented, rotation overshoot was a problem, in this study it is an advantage. Thus, the controllers that suffer from overshoot perform better. While some overshoot is still present in the `3D-0SP-V1`, `3D-0SP-V5` and `3D-5SP-V2` controllers, the fact that it can be corrected leads to a reduced rotation inertia from the rotation performed last, increasing the time spent in a dead sensing zone. As a result, the controllers that do not use shared power lines are the worst performing as they spend a significant amount of time moving just by inertia.

5.2.4 Discussion

In the previous section we presented multiple simulation studies that aimed to evaluate the performance of the proposed controllers, and put into perspective by using a controller from the literature. Through the analysis of the results presented it is possible to present a comparison of the different controllers, where the trade-offs between complexity and performance can be detailed.

With the exception of the study with robots of random morphologies, the controller that consistently demonstrated the best performance was the `3D-5SP-V2`. This controller presented results comparable to the centralised controller from the literature[7], while requiring less computational resources. When compared with the `3D-5SP-V1` controller, the `3D-5SP-V2` controller performs equally or even better, and the computational resources used for both controllers are the same. As a result, the `3D-5SP-V2` controller is more flexible and adaptable, making it a better choice. Exceptions to this are situations involving sensor noise, where no controller that uses shared power lines can complete the task, and for robots with small sensing angle, as in this case controllers that result in a large overshoot have the advantage.

The `3D-0SP-V1` is the controller capable of best performance, while also requiring the minimum computational resources of the proposed controllers. While this controller is less flexible than the `3D-5SP-V2` controller, as it is not in all conditions that it can perform optimally, it frequently outperformed the other controllers that require shared power lines. Additionally, while in certain cases it was outperformed by the `3D-0SP-V2` controller, the `3D-0SP-V1` controller does not impose requirements on the shape of the robot. Additionally, the `3D-0SP-V2` controller has additional costs in terms of energy expenditure and a larger average completion time. While the `3D-0SP-V1` controller is better suited for situations where sensing noise is involved, it performs even worse than the `3D-5SP-V2` controller when the sensing angle is reduced. As a result, it is also inadequate to deal with these situations.

[7]The `3D-5SP-V2` controller was outperformed by the centralised controller in the sensing related studies, but since the centralised controller does not use goal detection sensing, these studies were not considered for this particular comparison.

The remaining controller that makes use of shared power lines, the `3D-2SP` controller, was expected to be a solution with a performance between the `3D-0SP-V1` controller, that requires no shared power lines, and the `3D-5SP-V2` controller, that makes use of five shared power lines. However, this controller's performance, while excellent in certain scenarios, was frequently outperformed by the `3D-0SP-V1` controller. Additionally, its performance is highly dependent on the environment, making it the least flexible of all proposed controllers. The only situation in which the `3D-2SP` controller outperforms all other proposed controllers is when considering reduced sensing angle, due to the high overshoot generated.

As a result, from the analysis of all studies performed, in a situation where resources are available, or guaranteed success is needed, the best controller to use is the `3D-5SP-V2` controller[8]. In situations where resources are scarce, and there is enough manoeuvring space, the `3D-0SP-V1` controller can be the better option. The only exception to this is if sensing noise is expected, where the `3D-5SP-V2` controller is not a viable solution (as well as any of the controllers that make use of shared power lines). Additionally, for robots with a sensing angle smaller than $180\,^{\circ}$, neither controller performs adequately, with the `3D-2SP` controller being a better choice for these situations.

5.3 Physical Experiments

In the previous section, the proposed pose controllers were validated and studied in simulation, in 3D environments. However, no validation was performed for the pose controllers proposed for 2D movement. In this section we present a set of physical experiments, using the MHP robots developed in Chapter 3. The aim is to validate the performance of the pose controllers proposed in Section 4.4 in a physical system and compare their performance with the results obtained in simulation.

5.3.1 Experimental Setup

To validate the controllers in a physical platform we perform experiment using four modules arranged in a 2×2 configuration, with the controllers `2D-1SP` and `2D-2SP`. The `2D-0SP-V1` and `2D-0SP-V2` controllers are not tested as the `2D-0SP-V2` requires at least three modules

[8]This can be correct for specific non-convex morphologies, although not all, as showed in Figure 5.22

per dimension, and the theoretically expected trajectory of the2D-0SP-V1 controller requires a larger water tank[9].

The experimental setup is identical to the one presented in Section 3.5.1. At the beginning of each trial the robot starts at one end of the water tank, approximately equidistant to the tank's long boundaries (see Figure 3.7). The robot orientation is chosen from a set of four orientations, with a rotation of -90 °, 0 °, 90 ° and 180 ° from the preferred orientation. For each controller tested, five trials are performed for each orientation.

Each trial lasts for a duration of 90 seconds. A trial is deemed successful if, before the time limit is reached, the robot's centroid reaches the finish line with the preferred orientation. For the purpose of the experiments, a robot is assumed to be in the preferred orientation if the angle formed between the line connecting the robot's centroid and the goal and the normal of the preferred face is smaller than 45 °[10].

5.3.2 Controller Implementation

The 2D-1SP and 2D-2SP controllers require the use of shared power lines. However, the physical robots do not possess the capabilities that allow the existence of the shared power lines. While the shared power lines could be virtually created, through the use of memory and communication, this would require the robots to be able to communicate more than one bit of information. Although possible, once in water, the communication becomes unreliable, not allowing the correct behaviour to be used by the robot. To circumvent this issue, the four modules are connected to an Arduino micro board, mounted on top of the robot, that acts as the shared power lines.

The implementation of the controllers is presented in Algorithm 4. At the beginning of each cycle, each module reads its light and connection sensing (lines 8–11). From the sensing readings each module evaluates the state of the power line(s), as if it were the only module on the robot, using the function described in Section 4.4.1. This value is then sent to the external Arduino (lines 12–14), that by using the power line(s) state(s) of all modules, determines the state(s) of the shared power line(s), which is then sent to each module (lines

[9]The expected trajectories and justification are presented in Appendix F.

[10]In some situations, the data obtained through the tracking software indicates an angle higher than 45 °, and the trial is still counted as successful. The occurs because the robot is not centred in the Y axis in the water tank. As a result, even though the angle obtained though the tracking software is higher than 45 °, the angle formed between the line connecting the robot's centroid and the goal and the normal of the preferred face is still smaller than 45 °.

Algorithm 4 Implementation of controller requiring shared power lines

```
1: $c_j$, $d_j$ and $p_j$ correspond to the states of the sensors and pump of face $k$
2: $B_k$ corresponds to an internal evaluation of the power line, sent to the external arduino
3: $b_k$ corresponds to the state of the $k$-th shared power line
4: $n_{faces} = 4$
5: $t_{cycle} = 100$                                   ▷ Time in milliseconds.
6: loop
7:     $t_1 \leftarrow$ time
8:     for $j \leftarrow 1, n_{faces}$ do                 ▷ Go through all faces.
9:         $c_j \leftarrow$ Connection sensor reading      ▷ Get connection sensor readings.
10:        $d_j \leftarrow$ Goal sensor reading            ▷ Get light sensor readings.
11:    end for
12:    for $k \leftarrow 1, n_{powerlines}$ do            ▷ Evaluate internal state of power lines
13:        $B_k \leftarrow g_k(\mathbf{c}, \mathbf{d})$
14:    end for
15:    for $k \leftarrow 1, n_{powerlines}$ do            ▷ Read external power lines
16:        $b_k \leftarrow$ read from arduino
17:    end for
18:    for $j \leftarrow 1, n_{faces}$ do                 ▷ Go through all faces.
19:        $p_j \leftarrow g_b(\mathbf{c}, \mathbf{d}, \mathbf{b})$
20:    end for
21:    $t_2 \leftarrow$ time
22:    if $t_2 - t_1 < t_{cycle}$ then    ▷ If the loop was faster than the minimum cycle time, wait.
23:        wait($t_{cycle} - (t_2 - t_1)$)
24:    end if
25: end loop
```

15–17). Using the policies presented in Section 4.4.1, and using the value(s) for the shared power line(s) sent by the external arduino, the activation state of the pumps is determined (lines 18–20). Due to the limited response times of the pumps, we ensure pump states are only updated up to ten times per second (lines 21–24).

5.3.3 Results

In this section we present and discuss the results obtained in the physical experiments. The 2D-1SP and 2D-2SP controllers had a success rate of 40 % (8 out of 20) and 100 % (19 out of 19). The analysis of the results is based on visual analysis of the recordings and tracking of the robot's movement using visual tracking software. During the analysis, one of the trials for the 2D-2SP controller had to be removed, due to a recording error. The overall higher

Table 5.2 Summary of the results for completion time for the 2D-1SP and 2D-2SP controllers

Completion Time	2D-1SP	2D-2SP
Minimum [s]	17.8	16.7
Average [s]	25.8	32.1
Maximum [s]	38.6	46.1

success rate of the 2D-2SP controller is expected, as the pumps used were not calibrated, leading to translation during rotation. As a result, since the 2D-2SP controller can correct overshoot, it reduces the time for rotation, allowing the robot to reach the finish line with an orientation close to the desired orientation.

For the successful trials of the 2D-1SP controller, four have an initial orientation of 0°, two trials with initial orientation of 180° and a single successful trial for the initial orientations of -90° and 90°. The failed trial with initial orientation of 0°, failed due to excessive robot rotation, on collision with the wall. For the initial orientation of 180°, the robot ended moving away from the goal, and latching to a corner of the water tank, for the remaining duration of the trial. The failed trials for the remaining orientations were mostly caused by the inability of the controller to recover from overshoot, allowing it to rotate until the end of the trial. While some of the reasons for trial failure are controller dependent (single direction of rotation), some other reasons can be presented.

The size of the water tank might not be adequate for these particular experiments, as collisions with the walls lead to changes in orientation of the robot. As a result, performing the experiments in a larger environment might improve the observed controller performance. An additional reason could be the lack of calibration of the pumps. While it is interesting to demonstrate that even with non-calibrated pumps the controllers are somewhat capable of completing the task, the lack of calibration introduces translation movements during the rotation period. This leads to the robot having a reduced time to reach the desired orientation and colliding with the walls of the water tank, which can interfere with its rotation[11]. Lastly, as shown through simulations, the small sensing angle of the sensors creates dead sensing angles, which limit robot performance.

Figure 5.25 shows the distribution of time taken for successful trials of both controllers. From the simulation results presented in the previous section, and assuming the relationship between the 2D-1SP and the 2D-2SP controllers is similar to the relationship between the 3D-2SP and the 3D-5SP-V1 controllers, it was expected that the time taken for completion

[11]Note that even with calibrated pumps the water tanks might be limiting factor as the robots will not necessarily move in a strait line due to existing overshoot.

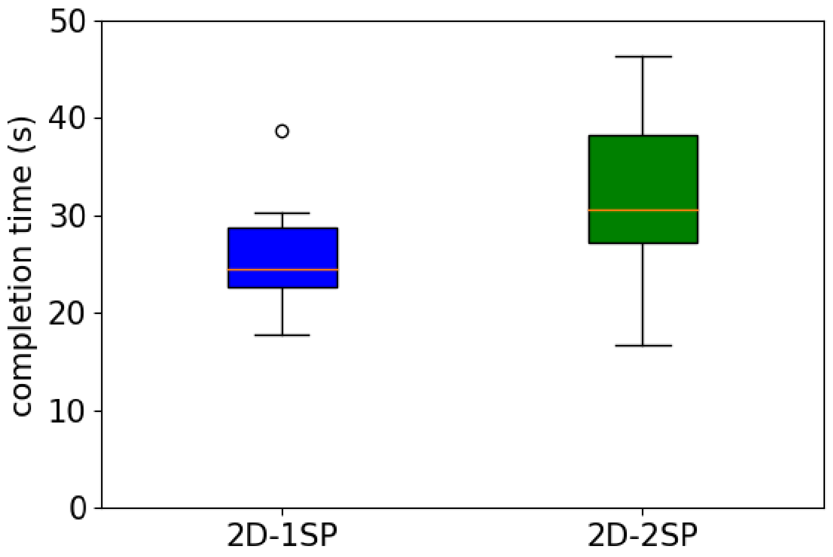
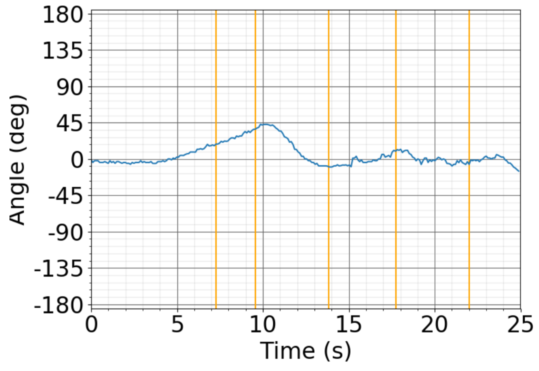


Fig. 5.25 Time taken for a physical 2×2 MHP robot to move 60 cm towards the goal, with a predefined orientation, using the 2D-1SP and 2D-2SP controllers. Successful trials only; the boxes on the left and right represent 8 and 20 trials, respectively.

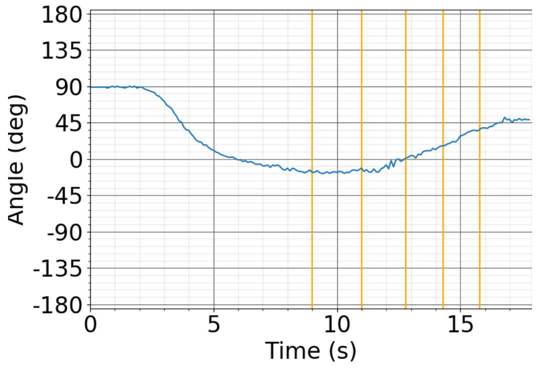
of the trials would be shorter for a robot using the 2D-2SP controller than for a robot using the 2D-1SP controller. However, when looking into the study related to the sensing angle, the 3D-2SP controller outperforms the 3D-5SP-V1 controller, for low drag coefficients[12], in terms of success, while having a larger completion time. The results obtained and presented in Figure 5.25, contradict these expectations, with the 2D-2SP controller having a larger success rate, with a larger completion time. Table 5.2 summarises the results presented in Figure 5.25. While the minimum completion time is smaller for the 2D-2SP controller, both the average and maximum completion times are larger. One explanation for this is the size of the sample used for calculation of these values. Additionally, since the actuators are not calibrated, when changing between rotation directions, a robot can move backwards slightly, which increases the time of the trial.

Figures 5.26 and 5.27 show angle variations, over time, for a sample of four trials of each controller, one for each initial orientation. These trials were chosen for their success and behaviour best matching the one expected for each controller. Results for the remaining trials, including unsuccessful, are presented in Appendix G. Each vertical line corresponds to a translation of 10 cm towards the goal. As mentioned above, these results were obtained

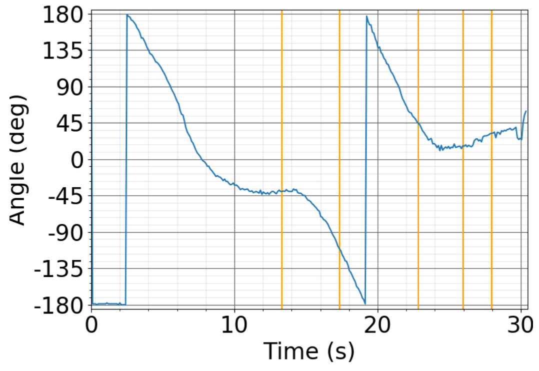
[12]Conditions closer to the experimental setup.



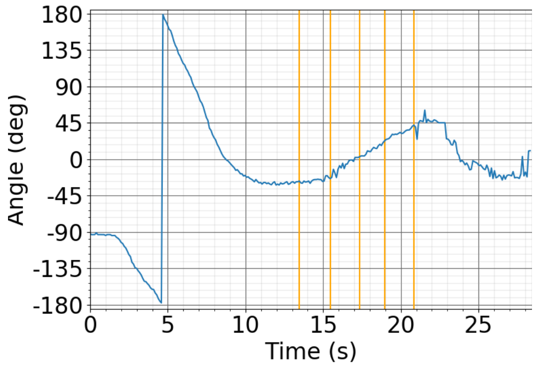
(a)



(b)



(c)

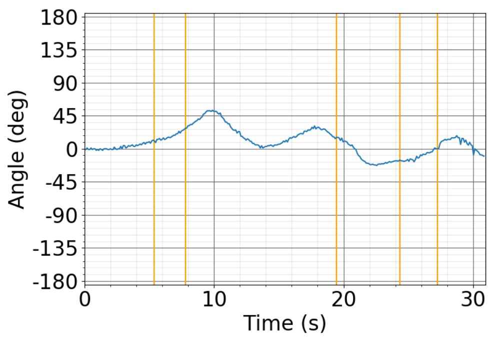


(d)

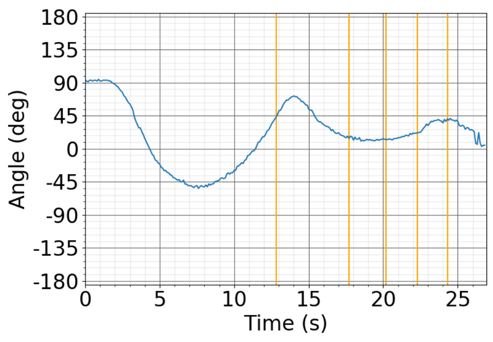
Fig. 5.26 Examples of the changes in angle during a trial for a robot using the `2D-1SP` controller, with starting orientation of (a) 0°, (b) 90°, (c) 180° and (d) -90°, from the preferred orientation. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.

through visual software tracking, used on the recordings of the trials. However, as the light source used as goal emitted white light, the closer the robot is from the goal, the more inaccurate the tracking. As a result, the tracking of the orientation of the robot, obtained for translations above 50 cm might not be accurate.

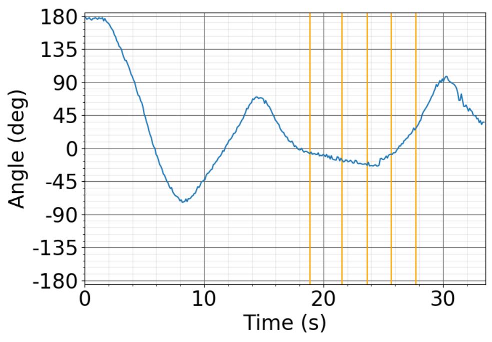
When comparing the results obtained for each controller in terms of changes in orientation during the trial, the changes in orientation tend to be in a single direction for the `2D-1SP` controller, i.e. always decreasing (particularly for Figures 5.26c and 5.26d), whilst in trials using the `2D-2SP` controller, there are clear changes in direction of rotation. These results are in accordance with the expectation for the control policies presented in Section 4.4.1. The increases in angle seen in trials performed with controller `2S-1SP` can be attributed



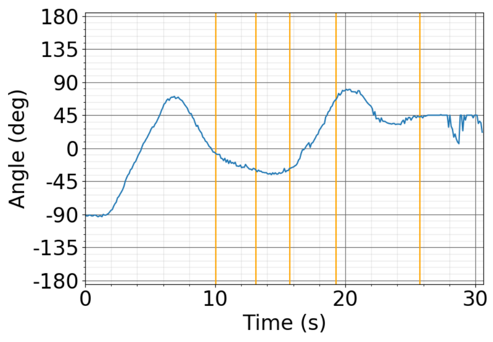
(a)



(b)



(c)



(d)

Fig. 5.27 Examples of the changes in angle during a trial for a robot using the `2D-2SP` controller, with starting orientation of (a) 0°, (b) 90°, (c) 180° and (d) -90°, from the preferred orientation. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the x axis.

to differences in output from the pump, and possible collisions with the elastic bands that surround the interior of the water tank.

5.4 Summary

In this chapter, we validated the 2D and 3D controllers proposed, through physics simulations and physical experiments. We performed multiple simulation studies, to evaluate the performance of the proposed 3D controllers in different scenarios. The simulator used was adapted from an existing one [115], and relies on the open source ODE physics library to simulate the robot's interactions with the environment. We tested the performance of the controllers to changes in the environment (drag coefficient), distance to the goal, modular resolution, actuator noise and failure and sensor noise and failure.

All of the controllers, with exception of the `3D-2SP` controller performed robustly for low to medium drag coefficient values. For very high drags, the controllers stated dropping in performance, due to velocity reduction caused by the drag. The `3D-2SP` controller, failed to perform in low drag environments, due to only being able to rotate in one direction.

The `3D-0SP-V1`, `3D-0SP-V2` and `3D-2SP` controllers' performance showed some dependency with distance between the goal and the robot. The `3D-2SP` controller performed acceptably for short distances, flawlessly for medium distances and for long distances is unable to reach the goal. Further tests indicated that the distances at which this controller is able to perform flawlessly are dependent on the environment (drag coefficient). The performance of the `3D-0SP-V1` and `3D-0SP-V2` increased with the distance. For very short distances both controllers perform poorly. As the distance increases the performance of both increases, with the `3D-0SP-V2` controller outperforming the `3D-0SP-V1` controller, until both achieve flawless performance. However, the better performance of the `3D-0SP-V2` for shorter distances is paired with increased trial completion times and energy consumption. The remaining controllers performed comparatively well for all distances.

The performance of the controllers that use shared power lines scaled well for the modular resolution values tested. However, if the modular resolution is increased further, we predict that the performance will start degenerating. The `3D-0SP-V1` and `3D-0SP-V2` controllers showed performance drops even for slight increases in modular resolution. The performance drops are caused by the reduced torques acting during rotation, making time one of the limiting factors for this study. However, increased modular resolution brings an increase in the accuracy of the movement.

When subject to actuator noise or failure, all controllers showed no drops in performance for low noise/failure values. As the probability of a malfunctioning or noisy actuator increased, the performance of the controllers started falling, up to 50% probability, with the controllers producing the least effective movement dropping first. Above this value, no controller was able to perform consistently.

While the reactions to actuator noise or failure were similar, the performance observed for the controllers under sensor noise or failure were significantly different. Under sensor noise only the `3D-0SP-V1` controller was able to perform consistently. The `3D-0SP-V2` showed some resilience to low sensor noise values, but rapidly dropped performance for higher noise probability values. On the other hand, the `3D-0SP-V1` controller is the most affected by sensor failure, while the remaining controllers performed robustly up to high sensor failure values.

When considering convex non cubic robots, while every controller was able to somewhat complete the task, the performance varied, depending on the axis along the longest robot dimension. In general, robots with a larger X_1^{local} dimension performed worse, due to reduced translation velocity. For robots of random morphology, only the 3D-0SP-V1 controller was capable of completing all trials (other than the centralised controller), with the 3D-5SP-V2 controllers being the only other controller to be able to complete any trial. When testing a reduced sensing angle, the controllers that do not make use of share power lines were unable to complete the task, as when not detecting the goal, the robot only moves due to inertia. The other controllers were able to somewhat complete the task, with the 3D-2SP controller showing better performance, as higher overshoot leads to less time in an orientation where the goal is not detected.

The physical experiments were performed using the hardware developed in Chapter 3, with the addition of an Arduino, to serve as the shared power lines. The experiments tested the performance of the 2D-1SP and 2D-2SP controllers, with the latter showing a higher success rate but increased trial completion time.

Chapter 6

Conclusions

In this thesis we designed and built a new version of the MHP robot, the MHP 2, and developed novel minimalistic control strategies that can be used to allow an MHP robot to reach a goal with a specific orientation.

Chapter 3 presented MHP 2, a new implementation of the MHP concept, offering enhanced reliability and capabilities. This improved the robot's usability and utility, also allowing controllers with increased resource requirements to be realised. Moreover, each module's volume and mass was reduced by 63% and 73%, when compared to the original system. The reduction of the module's volume and mass was a first step towards miniaturisation of the robot, which could have the potential of opening new applications of the system. Additionally, the simplicity of the design helped lower the production cost, allowing for the validation of the controllers in a system comprised of a larger number of modules.

We performed a set of physical experiments with MHP 2 robots, using two decentralised and reactive controllers. The robots were tasked with moving towards a goal. The two controllers used are occlusion based, and were proposed and tested in simulation in [111]. The controllers differ only by the presence or absence of communication between modules and their respective faces. The controllers were compared in terms of trial completion time and energy used. When used in the physical robots, both controllers were successful in completing the task. When using communication, the energy consumed was reduced by 70%. However, the energy decrease came at the cost of an increased trial completion time. From the analysis of the results, we concluded that the placement and detection angle of the goal sensors led to the presence of false-negative sensor readings, which had a significant impact in the performance of the controllers. The significant decrease in energy consumption, even at the cost of increased trial completion time, can be advantageous in situations where the energy available to the system could be limited. For example, when miniaturising the modules, the power-source of the system could be significantly smaller. This could make the reduction of the energy consumption a priority, to allow extended periods of operation.

Chapter 4 aimed to develop novel, fully decentralised and reactive strategies that would allow an MHP robot to control the orientation with which the goal is reached. We established that there is a wide variety of unique total forces and net torques a 3D convex MHP robot can produce. The analysis performed quantified the number of controllable states a cubic MHP robot is capable of. It also gave insight on how finely the angular and linear velocities could be controlled. Multiple controller solutions were proposed, which differ in the amount of information shared between the modules, via shared power lines. This allowed for an evaluation of the trade-off between the simplicity of the controller solutions (by including communication) and performance. We proposed controller solutions using zero to two bits of shared information for 2D robots, and controller solutions using zero, two and five bits of shared information for 3D robots. For all controller solutions, the activation state of the pump of a given module's face was based on its local connectivity and sensor/shared power lines values. As a result, all of the controllers could be implemented on a hardware level, without requiring a dedicated processor. To the best of our knowledge, the controllers proposed in this thesis are the first set of controllers able to control both the translation and orientation of a modular robot in a fully decentralised and reactive manner.

The controllers that use shared power lines were formally proven correct for convex robots, under the assumption that in its initial position the robot was able to rotate in place without colliding with the goal. For the controllers that do not use shared power lines, this condition was not sufficient to guarantee successful task completion. Assuming a sufficiently high viscous environment, and slow rotation and translation velocities, we derived an initial distance condition that ensured the robot can reach the goal with the correct orientation, using these controllers. While this could limit their uses, obtaining controller solutions that make no use of communication allowed for truly minimalistic implementations. Additionally, there were less elements prone to failure. As a result, the reliability of the system was significantly increased. However, it did not allow for coordination between modules. In situations where the spaces in which robots need to move in could be severely constrained, using communication though shared power lines allows the robots to complete the task, while not significantly increasing the complexity of the system. An analysis of the controllers when applied to robots of generic shape was performed. Multiple counter-examples were found that indicate that relying on the local connectivity to determine active actuators might not be enough to guarantee task completion for robots of concave shape. However, we proved that for orthogonally convex robots that are symmetric along one axis perpendicular to the rotation axis the controllers are able to complete the task, without requiring changes.

Chapter 5 validated the controllers through computational simulations and physical experiments. The simulations were performed in a 3D unbounded environment, containing a single robot and a goal point. We performed multiple studies, comparing the performance of the decentralised controllers with that of a centralised controller from the literature [27]. In general, we observed that controllers that use zero or a high number of shared power lines were better equipped to deal with different scenarios, while maintaining a high success rate. On the other hand, controllers that use shared power lines were highly susceptible to sensor noise. However, this applies only to sensors malfunctioning by detecting the goal, when no goal was in sight (false positive sensor readings). All of the controllers using shared power lines showed high resilience to sensor failure. One of the controllers with no shared power lines also demonstrated a high level of flexibility, robustness to noise and outperformed all other controllers for robots of random morphologies.

Controllers able to rotate in multiple directions along one or all of the rotation axes were shown to be largely independent of the environment conditions in term of performance. Additionally, the larger the number of rotation options a controller was able to perform, the faster the rotation was performed. When increasing the robot's modular resolution, drops in performance were observed, associated with a slower rotation of the robot. However, the robot also performed more accurate rotations, showing more consistent collision points with the goal. All of these results allow for a better comprehension of the applicability of the proposed controllers. In turn, this will allow for the best option to a specific situation/task to be chosen, based on the environment conditions and limitations of the modular robot to be used. While the proposed controllers presented some limitations, they can be advantageous when minimalistic control is required, either due to the lack of resources or system miniaturisation.

The physical experiments were performed using the controllers proposed for 2D movement, and the hardware developed in Chapter 3. Only the controllers using shared power lines were tested due to the limited number of modules available and environment conditions. Similarly to the simulation results, the controller with a higher number of shared power lines performed better, in terms of success rate, although with increased trial completion time.

6.1 Future work

The work performed in this thesis produced encouraging results for controlling the pose of MHP robots with minimalistic control strategies. However, as previously discussed, there are some limitations to this work which can be addressed. Additionally, there are research avenues that the work performed has yet to explore.

Further improvements could be made in the MHP physical implementation. While a basic set of requirements was met, the use of the system in physical experiments brought to light elements not considered in the design process. The issues found could be addressed in a new version of the system. Moreover, we have yet to realise on the physical platform the potential for large reconfiguration space and scalability the concept presents [111], as only 2D reconfiguration is possible with the current system. To demonstrate these capabilities, further module miniaturisation, or design an implementation of the robot capable of 3D/underwater movement are desirable. Additionally, self-reconfiguration of the modules could be added in new versions to increase the adaptability of the system.

In this thesis we presented an analysis of the number of unique total forces and net torques a 3D convex MHP robot is able to produce. This analysis gives insight into how finely one can control the linear and angular velocities of the robot, allowing the development of control strategies capable of precise movement. However, for multiple simultaneous rotation axes, the analysis was only performed for robots that have the same number of modules along each axis. Future work could expand this analysis for convex robots with different side lengths. Additionally, this analysis does not specify how many different movements a convex MHP robot is capable of performing, which can also be a requirement for precise movement. As a result, an in depth analysis on the number of different movement an MHP robot is able to perform could be done. By determining the multiple ways an MHP robot can move, new control strategies that generate a more efficient movement could be developed.

Convex robots can have limited usefulness in real applications. However, as showed in Section 4.6, the controllers proposed are only guaranteed to succeed for convex or orthogonally convex with a symmetry axis robots. On one hand, this limits the geometries for which the controllers can be used. On the other hand, the counter-examples found correspond to geometries that would have little practical application. Future research could be performed in this area, not only to expand the robot configurations the controllers are applicable, but also to investigate what classes of geometries guarantee successful task completion in specific applications. With the increased set of geometries where the controllers could be used, the applicability of the controllers to real applications increases.

One of the largest limitations of the proposed controllers was their susceptibility to sensor noise. One solution could be for the modules to filter their sensor and/or power line readings. Another solution could be for them to reach consensus on the power line state (e.g., via quorum sensing). However, these solutions likely require the use of run-time memory and/or computation. Future research could attempt to improve the controllers response to sensor noise, both with and without requiring arithmetic computation and run-time memory. By improving the resilience of the controllers to sensor noise the reliability of the controllers increases.

When using robots with a reduced sensing angle and/or of random morphology, the overall performance of the proposed controllers is poor. However, in these situations there was still one controller (for each case) that showed reasonable to good performance.

While validation was performed through physical experiments, only the 2D controllers that use shared power lines were tested, due to hardware and experimental setup limitations. Future work should present a comprehensive comparison between the performance of all 2D controllers, using a better suited experimental setup. Additionally, in order to to gain insight on the limitations of hardware that do not allow the performance of the controller to increase, a comparison between simulation and experimental results could be performed. While the simulation study performed with a reduced sensing angle was a step, additional studies should be performed.

In this thesis we proposed and validated strategies that allow a robot to reach a goal with a predefined orientation. However, the use of these controllers was not demonstrated in real-life applications. Future work could study how the proposed controllers can be used in specific applications. For example, the controllers could be tested in the tracking of a dynamic goal [118]. In this type of task, the robot could be required to detect and approach the goal, as well as accompany it, while maintaining a fixed orientation relative to the goal. Other tasks, such as docking, object transportation, structure building and inspection could be considered.

References

[1] Z. Butler and A. Rizzi, “Distributed and cellular robots,” in *Springer Handbook of Robotics*, pp. 911–920, Springer, 2008.

[2] L. E. Parker, “Multiple mobile robot systems,” in *Springer Handbook of Robotics*, pp. 921–941, Springer, 2008.

[3] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, “Swarm robotics: a review from the swarm engineering perspective,” *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[4] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, E. Bonabeau, and G. Theraula, *Self-organization in biological systems*. Princeton university press, 2003.

[5] K. Støy, D. Brandt, and D. J. Christensen, *Self-Reconfigurable Robots: An Introduction*. The MIT Press, 2010.

[6] M. Yim and D. Duff, “Modular robots,” *IEEE Spectrum*, vol. 39, pp. 30–34, Feb 2002.

[7] P. Moubarak and P. Ben-Tzvi, “Modular and reconfigurable mobile robotics,” *Robotics and Autonomous Systems*, vol. 60, no. 12, pp. 1648–1663, 2012.

[8] M. Yim, W. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, “Modular self-reconfigurable robot systems [grand challenges of robotics],” *IEEE Robotics Automation Magazine*, vol. 14, pp. 43–52, March 2007.

[9] S. Murata and H. Kurokawa, “Self-reconfigurable robots,” *IEEE Robotics Automation Magazine*, vol. 14, pp. 71–78, March 2007.

[10] J. Seo, J. Paik, and M. Yim, “Modular reconfigurable robotics,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 63–88, 2019.

[11] W. Savoie, S. Cannon, J. J. Daymude, R. Warkentin, S. Li, A. Richa, D. Randall, and D. I. Goldman, “Phototactic supersmarticles,” *Artificial Life and Robotics*, vol. 23, pp. 459–468, 1 2018.

[12] S. Kernbach, O. Scholz, K. Harada, S. Popesku, J. Liedke, H. Raja, W. Liu, F. Caparrelli, J. Jemai, J. Havlik, E. Meister, and P. Levi, “Multi-Robot Organisms: State of the Art,” *arXiv Prepr. arXiv1108.5543*, 2011.

[13] A. L. Christensen, O. Rehan, M. Dorigo, *et al.*, “Morphology control in a multirobot system,” *IEEE Robotics & Automation Magazine*, vol. 14, no. 4, pp. 18–25, 2007.

[14] R. O'Grady, A. L. Christensen, and M. Dorigo, "Swarmorph: multirobot morphogenesis using directional self-assembly," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 738–743, 2009.

[15] S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, and S. Kokaji, "Hardware design of modular robotic system," in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2210–2217, IEEE, 2000.

[16] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. Homans, "Modular reconfigurable robots in space applications," *Autonomous Robots*, vol. 14, pp. 225–237, Mar 2003.

[17] J. Yuh, "Design and Control of Autonomous Underwater Robots: A Survey," *Autonomous Robots*, vol. 8, no. 1, pp. 7–24, 2000.

[18] X. Li, J.-F. Martínez, J. Rodríguez-Molina, and N. Martínez, "A survey on intermediation architectures for underwater robotics," *Sensors*, vol. 16, no. 2, p. 190, 2016.

[19] V. Costa, M. Duarte, T. Rodrigues, S. M. Oliveira, and A. L. Christensen, "Design and development of an inexpensive aquatic swarm robotics system," in *OCEANS 2016-Shanghai*, pp. 1–7, IEEE, 2016.

[20] M. Duarte, J. Gomes, V. Costa, T. Rodrigues, F. Silva, V. Lobo, M. M. Marques, S. M. Oliveira, and A. L. Christensen, "Application of swarm robotics systems to marine environmental monitoring," in *OCEANS 2016-Shanghai*, pp. 1–8, IEEE, 2016.

[21] N. Sakagami, T. Kanayama, T. Ueda, H. Hashizume, M. Shibata, H. Onishi, S. Murakami, and S. Kawamura, "Design and development of an attitude control system for a human-sized rov," in *2010 11th International Conference on Control Automation Robotics & Vision*, pp. 2141–2146, IEEE, 2010.

[22] S. Zhao and J. Yuh, "Experimental study on advanced underwater robot control," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 695–703, 2005.

[23] T. Fukuda, S. Nakagawa, Y. Kawauchi, and M. Buss, "Structure decision method for self organising robots based on cell structures-cebot," in *Proceedings, 1989 International Conference on Robotics and Automation*, pp. 695–700, May 1989.

[24] Z. Nagy, R. Oung, J. J. Abbott, and B. J. Nelson, "Experimental investigation of magnetic self-assembly for swallowable modular robots," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1915–1920, IEEE, 2008.

[25] B. J. Nelson, I. K. Kaliakatsos, and J. J. Abbott, "Microrobots for minimally invasive medicine," *Annual review of biomedical engineering*, vol. 12, pp. 55–85, 2010.

[26] M. J. Doyle, X. Xu, Y. Gu, F. Perez-Diaz, C. Parrott, and R. Groß, "Modular hydraulic propulsion: A robot that moves by routing fluid through itself," in *2016 IEEE International Conference on Robotics and Automation*, pp. 5189–5196, 2016.

[27] M. Doniec, I. Vasilescu, C. Detweiler, and D. Rus, “Complete se3 underwater robot control with arbitrary thruster configurations,” in *2010 IEEE International Conference on Robotics and Automation*, pp. 5295–5301, 2010.

[28] M. Yim, Y. Zhang, K. Roufas, D. Duff, and C. Eldershaw, “Connecting and disconnecting for chain self-reconfiguration with polybot,” *IEEE/ASME Transactions on mechatronics*, vol. 7, no. 4, pp. 442–451, 2002.

[29] E. H. Østergaard, K. Kassow, R. Beck, and H. H. Lund, “Design of the atron lattice-based self-reconfigurable robot,” *Autonomous Robots*, vol. 21, no. 2, pp. 165–183, 2006.

[30] F. Mondada, A. Guignard, M. Bonani, D. Bar, M. Lauria, and D. Floreano, “Swarm-bot: From concept to implementation,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 2, pp. 1626–1631, IEEE, 2003.

[31] T. Fukuda and S. Nakagawa, “Dynamically reconfigurable robotic system,” in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, pp. 1581–1586, IEEE, 1988.

[32] T. Fukuda, “Self organizing robots based on cell structures-cebot,” in *Proc. IEEE Int. Workshop on Intelligent Robots and Systems (IROS’88)*, pp. 145–150, 1988.

[33] T. Fukuda and Y. Kawauchi, “Cellular robotic system (cebot) as one of the realization of self-organizing intelligent universal manipulator,” in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 662–667, IEEE, 1990.

[34] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, and S. Murata, “Distributed self-reconfiguration of m-tran iii modular robotic system,” *The International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 373–386, 2008.

[35] V. Zykov, E. Mytilinaios, M. Desnoyer, and H. Lipson, “Evolved and designed self-reproducing modular robotics,” *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 308–319, 2007.

[36] J. W. Romanishin, K. Gilpin, S. Claici, and D. Rus, “3d m-blocks: Self-reconfiguring robots capable of locomotion via pivoting in three dimensions,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1925–1932, IEEE, 2015.

[37] M. Yim, *Locomotion with a unit-modular reconfigurable robot.* PhD thesis, Citeseer, 1994.

[38] M. Yim, “New locomotion gaits,” in *Proceedings of the 1994 IEEE International conference on Robotics and Automation*, pp. 2508–2514, IEEE, 1994.

[39] J. Sastra, S. Chitta, and M. Yim, “Dynamic rolling for a modular loop robot,” *The International Journal of Robotics Research*, vol. 28, no. 6, pp. 758–773, 2009.

[40] M. Yim, D. G. Duff, and K. D. Roufas, “PolyBot: a modular reconfigurable robot,” in *Proceedings., 2000 IEEE International Conference on Robotics and Automation.*, vol. 1, pp. 514–520, IEEE, 2000.

[41] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji, “M-TRAN: Self-reconfigurable modular robotic system,” *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 4, pp. 431–441, 2002.

[42] H. Kurokawa, A. Kamimura, E. Yoshida, K. Tomita, S. Kokaji, and S. Murata, “M-tran ii: metamorphosis from a four-legged walker to a caterpillar,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3, pp. 2454–2459, IEEE, 2003.

[43] J. Davey, N. Kwok, and M. Yim, “Emulating self-reconfigurable robots-design of the smores system,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4464–4469, IEEE, 2012.

[44] V. Zykov, A. Chan, and H. Lipson, “Molecubes: An open-source modular robotics kit,” in *IROS-2007 Self-Reconfigurable Robotics Workshop*, pp. 3–6, 2007.

[45] V. Zykov, P. Williams, N. Lassabe, and H. Lipson, “Molecubes extended: Diversifying capabilities of open-source modular robotics,” in *IROS-2008 Self-Reconfigurable Robotics Workshop*, pp. 22–26, 2008.

[46] A. Sprowitz, S. Pouya, S. Bonardi, J. Van Den Kieboom, R. Mockel, A. Billard, P. Dillenbourg, and A. J. Ijspeert, “Roombots: reconfigurable robots for adaptive furniture,” *IEEE Computational Intelligence Magazine*, vol. 5, no. 3, pp. 20–32, 2010.

[47] A. Sproewitz, A. Billard, P. Dillenbourg, and A. J. Ijspeert, “Roombots-mechanical design of self-reconfiguring modular robots for adaptive furniture,” in *2009 IEEE International Conference on Robotics and Automation*, pp. 4259–4264, IEEE, 2009.

[48] A. Sproewitz, P. Laprade, S. Bonardi, M. Mayer, R. Moeckel, P.-A. Mudry, and A. J. Ijspeert, “Roombots—towards decentralized reconfiguration with self-reconfiguring modular robotic metamodules,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1126–1132, IEEE, 2010.

[49] A. Spröwitz, R. Moeckel, M. Vespignani, S. Bonardi, and A. J. Ijspeert, “Roombots: A hardware perspective on 3d self-reconfiguration and locomotion with a homogeneous modular robot,” *Robotics and Autonomous Systems*, vol. 62, no. 7, pp. 1016–1033, 2014.

[50] J. W. Romanishin, K. Gilpin, and D. Rus, “M-blocks: Momentum-driven, magnetic modular robots,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4288–4295, IEEE, 2013.

[51] M. W. Jorgensen, E. H. Ostergaard, and H. H. Lund, “Modular ATRON: Modules for a self-reconfigurable robot,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 2, pp. 2068–2073, IEEE, 2004.

[52] C. Parrott, T. J. Dodd, and R. Groß, “HyMod: A 3-DOF hybrid mobile and self-reconfigurable modular robot and its extensions,” in *Distributed Autonomous Robotic Systems*, vol. 6, pp. 401–414, Springer, 2018.

[53] C. Parrott, T. J. Dodd, and R. Groß, “HiGen: A high-speed genderless mechanical connection mechanism with single-sided disconnect for self-reconfigurable modular robots,” in *2014 IEEE/RSJ International Conference on Intellegent Robots and Systems*, pp. 3926–3932, IEEE, 2014.

[54] C. Parrot, *A Hybrid and Extendable Self-Reconfigurable Modular Robotic Systems*. PhD thesis, The University of Sheffield, 2016.

[55] M. Dorigo, E. Tuci, R. Groß, V. Trianni, T. H. Labella, S. Nouyan, C. Ampatzis, J.-L. Deneubourg, G. Baldassarre, S. Nolfi, *et al.*, “The swarm-bots project,” in *International Workshop on Swarm Robotics*, pp. 31–44, Springer, 2004.

[56] F. Mondada, L. M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneuborg, and M. Dorigo, “The cooperation of swarm-bots: Physical interactions in collective robotics,” *IEEE Robotics & Automation Magazine*, vol. 12, no. 2, pp. 21–28, 2005.

[57] R. Groß, M. Bonani, F. Mondada, and M. Dorigo, “Autonomous self-assembly in a swarm-bot,” *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1115–1130, 2006.

[58] S. C. Goldstein and T. C. Mowry, “Claytronics: A scalable basis for future robots,” in *RoboSphere*, (Moffett Field, CA), Nov 2004.

[59] S. C. Goldstein, J. D. Campbell, and T. C. Mowry, “Programmable matter,” *Computer*, vol. 38, no. 6, pp. 99–101, 2005.

[60] M. E. Karagozler, S. C. Goldstein, and J. R. Reid, “Stress-driven mems assembly+ electrostatic forces= 1mm diameter robot,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2763–2769, IEEE, 2009.

[61] B. Piranda and J. Bourgeois, “Designing a quasi-spherical module for a huge modular robot to create programmable matter,” *Autonomous Robots*, vol. 42, no. 8, pp. 1619–1633, 2018.

[62] K. Gilpin, A. Knaian, and D. Rus, “Robot pebbles: One centimeter modules for programmable matter through self-disassembly,” in *2010 IEEE International Conference on Robotics and Automation*, pp. 2485–2492, IEEE, 2010.

[63] R. Oung and R. D’Andrea, “The distributed flight array: Design, implementation, and analysis of a modular vertical take-off and landing vehicle,” *The International Journal of Robotics Research*, vol. 33, no. 3, pp. 375–400, 2014.

[64] D. Saldaña, B. Gabrich, G. Li, M. Yim, and V. Kumar, “Modquad: The flying modular structure that self-assembles in midair,” in *2018 IEEE International Conference on Robotics and Automation*, pp. 691–698, 2018.

[65] S. Braley, C. Rubens, T. Merritt, and R. Vertegaal, “Griddrones: A self-levitating physical voxel lattice for interactive 3d surface deformations,” in *The 31st Annual ACM Symposium on User Interface Software and Technology*, pp. 87–98, ACM, 2018.

[66] R. Oung, A. Ramezani, and R. D'Andrea, "Feasibility of a distributed flight array," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 3038–3044, IEEE, 2009.

[67] R. Oung and R. D'Andrea, "The distributed flight array," *Mechatronics*, vol. 21, no. 6, pp. 908–917, 2011.

[68] G. Li, B. Gabrich, D. Saldaña, J. Das, V. Kumar, and M. Yim, "ModQuad-Vi: A vision-based self-assembling modular quadrotor," in *2019 IEEE International Conference onRobotics and Automation (ICRA),*, pp. 346–352, IEEE, IEEE, 2019.

[69] R. Naldi, F. Forte, and L. Marconi, "A class of modular aerial robots," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 3584–3589, IEEE, 2011.

[70] R. Naldi, F. Forte, A. Serrani, and L. Marconi, "Modeling and control of a class of modular aerial robots combining under actuated and fully actuated behavior," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1869–1885, 2015.

[71] P. White, V. Zykov, J. Bongard, and H. Lipson, "Three dimensional stochastic re-configuration of modular robots.," in *Robotics: Science and Systems*, pp. 161–168, 2005.

[72] P. White, K. Kopanski, and H. Lipson, "Stochastic self-reconfigurable cellular robotics," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 3, pp. 2888–2893, IEEE, 2004.

[73] M. T. Tolley, M. Kalontarov, J. Neubert, D. Erickson, and H. Lipson, "Stochastic modular robotic systems: A study of fluidic assembly strategies," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 518–530, 2010.

[74] J. Neubert, A. P. Cantwell, S. Constantin, M. Kalontarov, D. Erickson, and H. Lipson, "A robotic module for stochastic fluidic assembly of 3d self-reconfiguring structures," in *2010 IEEE International Conference on Robotics and Automation*, pp. 2479–2484, 2010.

[75] S. Miyashita, M. Hadorn, and P. E. Hotz, "Self-assemble of water floating active units," in *2007 IEEE Interntional Conference on Mechatronics*, pp. 8–10, 2007.

[76] B. Haghighat, E. Droz, and A. Martinoli, "Lily : A Miniature Floating Robotic Platform for Programmable Stochastic Self-Assembly," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1941–1948, IEEE, 2015.

[77] M. T. Tolley and H. Lipson, "Fluidic manipulation for scalable stochastic 3D assembly of modular robots," in *2010 IEEE international conference on robotics and automation*, pp. 2473–2478, 2010.

[78] S. Miyashita, F. Casanova, M. Lungarella, and R. Pfeifer, "Tribolon: Water based self-assembly robot with freezing connector (video)," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4147–4148, IEEE, 2008.

[79] I. Vasilescu, C. Detweiler, M. Doniec, D. Gurdan, S. Sosnowski, J. Stumpf, and D. Rus, "Amour v: A hovering energy efficient underwater robot capable of dynamic payloads," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 547–570, 2010.

[80] I. O'Hara, J. Paulos, J. Davey, N. Eckenstein, N. Doshi, T. Tosun, J. Greco, J. Seo, M. Turpin, V. Kumar, and M. Yim, "Self-assembly of a swarm of autonomous boats into floating structures," in *2014 IEEE International Conference on Robotics and Automation*, pp. 1234–1240, 2014.

[81] S. Mintchev, C. Stefanini, A. Girin, S. Marrazza, S. Orofino, V. Lebastard, L. Manfredi, P. Dario, and F. Boyer, "An underwater reconfigurable robot with bioinspired electric sense," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1149–1154, IEEE, 2012.

[82] I. Vasilescu, P. Varshavskaya, K. Kotay, and D. Rus, "Autonomous Modular Optical Underwater Robot (AMOUR) design, prototype and feasibility study," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1603–1609, IEEE, 2005.

[83] J. Paulos, N. Eckenstein, T. Tosun, J. Seo, J. Davey, J. Greco, V. Kumar, and M. Yim, "Automated self-assembly of large maritime structures by a team of robotic boats," *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 958–968, July 2015.

[84] W. Wang, L. A. Mateos, S. Park, P. Leoni, B. Gheneti, F. Duarte, C. Ratti, and D. Rus, "Design, modeling, and nonlinear model predictive tracking control of a novel autonomous surface vehicle," in *2018 IEEE International Conference on Robotics and Automation*, pp. 6189–6196, 2018.

[85] L. A. Mateos, W. Wang, B. Gheneti, F. Duarte, C. Ratti, and D. Rus, "Autonomous latching system for robotic boats," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7933–7939, IEEE, 2019.

[86] G. Konidaris, T. Taylor, and J. Hallam, "Hydrogen: Automatically generating self-assembly code for hydron units," in *Distributed Autonomous Robotic Systems 6*, pp. 33–42, Springer, 2007.

[87] B. von Haller, A. Ijspeert, and D. Floreano, "Co-evolution of structures and controllers for neubot underwater modular robots," in *European Conference on Artificial Life*, pp. 189–199, Springer, 2005.

[88] D. J. Christensen, J. C. Andersen, M. Blanke, L. Furno, R. Galeazzi, P. N. Hansen, and M. C. Nielsen, "Collective modular underwater robotic system for long-term autonomous operation," in *ICRA 2015 Workshop on Persistent Autonomy for Aquatic Robotics: the Role of Control and Learning in Single and Multi-Robot Systems*, IEEE, 2015.

[89] L. Furno, M. Blanke, R. Galeazzi, and D. J. Christensen, "Self-reconfiguration of modular underwater robots using an energy heuristic," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6277–6284, IEEE, 2017.

[90] S. Mintchev, R. Ranzani, F. Fabiani, and C. Stefanini, “Towards docking for small scale underwater robots,” *Autonomous Robots*, vol. 38, no. 3, pp. 283–299, 2015.

[91] S. Miyashita, M. Kessler, and M. Lungarella, “How morphology affects self-assembly in a stochastic modular robot,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 3533–3538, 2008.

[92] S. Miyashita, M. Göldi, and R. Pfeifer, “How reverse reactions influence the yield of self-assembly robots,” *The International Journal of Robotics Research*, vol. 30, no. 5, pp. 627–641, 2011.

[93] B. Haghighat, M. Mastrangeli, G. Mermoud, F. Schill, and A. Martinoli, “Fluid-mediated stochastic self-assembly at centimetric and sub-millimetric scales: Design, modeling, and control,” *Micromachines*, vol. 7, no. 8, p. 138, 2016.

[94] B. Haghighat, B. Platerrier, L. Waegeli, and A. Martinoli, “Synthesizing rulesets for programmable robotic self-assembly: A case study using floating miniaturized robots,” in *International Conference on Swarm Intelligence*, pp. 197–209, Springer, 2016.

[95] B. Haghighat, H. Khodr, and A. Martinoli, “Lightweight physics-based models for the control of fluid-mediated self-assembly of robotic modules,” *Robotics and Autonomous Systems*, 2019.

[96] V. Ganesan and M. Chitre, “On stochastic self-assembly of underwater robots,” *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 251–258, 2016.

[97] H. Bojinov, A. Casal, and T. Hogg, “Multiagent control of self-reconfigurable robots,” *Artificial Intelligence*, vol. 142, no. 2, pp. 99–120, 2002.

[98] C. Ampatzis, E. Tuci, V. Trianni, A. L. Christensen, and M. Dorigo, “Evolving self-assembly in autonomous homogeneous robots: Experiments with two physical robots,” *Artificial Life*, vol. 15, no. 4, pp. 465–484, 2009.

[99] R. O’Grady, R. Groß, A. L. Christensen, and M. Dorigo, “Self-assembly strategies in a group of autonomous mobile robots,” *Autonomous Robots*, vol. 28, no. 4, pp. 439–455, 2010.

[100] M. Doniec, C. Detweiler, and D. Rus, “Estimation of Thruster Configurations for Reconfigurable Modular Underwater Robots,” *Experimental Robotics*, vol. 79, pp. 655–666, 2014.

[101] S. Park, E. Kayacan, C. Ratti, and D. Rus, “Coordinated control of a reconfigurable multi-vessel platform: Robust control approach,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[102] M. Feemster, J. Esposito, and J. Nicholson, “Manipulation of large object by swarms of autonomous marine vehicles, part i: Rotational motions,” in *2006 Proceeding of the Thirty-Eighth Southeastern Symposium on System Theory*, pp. 205–209, IEEE, 2006.

[103] D. Braganza, M. Feemster, and D. Dawson, “Positioning of large surface vessels using multiple tugboats,” in *2007 American Control Conference*, pp. 912–917, IEEE, 2007.

[104] J. Esposito, M. Feemster, and E. Smith, “Cooperative manipulation on the water using a swarm of autonomous tugboats,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 1501–1506, IEEE, 2008.

[105] J. M. Esposito, “Distributed grasp synthesis for swarm manipulation with applications to autonomous tugboats,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 1489–1494, IEEE, 2008.

[106] G. Sartoretti, S. Shaw, and M. A. Hsieh, “Distributed planar manipulation in fluidic environments,” in *2016 IEEE International Conference on Robotics and Automation*, pp. 5322–5327, 2016.

[107] G. A. S. Pereira, M. F. M. Campos, and V. Kumar, “Decentralized algorithms for multi-robot manipulation via caging,” *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 783–795, 2004.

[108] Y. Hu, L. Wang, J. Liang, and T. Wang, “Cooperative box-pushing with multiple autonomous robotic fish in underwater environment,” *IET control theory & applications*, vol. 5, no. 17, pp. 2015–2022, 2011.

[109] J. Chen, M. Gauci, W. Li, A. Kolling, and R. Groß, “Occlusion-based cooperative transport with a swarm of miniature mobile robots,” *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 307–321, 2015.

[110] J. Chen, M. Gauci, and R. Groß, “A strategy for transporting tall objects with a swarm of miniature mobile robots,” in *2013 IEEE International Conference on Robotics and Automation*, pp. 863–869, IEEE, IEEE, 2013.

[111] M. J. Doyle, J. V. A. Marques, I. Vandermeulen, C. Parrot, Y. Gu, X. Xu, A. Kolling, and R. Groß, “Modular Fluidic Propulsion Robots,” *conditionally accepted in IEEE Ttransaction on Robotics*.

[112] N. Jakobi, P. Husbands, and I. Harvey, “Noise and the reality gap: The use of simulation in evolutionary robotics,” in *Advances in Artificial Life* (F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, eds.), pp. 704–720, Springer Berlin Heidelberg, 1995.

[113] G. Bradski, “Open source computer vision library,” 2000.

[114] J. V. A. Marques, A. Özdemir, M. J. Doyle, D. Rus, and R. Groß, “Decentralized pose control of modular reconfigurable robots operating in liquid environments,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, *in press*.

[115] M. J. Doyle, *The Propulsion of Reconfigurable Modular Robots in Fluidic Environments*. PhD thesis, The University of Sheffield, 2019.

[116] R. Campa and H. de la Torre, “Pose control of robot manipulators using different orientation representations: A comparative review,” in *2009 American Control Conference*, pp. 2855–2860, June 2009.

[117] R. Smith, “Open dynamics engine,” 2005.

[118] S. Martínez and F. Bullo, “Optimal sensor placement and motion coordination for target tracking,” *Automatica*, vol. 42, no. 4, pp. 661–668, 2006.

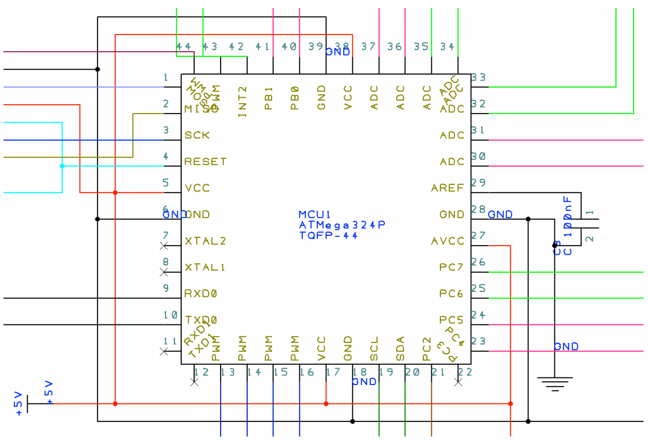
Appendix A

Circuit schematics for the new implementation of the MHP robots

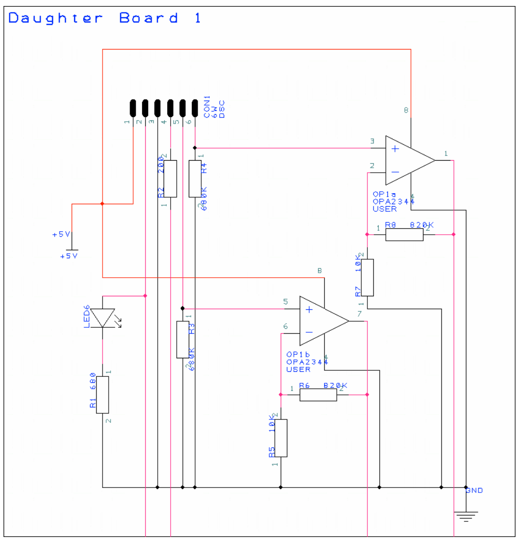
This appendix presents the circuit schematics for MHP 2 robot introduced in Chapter 3.

Figure A.1a depicts the microcontroller used, as well as its ports that are used in the circuit. The red line represents the 5 V power line while the black lines represent connections to ground. Pins 1 to 4 are the connections to the programming header. Pins 13 to 16 (dark blue lines) power the motor drivers. Pins 19 and 20 (dark green) are connected to the compass. Pin 21 is connected to the infra-red receiver responsible for starting the algorithm. Pins 23 to 26, 30 to 37 and 40 to 43 (light green and pink lines) connect to the daughter boards. Pin 44 is connected to the debug LED. Figure A.1b shows the detailed connections between the microcontroller and the connection pins with one of the daughter boards. As the daughter boards, all of the circuitry necessary for the sensors (operational amplifiers and LED's) was placed on the main board. The operational amplifiers connect to the output pins of the light and infra-red detectors. The resistors used for signal amplification are connected in a non-inverting amplification loop, with a gain of 83. An LED is connected to the output of the connection sensor, to indicate free faces. As there are four daughter boards per module, four copies of what is shown in Figure A.1b are used and connected to the microcontroller.

Figure A.2a shows the power regulation section of the circuit. The system is powered by two 3.7 V rechargeable batteries, which are connected to the pins on the left. The batteries are connected to a DPDT (double pole double throw) switch that closes the circuit (powering the module), in the up position, and connects the batteries to the charging pins in the down position. The voltage regulator, converts the 7.4 V from the batteries into the 5 V used by the

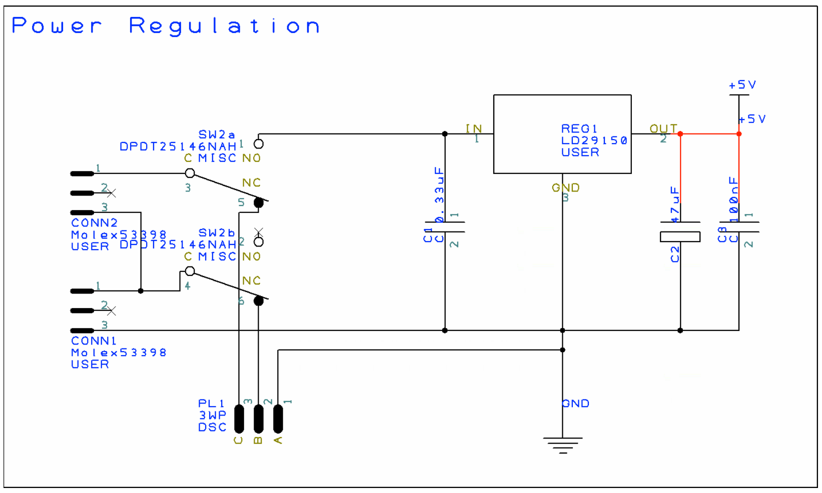


(a)

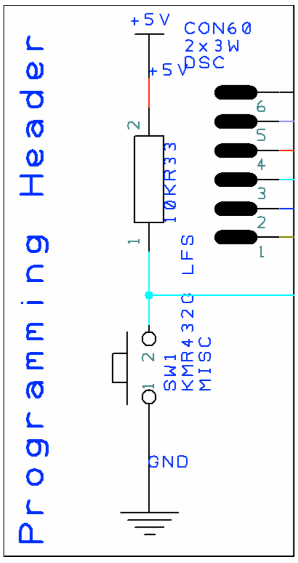


(b)

Fig. A.1 Microcontroller and daughter board connection schematics. (a) Section of the circuit showing the microcontroller and its connected pins. (b) Section of the circuit responsible for the connection with the daughter boards.

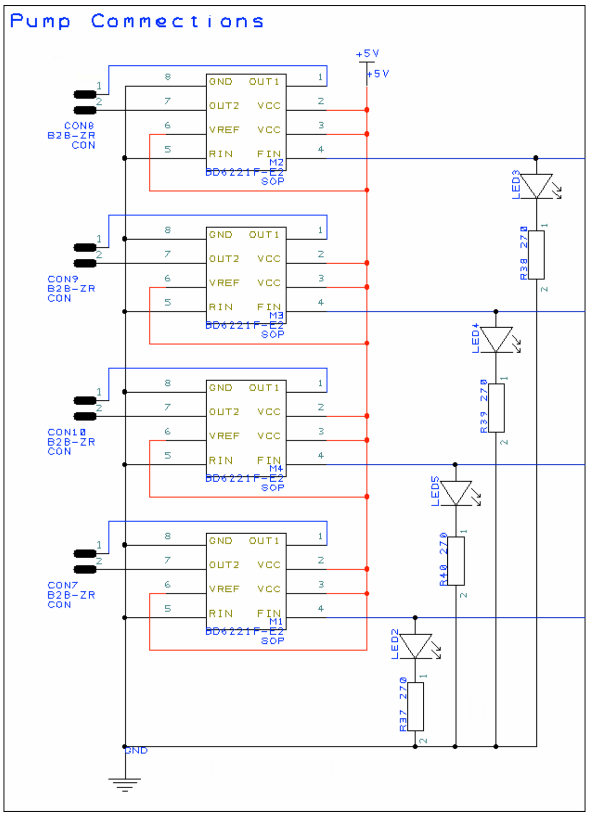


(a)

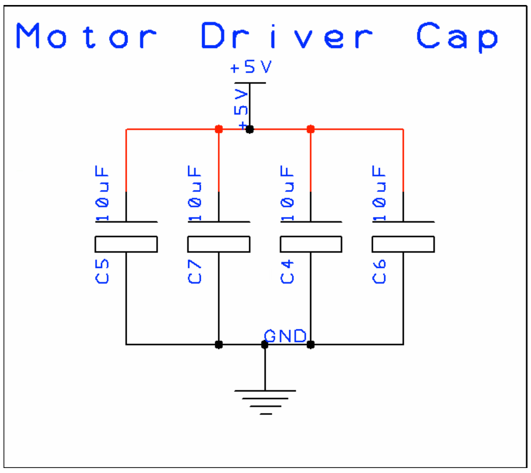


(b)

Fig. A.2 Schematics for (a) the power regulation and charging pins, and (b) the programming headed.



(a)



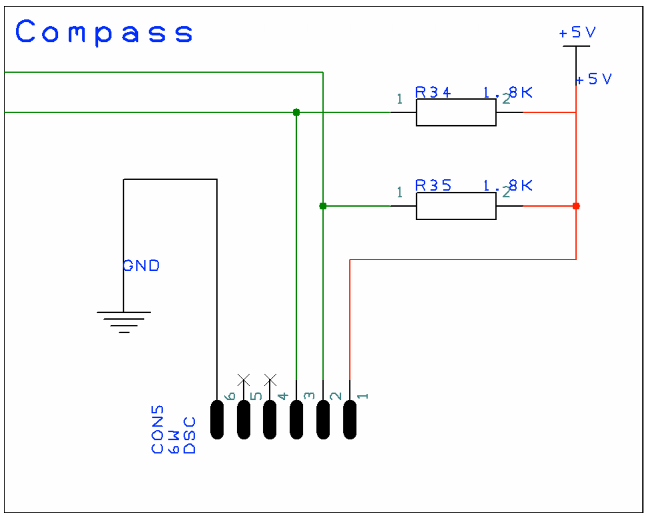
(b)

Fig. A.3 Schematics for the (a) connections between the motor drivers, the pumps and the microcontroller, and (b) capacitors used for smoothing the power signal used by the motor drivers.

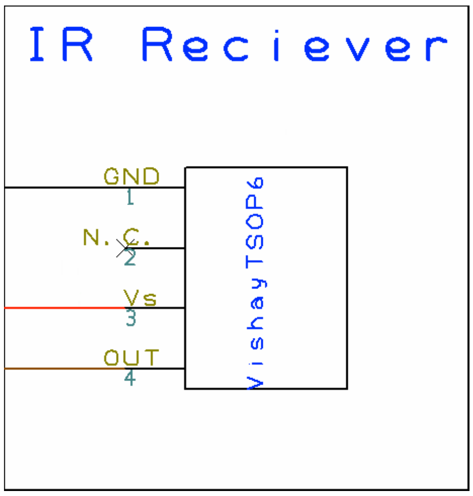
circuit components. The capacitors ensure correct filtration of the input and output voltage. Figure A.2b shows the connections to system's reset button and to the programming header of the microcontroller. These are the connections used for programming the module.

Figure A.3a shows the connectors to the pumps and the motor driving system. One half-bridge motor diver is used per pump. Additionally, LEDs are used to highlight the actuation status of each pump. Figure A.3b shows the capacitors that are used with the motor drivers. Even though these capacitors are only connected to the power rail and ground, they reduce the noise and smooth the signal coming out of the motor drivers to power the pumps. To ensure maximum effect for these capacitors, they have to be placed close to the motor drivers, even though they are not connected to them in any way.

Figure A.4a shows the connections of the compass. Two pull-up resistors are needed for the correct working of the compass. Figure A.4b shows the infra-red receiver used to start

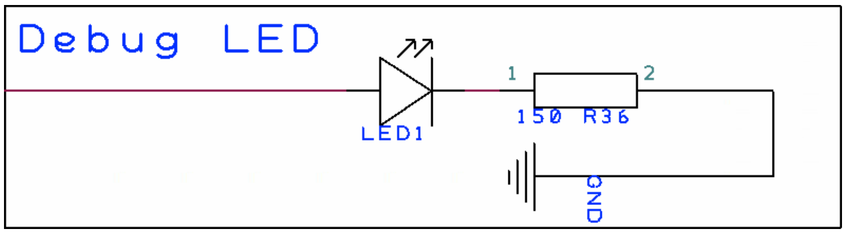


(a)

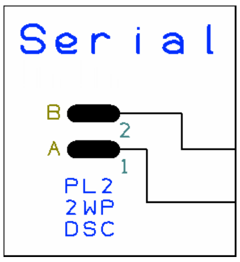


(b)

Fig. A.4 Schematics for the (a) the compass, the pull-up resistors and the connections to the microcontroller, and (b) infra-red receiver.



(a)



(b)

Fig. A.5 Schematics for the (a) LED of the system used for testing and debugging, and (b) serial communication pins.

the system. This component can be connected directly to the microcontroller without having to use any other components.

Figure A.5a shows the debug LED. This LED is used mainly for testing purposes. It cam also be used as an indicator of the state of the module; if the LED is off, the power is off, if the LED is on the system is on and on standby and if the LED is flashing the system is solving a task, for example. Figure A.5b shows the pins of the system for serial communication. These are used for communication with the computer. Figure A.6 shows the sensors placed in the daughter board and their connections.

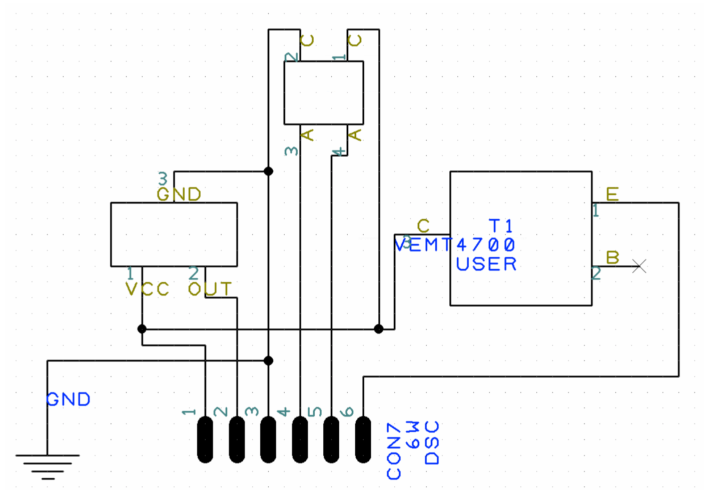


Fig. A.6 Section showing sensors placed in the daughter board and its connections. The rectangle on the left represents the Hall-effect sensor; the rectangle on the top represents the infra-red sensor; the square on the right represents the light sensor.

Appendix B

Static analysis - remaining cases

In Section 4.2, the number of actuation states that generate unique torques for a 3D convex MHP robot of dimensions $a_1\varepsilon \times a_2\varepsilon \times a_3\varepsilon$ was derived. However, for torques with a single non-zero component, only the derivation for the case where all robot side lengths are odd (i.e. $a_1, a_2, a_3 \in 2\mathbb{Z}^+ - 1$) was shown. Here, we show the derivation of the remaining cases.

Table 4.1 shows all seven remaining possible cases of robot side lengths. However, the results for some of the possible cases considered are similar to each other. For example, the results for the first two rows of Table 4.1 for torques applied along X_1 are identical, with the results for torques applied along X_2 and X_3 being swapped. Regarding the format of the number of unique torques, there are five unique cases:

- Two odd and one even side lengths, with non-zero torque component in the direction of one of the odd side length;
- Two odd and one even side lengths, with non-zero torque component in the direction of the even side length;
- One odd and two even side lengths, with non-zero torque component in the direction of the odd side length;
- One odd and two even side lengths, with non-zero torque component in the direction of the one of the even side length;
- Three even side lengths, with non-zero torque component in the direction of any side length.

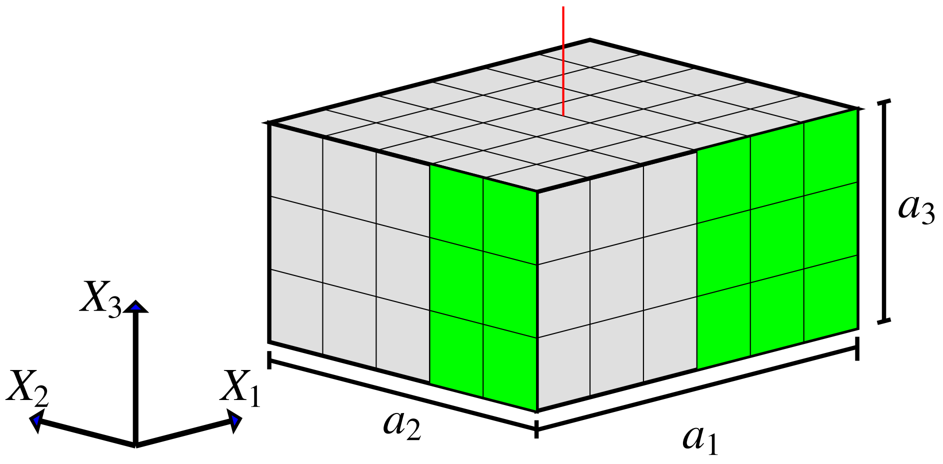


Fig. B.1 An a_1 by a_2 by a_3 robot (a_1 even, a_2 and a_3 odd) with active actuator configuration that achieves maximum torque along axis X_3. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

B.1 Robot with two odd and one even side lengths

Consider a 3D convex MHP robot, with $a_1 \in 2\mathbb{Z}^+$ and $a_2, a_3 \in 2\mathbb{Z}^+ - 1$.

B.1.1 Torque with non-zero component in the direction of an odd side length

The maximum torque along X_3 is obtained when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_3^\top(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{B.1}$$

The actuator firing configuration described in Equation (B.1) is illustrated in Figure B.1. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_3}(a_1,a_2,a_3) &= 2a_3\left[\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_1-1}{2}\right)+\left(1+2+\cdots+\frac{a_2-1}{2}\right)\right] \\ &= a_3\left(\frac{a_1^2}{4}+\frac{(a_2-1)(a_2+1)}{4}\right)=\frac{a_3(a_1^2+a_2^2-1)}{4}. \end{aligned} \tag{B.2}$$

Lemma 4. *For the case that $a_1 \in 2\mathbb{Z}^+$ and $a_2, a_3 \in 2\mathbb{Z}^+ - 1$, the robot can produce every half-integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,a_3),\dots,\tau^{X_3}_{max}(a_1,a_2,a_3)\}$, along the X_3 axis.*

Proof. We prove Lemma 4 by induction.

Base case: Consider the cases where $a_3 = 1$. If $a_1 = 2,4,\dots,k, \forall k \in 2\mathbb{Z}^+ \wedge a_2 = 1,3,\dots,k, \forall k \in 2\mathbb{Z}^+ - 1$, when the torque is applied along X_3, this corresponds to the 2D case presented in [115] and every half-integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,1),\dots,\tau^{X_3}_{max}(a_1,a_2,1)\}$ can be produced along X_3 axis.

Inductive step: We assume that Lemma 4 is correct for $a_3 = 1,3,\dots,k, \forall k \in 2\mathbb{Z}^+ - 1$. When $a_1 \times a_2 \times 1$ modules are added along X_3, symmetrically about the robot's center of mass, the positions of the existing modules does not change. This means that if the robot was able to produce a given torque for $a_3 = k$, that same torque can be produced for $a_3 = k+2$. For this reason, all half-integer torque values $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,k),\dots,\tau^{X_3}_{max}(a_1,a_2,k)\}$ can be produced.

Consider now the situation where the maximum torque $\tau^{X_3}_{max}(a_1,a_2,k+2) = \frac{(k+2)(a_1^2+a_2^2-1)}{4}$ is produced. Each of the two layers of $a_1 \times a_2 \times 1$ added modules is capable of producing every half-integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,1),\dots,\tau^{X_3}_{max}(a_1,a_2,1)\}$. So that the rotation axis does not change, the two sets of $a_1 \times a_2 \times 1$ added modules are capable of producing every integer torque $\tau^{X_3} \in \{0,1,\dots,2\tau^{X_3}_{max}(a_1,a_2,1)-1, 2\tau^{X_3}_{max}(a_1,a_2,1)\}$. The half-integer torque values $\tau^{X_3} \in \{\frac{1}{2},\frac{3}{2},\dots,2\tau^{X_3}_{max}(a_1,a_2,1)-\frac{1}{2}\}$ can be produced by activating/deactivating pumps p_{ej} for which $\mathbf{r}_{ej} = (\frac{a_1-1}{2},1,0)$ or $\mathbf{r}_{ej} = (1,\frac{a_2-1}{2},0)$. As such all half-integer torque values $\tau^{X_3} \in \{0,\dots,2\tau^{X_3}_{max}(a_1,a_2,1)\}$ can be produced.

Given that $\tau^{X_3}_{max}(a_1,a_2,k+2) - \tau^{X_3}_{max}(a_1,a_2,k) = 2\frac{a_1^2+a_2^2-1}{4} = 2\tau^{X_3}_{max}(a_1,a_2,1)$, the robot is able to produce every half-integer torque $\tau^{X^3} \in \{\tau^{X_3}_{max}(a_1,a_2,k+2) - \tau^{X_3}_{max}(a_1,a_2,k),\dots,\tau^{X_3}_{max}$ $(a_1,a_2,k+2)\}$. By symmetry, the robot is able to produce every half-integer torque $\tau^{X^3} \in \{-\tau^{X_3}_{max}(a_1,a_2,k+2),\dots,-\tau^{X_3}_{max}(a_1,a_2,k+2)+\tau^{X_3}_{max}(a_1,a_2,k)\}$. Therefore, the robot can produce every half-integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,k+2),\dots,\tau^{X_3}_{max}(a_1,a_2,k+2)\}$ and which concludes the inductive step. ■

Theorem 11. *The number of unique net torques that a 3D convex MHP robot with $a_1 \in 2\mathbb{Z}^+$ and $a_2, a_3 \in 2\mathbb{Z}^+ - 1$ is able to generate, with non-zero component along X_3 is $\eta_{\tau^{X_3}}(a_1,a_2,a_3) = a_3(a_1^2+a_2^2-1)+1$.*

Proof. The minimum and maximum net torques the robot can generate are $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$, respectively. From Lemma 4, the robot is able to produce all half-integer

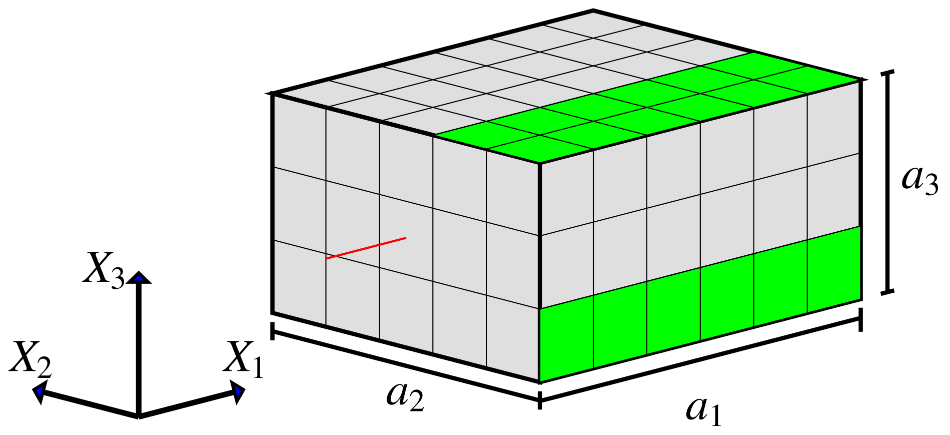


Fig. B.2 An a_1 by a_2 by a_3 robot (a_1 even, a_2 and a_3 odd) with active actuator configuration that achieves maximum torque along axis X_1. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

torques between $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$. Since each actuator is only able to produce an half-integer or integer torque, the robot can not produce any other net torque value. As a result, the number of unique net torques the robot ii able to produce is given by:

$$\eta_{\tau^{X_3}}(a_1,a_2,a_3) = 4\tau^{X_3}_{max}(a_1,a_2,a_3)+1 = a_3(a_1^2+a_2^2-1)+1. \tag{B.3}$$

■

B.1.2 Torque with non-zero component in the direction of an even side length

The maximum torque along X_1 is obtained when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_1^\top(\mathbf{x}_e \times \mathbf{u}_j) < 0 \land c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{B.4}$$

The actuator firing configuration described in Equation (B.4) is illustrated in Figure B.2. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_1}(a_1,a_2,a_3) &= 2a_1\left[\left(1+2+\cdots+\frac{a_2-1}{2}\right)+\left(1+2+\cdots+\frac{a_3-1}{2}\right)\right] \\ &= a_3\left(\frac{(a_2-1)(a_2+1)}{4}+\frac{(a_3-1)(a_3+1)}{4}\right)=\frac{a_1(a_2^2+a_3^2-2)}{4}. \qquad \text{(B.5)} \end{aligned}$$

Lemma 5. *For the case that $a_1 \in 2\mathbb{Z}^+$ and $a_2, a_3 \in 2\mathbb{Z}^+-1$, the robot can produce every even torque $\tau^{X_1} \in \{-\tau_{max}^{X_1}(a_1,a_2,a_3),\ldots,\tau_{max}^{X_1}(a_1,a_2,a_3)\}$, along the X_1 axis.*

Proof. We prove Lemma 5 by induction.

Base case: Consider the cases where $a_1 = 2$. If $a_2 = 1 \wedge a_3 = 1$, the robot consists of two modules stacked along X_1; from Equation (4.2), no torque can be produced along X_1. If $a_2 = 1,3,\ldots,k, \forall k \in 2\mathbb{Z}^+ \wedge a_3 = 1,3,\ldots,k, \forall k \in 2\mathbb{Z}^+-1$, when the torque is applied along X_3, this corresponds to two layers of $1\times a_2 \times a_3$. From Lemma 1, every even torque $\tau^{X_1} \in \{-2\tau_{max}^{X_1}(1,a_2,a_3), -2\tau_{max}^{X_1}(1,a_2,a_3)+2,\ldots,2\tau_{max}^{X_1}(1,a_2,a_3)-2, 2\tau_{max}^{X_1}(1,a_2,a_3)\}$ can be produced along X_1 axis.

Inductive step: We assume that Lemma 5 is correct for $a_1 = 2,4,\ldots,k, \forall k \in 2\mathbb{Z}^+$. When $1\times a_2\times a_3$ modules are added along X_1, symmetrically about the robot's center of mass, the positions of the existing modules does not change. This means that if the robot was able to produce a given torque for $a_1 = k$, that same torque can be produced for $a_1 = k+2$. For this reason, all even torque values $\tau^{X_1} \in \{-\tau_{max}^{X_1}(k,a_2,a_3),\ldots,\tau_{max}^{X_1}(k,a_2,a_3)\}$ can be produced.

Consider now the situation where the maximum torque $\tau_{max}^{X_1}(k+2,a_2,a_3) = \frac{(k+2)(a_2^2+a_3^2-2)}{4}$ is produced. Each of the two layers of $1\times a_2\times a_3$ added modules is capable of producing every integer torque $\tau^{X_1} \in \{-\tau_{max}^{X_1}(1,a_2,a_3),\ldots,\tau_{max}^{X_1}(1,a_2,a_3)\}$. So that the rotation axis does not change, the two sets of $1\times a_2\times a_3$ added modules are capable of producing every even torque $\tau^{X_1} \in \{0,2,\ldots,2\tau_{max}^{X_1}(1,a_2,a_3)-2, 2\tau_{max}^{X_1}(1,a_2,a_3)\}$.

Given that $\tau_{max}^{X_1}(k+2,a_2,a_3) - \tau_{max}^{X_1}(k,a_2,a_3) = 2\frac{a_2^2+a_3^2-2}{4} = 2\tau_{max}^{X_1}(1,a_2,a_3)$, the robot is able to produce every even torque $\tau^{X^1} \in \{\tau_{max}^{X_1}(k+2,a_2,a_3) - \tau_{max}^{X_1}(k,a_2,a_3),\ldots,\tau_{max}^{X_1}(k+2,a_2,a_3)\}$. By symmetry, the robot is able to produce every even torque $\tau^{X^1} \in \{-\tau_{max}^{X_2}(k+2,a_2,a_3),\ldots,-\tau_{max}^{X_1}(k,a_2,a_3)+\tau_{max}^{X_1}(k,a_2,a_3)\}$. Therefore, the robot can produce every even torque $\tau^{X_1} \in \{-\tau_{max}^{X_1}(k+2,a_2,a_3),\ldots,\tau_{max}^{X_1}(k+2,a_2,a_3)\}$ which concludes the inductive step. ■

Theorem 12. *The number of unique net torques that a 3D convex MHP robot with $a_1 \in 2\mathbb{Z}^+$ and $a_2, a_3 \in 2\mathbb{Z}^+ - 1$ is able to produce, with non-zero component along X_1 is $\eta_{\tau^{X_1}}(a_1,a_2,a_3) = a_1\frac{(a_2^2+a_3^2-2)}{4}+1$.*

Proof. The minimum and maximum net torques the robot can generate are $-\tau^{X_1}(a_1,a_2,a_3)$ and $\tau^{X_1}(a_1,a_2,a_3)$, respectively. From Lemma 5, the robot is able to produce all even torque between $-\tau^{X_1}(a_1,a_2,a_3)$ and $\tau^{X_1}(a_1,a_2,a_3)$. Since each actuator is only able to produce an integer torque, the robot can not produce any other net torque value. As a result, the number of unique net torques the robot is able to produce is given by:

$$\eta_{\tau^{X_1}}(a_1,a_2,a_3) = \tau_{max}^{X_1}(a_1,a_2,a_3)+1 = a_1\frac{(a_2^2+a_3^2-2)}{4}+1. \tag{B.6}$$

■

B.2 Robot with one odd and two even side lengths

Consider a 3D convex MHP robot, with $a_1, a_2 \in 2\mathbb{Z}^+$ and $a_3 \in 2\mathbb{Z}^+ - 1$.

B.2.1 Torque with non-zero component in the direction of an odd side length

The maximum torque along X_3 is obtained when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_3^\top(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{B.7}$$

The actuator firing configuration described in Equation (B.7) is illustrated in Figure B.3. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_3}(a_1,a_2,a_3) &= 2a_3\left[\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_1-1}{2}\right)+\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_2-1}{2}\right)\right] \\ &= a_3\left(\frac{a_1^2}{4}+\frac{a_2^2}{4}\right) = \frac{a_3(a_1^2+a_2^2)}{4}. \end{aligned} \tag{B.8}$$

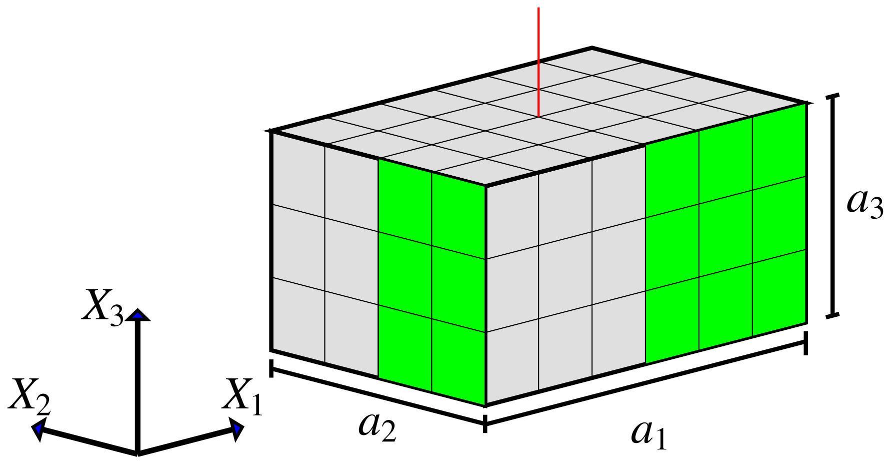


Fig. B.3 An a_1 by a_2 by a_3 robot (a_1 and a_2 even, a_3 odd) with active actuator configuration that achieves maximum torque along axis X_3. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

Lemma 6. *For the case that $a_1, a_2 \in 2\mathbb{Z}^+$ and $a_3 \in 2\mathbb{Z}^+ - 1$, the robot can produce every half-integer torque $\tau^{X_3} \in \{-\tau^{X_3}_{max}(a_1,a_2,a_3), \ldots, \tau^{X_3}_{max}(a_1,a_2,a_3)\}$, along the X_3 axis.*

Proof. The proof is omitted, as it is identical to the proof of Lemma 4. ■

Theorem 13. *The number of unique net torques that a 3D convex MHP robot with $a_1, a_2 \in 2\mathbb{Z}^+$ and $a_3 \in 2\mathbb{Z}^+ - 1$ is able to generate, with non-zero component along X_3 is $\eta_{\tau^{X_3}}(a_1,a_2,a_3) = a_3(a_1^2 + a_2^2) + 1$.*

Proof. The minimum and maximum net torques the robot is able to generate are $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$, respectively. From Lemma 6, the robot is able to produce all half-integer torque between $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$. Since each actuator is only able to produce an half-integer torque, the robot can not produce any other net torque value. As a result, the number of unique net torques the robot is able to produce is given by:

$$\eta_{\tau^{X_3}}(a_1,a_2,a_3) = 4\tau^{X_3}_{max}(a_1,a_2,a_3) + 1 = a_3(a_1^2 + a_2^2) + 1. \tag{B.9}$$

■

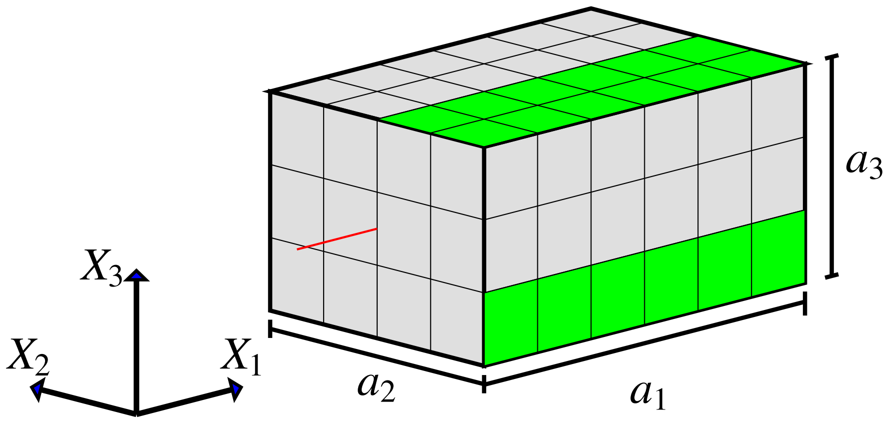


Fig. B.4 An a_1 by a_2 by a_3 robot (a_1 and a_2 even, a_3 odd) with active actuator configuration that achieves maximum torque along axis X_1. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

B.2.2 Torque with non-zero component in the direction of an even side length

The maximum torque along X_1 is obtained when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_1^\top (\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{B.10}$$

The actuator firing configuration described in Equation (B.10) is illustrated in Figure B.2. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_1}(a_1,a_2,a_3) &= 2a_1\left[\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_2-1}{2}\right)+\left(1+2+\cdots+\frac{a_3-1}{2}\right)\right] \\ &= a_3\left(\frac{a_2^2)}{4}+\frac{(a_3-1)(a_3+1)}{4}\right) = \frac{a_1(a_2^2+a_3^2-1)}{4}. \end{aligned} \tag{B.11}$$

Lemma 7. *For the case that $a_1, a_2 \in 2\mathbb{Z}^+$ and $a_3 \in 2\mathbb{Z}^+ - 1$, the robot can produce every integer torque $\tau^{X_1} \in \{-\tau_{max}^{X_1}(a_1,a_2,a_3),\ldots,\tau_{max}^{X_1}(a_1,a_2,a_3)\}$, along the X_1 axis.*

Proof. We prove Lemma 7 by induction.

Base case: Consider the cases where $a_1 = 2$. If $a_2 = 2,4,\ldots,k, \forall k \in 2\mathbb{Z}^+ \wedge a_3 = 1,3,\ldots,k, \forall k \in 2\mathbb{Z}^+ - 1$, when the torque is applied along X_1, this corresponds to two layers of $1 \times a_2 \times a_3$. Each layer is capable of producing every half-integer torque $\tau^{X_1} \in \{-\tau^{X_1}_{max}(1,a_2,a_3),\dots,\tau^{X_1}_{max}(1,a_2,a_3)\}$. From Lemma 5 every integer torque $\tau^{X_1} \in \{-2\tau^{X_1}_{max}(1, a_2,a_3),\dots,2\tau^{X_1}_{max}(1,a_2,a_3)\}$ can be produced along X_1 axis.

Inductive step: We assume that Lemma 7 is correct for $a_1 = 2,4,\dots,k, \forall k \in 2\mathbb{Z}^+$. When $1 \times a_2 \times a_3$ modules are added along X_1, symmetrically about the robot's center of mass, the positions of the existing modules does not change. This means that if the robot was able to produce a given torque for $a_1 = k$, that same torque can be produced for $a_1 = k+2$. For this reason, all integer torque values $\tau^{X_3} \in \{-\tau^{X_1}_{max}(k,a_2,a_3),\dots,\tau^{X_1}_{max}(k,a_2,a_3)\}$ can be produced.

Consider now the situation where the maximum torque $\tau^{X_1}_{max}(k+2,a_2,a_3) = \frac{(k+2)(a_2^2+a_3^2-1)}{4}$ is produced. Each of the two layers of $1 \times a_2 \times a_3$ added modules is capable of producing every half-integer torque $\tau^{X_1} \in \{-\tau^{X_1}_{max}(1,a_2,a_3),\dots,\tau^{X_1}_{max}(1,a_2,a_3)\}$. So that the rotation axis does not change, the two sets of $1 \times a_2 \times a_3$ added modules are capable of producing every integer torque $\tau^{X_1} \in \{0,1,\dots,2\tau^{X_1}_{max}(1,a_2,a_3)-1,2\tau^{X_1}_{max}(1,a_2,a_3)\}$.

Given that $\tau^{X_1}_{max}(k+2,a_2,a_3) - \tau^{X_1}_{max}(k,a_2,a_3) = 2\frac{a_2^2+a_3^2-1}{4} = 2\tau^{X_1}_{max}(1,a_2,a_3)$, the robot is able to produce every integer torque $\tau^{X^1} \in \{\tau^{X_1}_{max}(k+2,a_2,a_3) - \tau^{X_1}_{max}(k,a_2,a_3),\dots,\tau^{X_1}_{max}(k+2,a_2,a_3)\}$. By symmetry, the robot is able to produce every integer torque $\tau^{X^1} \in \{-\tau^{X_2}_{max}(k+2,a_2,a_3),\dots,-\tau^{X_1}_{max}(k,a_2,a_3)+\tau^{X_1}_{max}(k,a_2,a_3)\}$. Therefore, the robot can produce every even torque $\tau^{X_1} \in \{-\tau^{X_1}_{max}(k+2,a_2,a_3),\dots,\tau^{X_1}_{max}(k+2,a_2,a_3)\}$ which concludes the inductive step. ■

Theorem 14. *The number of unique net torques that a 3D convex MHP robot with $a_1 \in 2\mathbb{Z}^+$ and $a_2,a_3 \in 2\mathbb{Z}^+ - 1$ is able to generate, when applied along X_1 is $\eta_{\tau^{X_1}}(a_1,a_2,a_3) = a_1\frac{(a_2^2+a_3^2-1)}{2}+1$.*

Proof. The minimum and maximum net torques the robot can generate are $-\tau^{X_1}(a_1,a_2,a_3)$ and $\tau^{X_1}(a_1,a_2,a_3)$, respectively. From Lemma 7, the robot is able to produce all even torque between $-\tau^{X_1}(a_1,a_2,a_3)$ and $\tau^{X_1}(a_1,a_2,a_3)$. Since each actuator is only able to produce an half-integer or integer torque, the robot can not produce any other net torque value. As a result, the number of unique net torques the robot is able to produce is given by:

$$\eta_{\tau^{X_1}}(a_1,a_2,a_3) = 2\tau^{X_1}_{max}(a_1,a_2,a_3)+1 = a_1\frac{(a_2^2+a_3^2-1)}{2}+1. \tag{B.12}$$

■

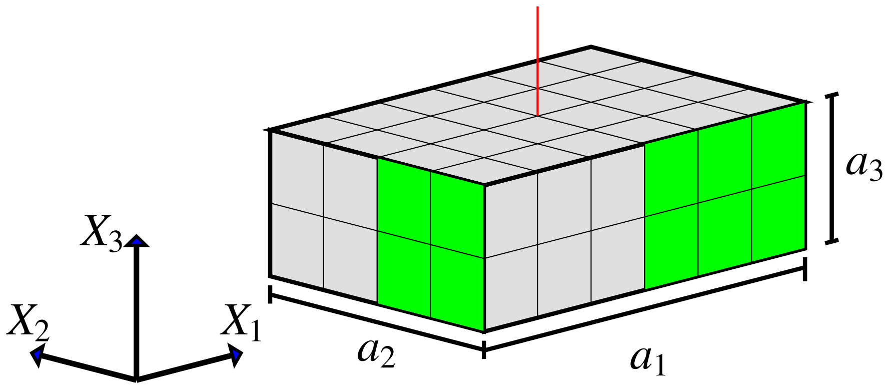


Fig. B.5 An a_1 by a_2 by a_3 robot (all side lengths even) with active actuator configuration that achieves maximum torque along axis X_3. Green faces represent the faces with active actuators. The rotation axis is represented by the red line, going through the robot face. On the non-visible faces the actuators in mirrored positions are firing.

B.3 Robot with three even side lengths

Consider a 3D convex MHP robot, with $a_1, a_2, a_3 \in 2\mathbb{Z}^+$. The maximum torque along X_3 is obtained when:

$$p_{ej} = \begin{cases} 1, & \mathbf{n}_3^\top(\mathbf{x}_e \times \mathbf{u}_j) < 0 \wedge c_{ej} = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{B.13}$$

The actuator firing configuration described in Equation (B.7) is illustrated in Figure B.3. Using Equation (4.2) for this actuator firing configuration, we obtain a net torque of:

$$\begin{aligned} \tau_{max}^{X_3}(a_1,a_2,a_3) &= 2a_3\left[\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_1-1}{2}\right)+\left(\frac{1}{2}+\frac{3}{2}+\cdots+\frac{a_2-1}{2}\right)\right] \\ &= a_3\left(\frac{a_1^2}{4}+\frac{a_1^2}{4}\right) = \frac{a_3(a_1^2+a_2^2)}{4}. \end{aligned} \tag{B.14}$$

Lemma 8. *For the case that $a_1, a_2, a_3 \in 2\mathbb{Z}^+$ the robot can produce every integer torque $\tau^{X_1} \in \{-\tau_{max}^{X_1}(a_1,a_2,a_3), \ldots, \tau_{max}^{X_1}(a_1,a_2,a_3)\}$, along the X_1 axis.*

Proof. The proof is omitted, as it is identical to the proof of Lemma 7. ■

Theorem 15. *The number of unique net torques that a 3D convex MHP robot with $a_1, a_2, a_3 \in 2\mathbb{Z}^+$ is able to generate, when applied along X_3 is $\eta_{\tau^{X_3}}(a_1,a_2,a_3) = a_3\frac{(a_1^2+a_2^2)}{2}+1$.*

Proof. The minimum and maximum net torques the robot can generate are $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$, respectively. From Lemma 8, the robot is able to produce all integer torque between $-\tau^{X_3}(a_1,a_2,a_3)$ and $\tau^{X_3}(a_1,a_2,a_3)$. Since each actuator is only able to produce an half-integer torque, the robot can not produce any other net torque vale. As a result, the number of unique net torques the robot is able to produce is given by:

$$\eta_{\tau^{X_3}}(a_1,a_2,a_3) = 2\tau^{X_3}_{max}(a_1,a_2,a_3)+1 = a_3\frac{(a_1^2+a_2^2)}{2}+1. \tag{B.15}$$

■

Appendix C

Hybrid pose controllers

In here, a controller solution for 2D and 3D, that makes use of a single shared power line, but performs some translation during the rotation process is described.

C.1 `2D-1SP-H` Hybrid controller

In Section 4.4.1, we proposed controller solutions that use $M \in \{0,1,2\}$ shared power lines. Additionally, the controllers that use shared power lines are able to perform the rotation and translation components of the movement separately. When presenting the `2D-2SP` controller, we state that two bits of shared information is the minimum required for the robot to differentiate between translation, clockwise rotation and counter-clockwise rotation. However, this is only correct if it is required that the robot does not translate while rotating. If some translation is allowed, by combining the rotation component of the `2D-0SP-V1` and `2D-0SP-V2` controllers and the translation component of the `2D-1SP` controller, a controller using only a single shared power line that is capable of both clockwise and counter-clockwise rotation can be obtained. Additionally, this controller is guaranteed to succeed, if the initial distance condition is valid, that is, if the robot is able to rotate on the spot without colliding with the goal.

C.1.1 Control policy

As previously mention this controller requires a single shared power line (i.e., $M = 1$). The shared power line activation activation policy, $f_{b,1}(\mathbf{c}, \mathbf{d})$ is identical to the one used by the

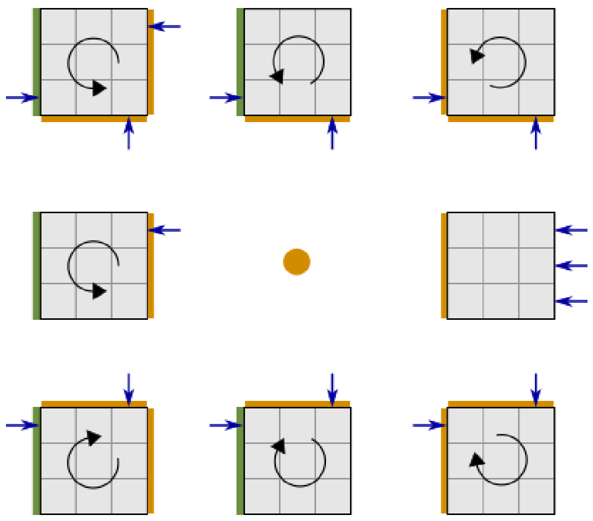


Fig. C.1 Decentralised pose control without communication (`2D-1SP-H` controller). Shown are eight scenarios of a robot that needs to touch the goal (orange point) with its preferred face (face 1). The goal is visible from any module face marked with orange. Up to three pumps are active, causing the robot to rotate. However, as the pumps active for each face are not symmetrical, some undesired translation, away from the goal, occurs.

`2D-1SP` controller (Equation 4.46).The control policy for this controller variant (`2D-1SP-H` controller), $f_p(\mathbf{c},\mathbf{d},\mathbf{b})$, is given by:

$$\begin{aligned} p_1 &= \bar{c}_1\bar{c}_3 d_3 &&\vee \bar{c}_1\bar{c}_4 d_4, && \text{(C.1)}\\ p_2 &= \bar{c}_2\bar{c}_4\bar{d}_4 && &&\vee \bar{c}_2\bar{b}_2, \quad \text{(C.2)}\\ p_3 &= \bar{c}_3\bar{c}_2 d_3, && && \text{(C.3)}\\ p_4 &= && \bar{c}_4\bar{c}_2 d_4. && \text{(C.4)} \end{aligned}$$

In the alignment of the equations, the first column is responsible for the counter-clockwise rotation, the second column for the clockwise rotation and the last column for the translation of the robot. With this control policy, the robot only actively translates once it is correctly oriented towards the goal. If any module face, other than the preferred face, detects the goal, the robot rotates. However, since the decisions on rotation are made locally by the modules, some undesired translation occurs. Figure C.1 shows the different pump activation scenarios possible.

C.1.2 Controller analysis

As the torque applied on the robot is non-zero, if the robot is not in the preferred orientation, from Theorem 5 the robot is able to reach the desired orientation in finite time. If the robot is in the preferred orientation, from Lemma 2, the robot is able to reach the target in finite time. Similarly to the 2D-0SP-V1 and 2D-0SP-V2 controllers, as there is some translation during the rotation movement, to guarantee the success of the controller, it is required to show that there is no collision between the robot and the goal during rotation.

From the actuation states depicted in Figure C.1, it can be observed that the undesired translation moves the robot away from the target. As a result, if the robot is able to rotate in place in the initial position, the robot is guaranteed to complete the task using this controller. This means that,the initial distance condition is sufficient to guarantee the success of the controller.

C.2 3D-1SP-H Hybrid controller

In Section 4.4.2 we stated that a single bit of shared information was not sufficient to guarantee that the robot completes the task, leading to a minimum of two bits of shared information being required. However, that is only correct if no translation can occur while the robot is rotating. Additionally, using two bits of information, the robot is only able to rotate in a single direction along each of the axis. If some translation is allowed, by combining the rotation component of the 3D-0SP-V1 and 3D-0SP-V2 controllers and the translation component of the 3D-2SP controller, a controller using only a single shared power line that is capable of both clockwise and counter-clockwise rotation, along each of the axis, can be obtained. Additionally, this controller is guaranteed to succeed, if the initial distance condition is valid, that is, if the robot is able to rotate on the spot without colliding with the goal.

C.2.1 Control policy

As previously mention this controller requires a single shared power line (i.e., $M = 1$). The shared power line activation activation policy, $f_{b,1}(\mathbf{c}, \mathbf{d})$ is given by:

$$b_1 = d_2 \vee d_3 \vee d_4 \vee d_5 \vee d_6. \tag{C.5}$$

The control policy for this controller variant (`3D-1SP-H` controller), $f_p(\mathbf{c},\mathbf{d},\mathbf{b})$, is given by:

$$\begin{aligned}
p_1 &= \bar{c}_1\bar{c}_3 d_3 \vee \bar{c}_1\bar{c}_4 d_4 \vee \bar{c}_1\bar{c}_5 d_5 \vee \bar{c}_1\bar{c}_6 d_6, && \text{(C.6)}\\
p_2 &= \bar{c}_2\bar{c}_4\bar{d}_4 \qquad\qquad\qquad\qquad \vee \bar{c}_2\bar{b}_2, && \text{(C.7)}\\
p_3 &= \bar{c}_3\bar{c}_2 d_3, && \text{(C.8)}\\
p_4 &= \qquad \bar{c}_4\bar{c}_2 d_4, && \text{(C.9)}\\
p_5 &= \qquad\qquad \bar{c}_5\bar{c}_2 d_5, && \text{(C.10)}\\
p_6 &= \qquad\qquad\qquad \bar{c}_6\bar{c}_2 d_6. && \text{(C.11)}
\end{aligned}$$

In the alignment of the equations, the first column is responsible for the counter-clockwise rotation along X_3^{local}, the second column for the clockwise rotation along X_3^{local}, the third column is responsible for the counter-clockwise rotation along X_2^{local}, the second column for the clockwise rotation along X_2^{local}, and the last column for the translation of the robot. With this control policy, the robot only actively translates once it is correctly oriented towards the goal. If any module face, other than the preferred face, detects the goal, the robot rotates. However, since the decisions on rotation are made locally by the modules, some undesired translation occurs.

C.2.2 Controller analysis

As the torque applied on the robot is non-zero, if the robot is not in the preferred orientation, from Theorem 8 the robot is able to reach the desired orientation in finite time. If the robot is in the preferred orientation, from Lemma 3, the robot is able to reach the target in finite time. Similarly to the `3D-0SP-V1` and `3D-0SP-V2` controllers, as there is some translation during the rotation movement, to guarantee the success of the controller, it is required to show that there is no collision between the robot and the goal during rotation.

As with the 2D version of this controller, only faces that detect the goal (and possibly the preferred face) have active actuators the undesired translation moves the robot away from the target. As a result, if the robot is able to rotate in place in the initial position, the robot is guaranteed to complete the task using this controller. This means that,the initial distance condition is sufficient to guarantee the success of the controller.

C.3 Simulation results

In this section, we present the results of physics simulations, using the `3D-1SP-H` controller. The setup for the simulations is the same as the one described in Section 5.2.1. We start by performing an analysis of the movement of the robot during a successful trial. Additionally, the performance of the controller, in terms of success rate and trial completion time is compared to the other proposed controllers (and centralised controller) for short, medium and long distances.

C.3.1 Movement analysis

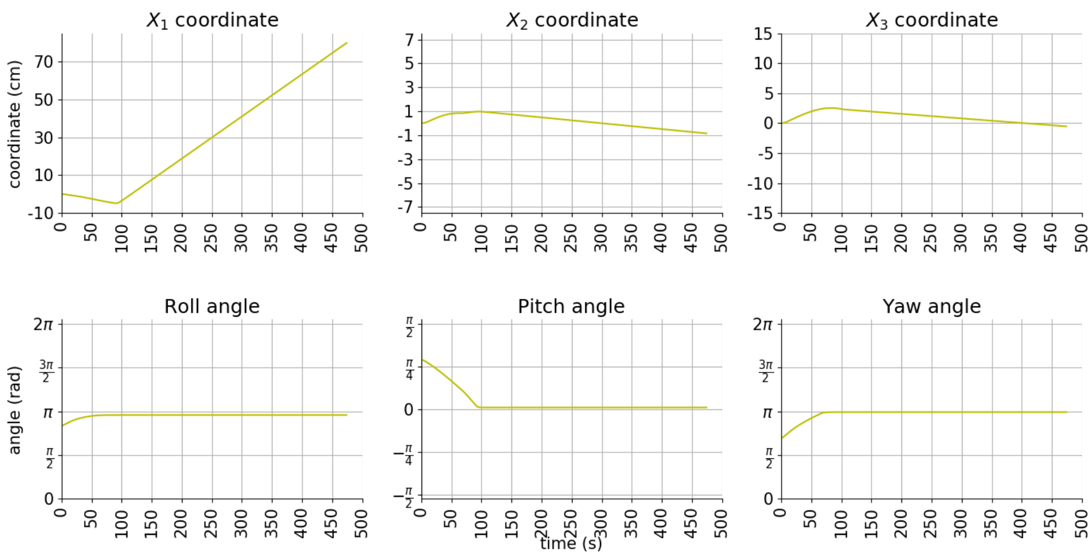
Figure C.2a shows the position and orientation (in Euler angles) of the robot over time, for the duration of the trial. At the start of the trial the robot is positioned at $(0,0,0)$ with a non-preferred orientation. The goal is positioned at $(85.5,0,0)$, and the desired orientation is $(\psi = \pi, \theta = 0, \phi)$, where ψ, θ and ϕ correspond to the yaw, pith and roll angles[1]. In order to reach the goal, the robot is only required to translate along X_1^{local}. The robot takes approximately 100 s to reach the desired orientation, from the initial orientation. In that time period, the distance between the robot and the goal increases. Once the robot is correctly oriented, the robot translates towards the goal. As the robot moved from the initial position during rotation, translation occurs along all axis.

Figure C.2b shows the trajectory of the robot in two planes: X_1X_2 plane (right view) and X_1X_3 plane (top view). The robot moves away from the target from the initial position, while it rotates, in an arching trajectory. Once the rotation is completed (preferred orientation has been reached) the robot translates in a strait line towards the goal. The behaviour of the robot when using this controller is in accordance with what is expected from the controller analysis.

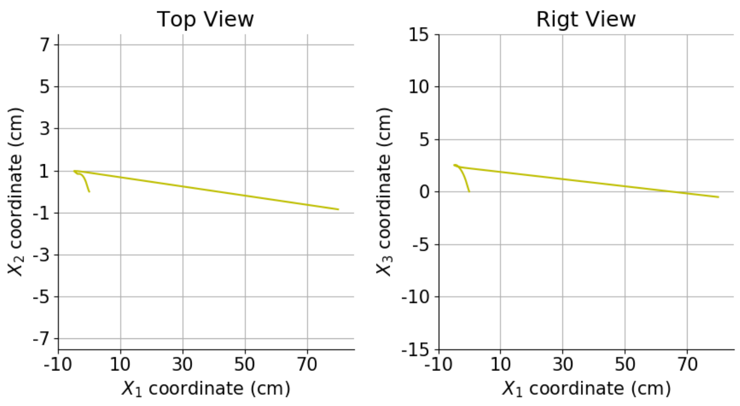
C.3.2 Comparison with the controllers proposed in Chapter 4

Figure C.3 compares the performance of the `3D-1SP-H` controller with the controllers proposed in Chapter 4 for short, medium and long distances. The `3D-1SP-H` controller consistently outperforms the `3D-2SP` controller. For all three distances, the `3D-1SP-H` controller presents a nearly perfect success rate (98% for short distances and 100% for medium and

[1]Note that the desired orientation is dependent only of the yaw and pitch angles.



(a)



(b)

Fig. C.2 Movement of the robot over a successful trial using the `3D-1SP-H` controller. (a) Robot's position and orientation over time. (b) Robot's trajectory when observed from the top (left) and right (right) views.

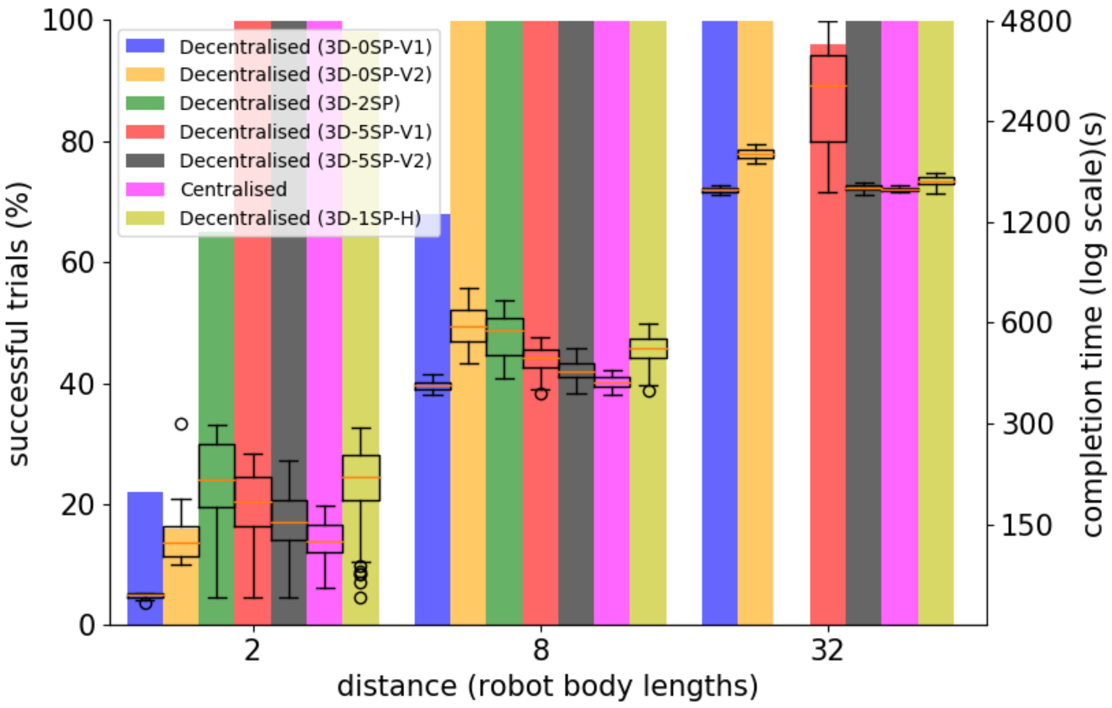


Fig. C.3 Performance of a 10 × 10 × 10 robot when the goal is initially a short, medium, or large distance away from the robot (100 observations per setting). The controllers are (from the left to the right): the decentralized controllers, `3D-0SP-V1`, `3D-0SP-V2`, `3D-2SP`, `3D-5SP-V1`, `3D-5SP-V2`, the centralized controller and decentrlised hybrid controller `3D-1SP-H`. Bars represent the percentage of successful trials. Box plots represent completion times (successful trials only).

long distances), while presenting similar completion times. The `3D-2SP` controller is only capable of generating rotations along one of the axis of the robot's reference frame at the time, the `3D-1SP-H` controller is capable of generating rotations along two of the axis of the robot's reference frame simultaneously. Additionally, the `3D-1SP-H` controller is capable of generating rotations along both directions, while the `3D-2SP` controller can only generate rotations along one. As a result, the `3D-1SP-H` controller generates a shorter rotation towards the preferred orientation than the `3D-2SP` controller. While the terminal angular velocities along each of the robot's axes is smaller for the `3D-1SP-H` controller (less modules contributing to the rotation), as the rotation is significantly shorter, the rotation time is also shorter. The shorter rotation times justify the better success rates for short distances, as the high drag coefficient value does not impact as much as for the `3D-2SP` controller. For longer distances, the `3D-1SP-H` controller is able to compensate for the overshoot, by rotating both clockwise and counter-clockwise. The similarities between the completion times are justified by a longer translation for the 3D-1SP-H controller.

While performing better than the 3D-2SP controller, the 3D-1SP-H controller is outperformed by the 3D-5SP-V2 and centralised controllers for all distances, and by the 3D-5SP-V1 controller for short and medium distances. The 3D-5SP-V2 controller performs the same rotations as the 3D-1SP-H controller. However, as more modules contribute to the rotation, the preferred orientation is reached faster. Additionally, as the 3D-1SP-H controller performs a longer translation, the translation time is also shorter for the 3D-5SP-V2 controller. When compared to the 3D-5SP-V1 controller, the 3D-1SP-H controller presents slightly higher completion times (except for long distances). While the rotations for the 3D-5SP-V1 controller are longer, the terminal angular velocities long each axis are higher. This leads to similar rotation times. With longer translations for the 3D-1SP-H controller, the 3D-5SP-V1 controller completes the task faster. Long distances represent an exception, due to the overshoot the 3D-5SP-V1 controller suffers when rotating along X_1^{local}.

Compared to the 3D-0SP-V2 controller, the 3D-1SP-H controller presents better results in terms of either success rate (short distances) or completion time (medium and long distances). Additionally, it outperforms the 3D-0SP-V1 controller for short and medium distances (higher success rate). However, for long distances, the 3D-0SP-V1 controller performs better, due to its shorter completion times.

Appendix D

Behaviour comparison for controllers using more/less shared power lines than the predictied minimum required

Table D.1 compares the behaviours generated by using four and five bits of information for the `3D-5SP-V1` controller. It can be seen that by combining the sensing status of the face with outward normal parallel to X_1^{local} with the sensing status of adjacent face changes the rotation axis of some of the global sensing states.

Table D.2 shows the behaviours generated by using six bits of information for the `3D-5SP-V2` controller. It can be seen that the last eight situations for which the extra shared power line is active ($b_1 = 1$) are identical to the first eight situations for which the extra shared power line is inactive ($b_1 = 0$). Table D.3 shows the same results for the `3D-5SP-V1` controller. As a result, adding an extra shared power line, would bring no improvements to both controllers, while increasing their complexity.

Table D.1 Behaviours obtained by using 5 and 4-bit of shared information in the `3D-5SP-V1` controller. For the 5-bit case, the value of the shared power lines is given by $b_j = d_j$. For the 5-bit case, the value of the shared power lines is given by $b_3 = d_2 \vee d_3$, $b_4 = d_4$, $b_5 = d_5$, $b_6 = d_6$.

5-bit shared power line						4-bit shared power line				
b_2	b_3	b_4	b_5	b_6	Behaviour	b_3	b_4	b_5	b_6	Behaviour
0	0	0	0	0	T	0	0	0	0	T
0	0	0	0	1	CCW X_2^{local}	0	0	0	1	CCW X_2^{local}
0	0	0	1	0	CW X_2^{local}	0	0	1	0	CW X_2^{local}
0	0	1	0	0	CW X_3^{local}	0	1	0	0	CW X_3^{local}
0	0	1	0	1	CW X_1^{local}	0	1	0	1	CW X_1^{local}
0	0	1	1	0	CW X_1^{local}	0	1	1	0	CW X_1^{local}
0	1	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
0	1	0	0	1	CCW X_1^{local}	1	0	0	1	CCW X_1^{local}
0	1	0	1	0	CCW X_1^{local}	1	0	1	0	CCW X_1^{local}
1	0	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
1	0	0	0	1	**CCW** X_2^{local}	1	0	0	1	**CCW** X_1^{local}
1	0	0	1	0	**CW** X_2^{local}	1	0	1	0	**CCW** X_2^{local}
1	0	1	0	0	CW X_3^{local}	1	1	0	0	CW X_3^{local}
1	0	1	0	1	CW X_1^{local}	1	1	0	1	CW X_1^{local}
1	0	1	1	0	CW X_1^{local}	1	1	1	0	CW X_1^{local}
1	1	0	0	0	CCW X_3^{local}	1	0	0	0	CCW X_3^{local}
1	1	0	0	1	CCW X_1^{local}	1	0	0	1	CCW X_1^{local}
1	1	0	1	0	CCW X_1^{local}	1	0	1	0	CCW X_1^{local}

Table D.2 Expected behaviour for the `3D-5SP-V2` controller using six shared power lines, for all possible global sensing states.

b_1	b_2	b_3	b_4	b_5	b_6	Behaviour
0	0	0	0	0	1	CCW X_2^{local}
0	0	0	0	1	0	CW X_2^{local}
0	0	0	1	0	0	CW X_3^{local}
0	0	0	1	0	1	CW X_3^{local} CCW X_2^{local}
0	0	0	1	1	0	CW X_3^{local} CW X_2^{local}
0	0	1	0	0	0	CCW X_3^{local}
0	0	1	0	0	1	CCW X_3^{local} CCW X_2^{local}
0	0	1	0	1	0	CCW X_3^{local} CW X_2^{local}
0	1	0	0	0	0	CCW X_3^{local}
0	1	0	0	0	1	CCW X_2^{local}
0	1	0	0	1	0	CW X_2^{local}
0	1	0	1	0	0	CW X_3^{local}
0	1	0	1	0	1	CW X_3^{local} CCW X_2^{local}
0	1	0	1	1	0	CW X_3^{local} CW X_2^{local}
0	1	1	0	0	0	CCW X_3^{local}
0	1	1	0	0	1	CCW X_3^{local} CCW X_2^{local}
0	1	1	0	1	0	CCW X_3^{local} CW X_2^{local}
1	0	0	0	0	0	T
1	0	0	0	0	1	CCW X_2^{local}
1	0	0	0	1	0	CW X_2^{local}
1	0	0	1	0	0	CW X_3^{local}
1	0	0	1	0	1	CW X_3^{local} CCW X_2^{local}
1	0	0	1	1	0	CW X_3^{local} CW X_2^{local}
1	0	1	0	0	0	CCW X_3^{local}
1	0	1	0	0	1	CCW X_3^{local} CCW X_2^{local}
1	0	1	0	1	0	CCW X_3^{local} CW X_2^{local}

Table D.3 Expected behaviour for the `3D-5SP-V1` controller using six shared power lines, for all possible global sensing states.

b_1	b_2	b_3	b_4	b_5	b_6	Behaviour
0	0	0	0	0	1	CCW X_2^{local}
0	0	0	0	1	0	CW X_2^{local}
0	0	0	1	0	0	CW X_3^{local}
0	0	0	1	0	1	CW X_1^{local}
0	0	0	1	1	0	CW X_1^{local}
0	0	1	0	0	0	CCW X_3^{local}
0	0	1	0	0	1	CCW X_1^{local}
0	0	1	0	1	0	CCW X_1^{local}
0	1	0	0	0	0	CCW X_3^{local}
0	1	0	0	0	1	CCW X_2^{local}
0	1	0	0	1	0	CW X_2^{local}
0	1	0	1	0	0	CW X_3^{local}
0	1	0	1	0	1	CW X_1^{local}
0	1	0	1	1	0	CW X_1^{local}
0	1	1	0	0	0	CCW X_3^{local}
0	1	1	0	0	1	CCW X_1^{local}
0	1	1	0	1	0	CCW X_1^{local}
1	0	0	0	0	0	T
1	0	0	0	0	1	CCW X_2^{local}
1	0	0	0	1	0	CW X_2^{local}
1	0	0	1	0	0	CW X_3^{local}
1	0	0	1	0	1	CW X_1^{local}
1	0	0	1	1	0	CW X_1^{local}
1	0	1	0	0	0	CCW X_3^{local}
1	0	1	0	0	1	CCW X_1^{local}
1	0	1	0	1	0	CCW X_1^{local}

Appendix E

Additional simulation results

In Section 5.2.3.3 the performance of the `3D-OSP-V1` and `3D-OSP-V2` controllers is compared, over increasing distance. However, that comparison only takes into account the success rate and completion time for each distance. In here, the comparison is extended to energy consumption and failure type.

Figure E.1 shows the average energy consumed in a successful trial, when using the `3D-OSP-V1` and `3D-OSP-V2` controllers, for initial distances between the robot and the goal of 1 to 20 robot body lengths. For initial distances up to 14 robot body lengths, the `3D-OSP-V2` controller consumes more energy than the `3D-OSP-V1` controller. The higher energy consumption is caused by the `3D-OSP-V2` controller having more active pumps in the initial of the trial, to avoid active translation towards the target. However, since the edge modules do not contribute to the translation, as the initial distance increases, the energy consumption increases are smaller. On the other hand, energy consumed by the `3D-OSP-V1` controller increases close to linearly with the distance. As a result, for longer distances the `3D-OSP-V1` controller consumes more energy than the `3D-OSP-V2` controller.

In Section 5.2.1, a trial was defined as successful is the robot reached the goal with the preferred face within the time limit. In here, we define a trial as partial successful if the goal is reached with a face other than the preferred, and as unsuccessful if there is no collision between the robot and the goal. Figure E.2 shows a comparison of the number of successful and partially successful trials for the `3D-OSP-V1` and `3D-OSP-V2` controllers, for multiple initial distances between the robot and the goal.

When using the `3D-OSP-V1` controller, the robot always reaches the target, due to constant active translation, regardless of the orientation it reaches it in. This leads to a high number

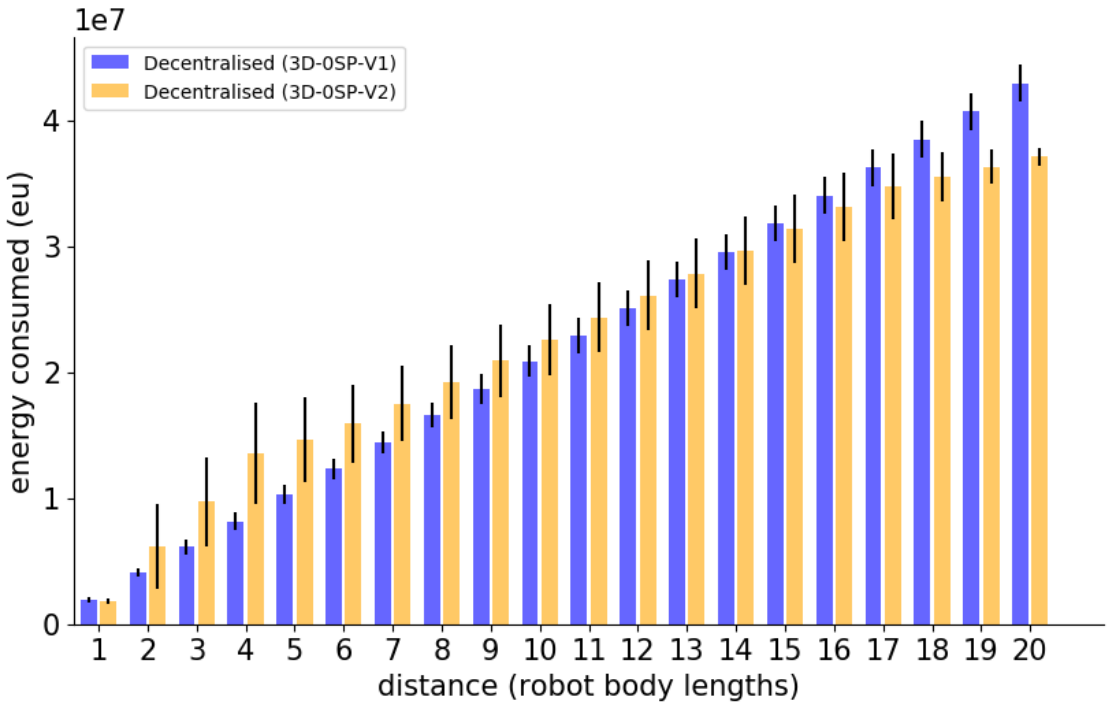


Fig. E.1 Average energy consumed per trial using the 3D-OSP-V1 and 3D-OSP-V2 controllers, for multiple initial distances between the robot and the goal. One energy unit is defined as the amount of energy consumed by a pump in one simulated time-step (time step $\Delta t = 0.01\,\text{s}$). The black lines represent the standard deviation for the 100 trials, for each distance. Results for successful trials only.

of partial successful trials for short distances. On the other hand, the 3D-OSP-V2 controller tends to only allow the robot to reach the goal with the correct orientation. As described in Section 5.2.3.3, if the robot does not reach the desired orientation in time, no collision with the goal is expected. For short distances, the obit radius can be small leading to unexpected collisions (partial successful trials).

In summary, two different failure behaviours can be observed. When using the 3D-OSP-V1 controller there are only successful and partially successful trials. When using the 3D-OSP-V2 controller, only successful and unsuccessful trials are expected.

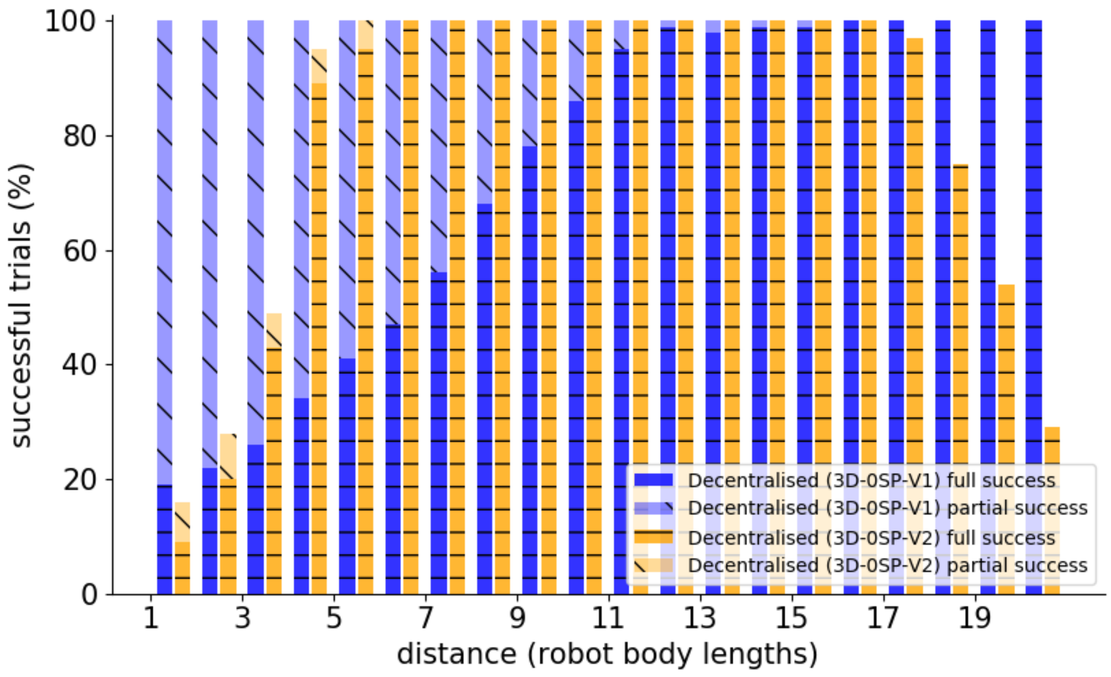


Fig. E.2 Full and partial success of each controller,for multiple initial distances between the robot and the goal.

Appendix F

2D-0SP - Experimental analysis

In Section 5.3.2 it is mentioned that the water tank used for the physical experiments is not adequate to perform experiments using the `2D-0SP-V1` controller.

In order to perform a proper estimation of the angle variations and trajectory, information on angular and linear accelerations, terminal velocities and de-accelerations is required. This data was obtained from recordings of a robot only performing rotation and translation (starting from rest, and upon reaching terminal velocity, de-accelerating). The acceleration and de-acceleration were approximated with a polynomial function of order 7.

For each initial orientation considered, the position and orientation of the robot were calculated, over time, taking into consideration the real sensing angle of the modules. The results, for orientation and trajectory are presented in Figures F.1 to F.4. In Figures F.2 and F.4 we can see that the center of the module reaches a Y position of close to 25 cm,

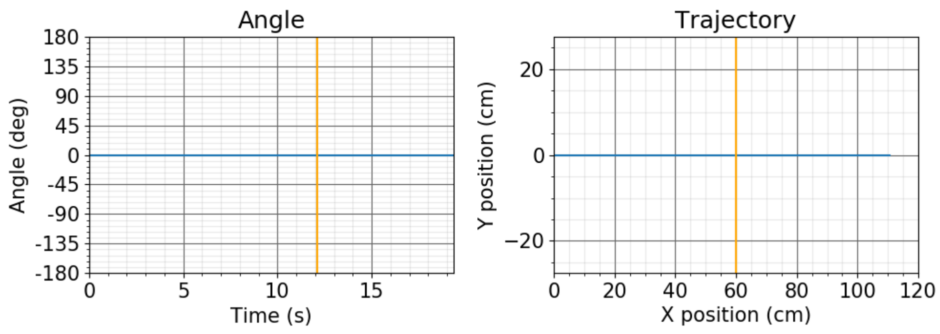


Fig. F.1 Theoretical angle variations and trajectory of a robot completing the task using the `2D-0SP-V1` controller. The initial orientation of the robot is 0°.

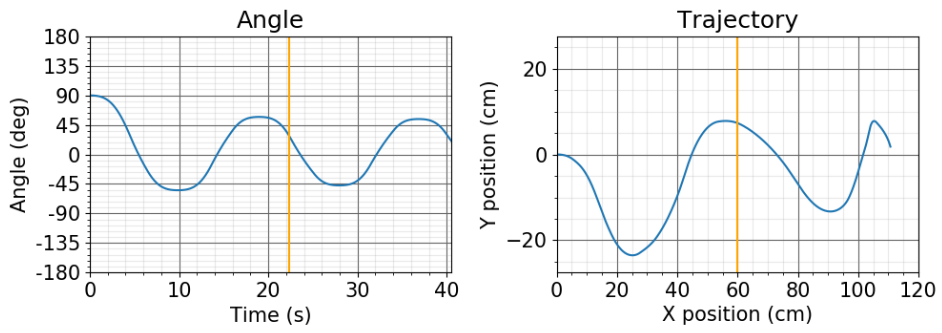


Fig. F.2 Theoretical angle variations and trajectory of a robot completing the task using the 2D-0SP-V1 controller. The initial orientation of the robot is 90 °.

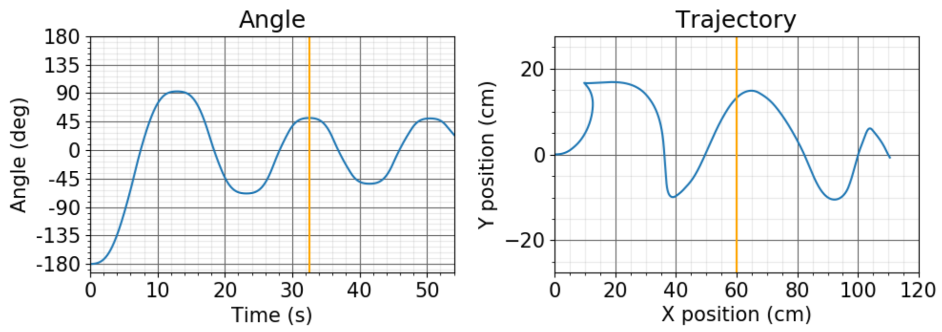


Fig. F.3 Theoretical angle variations and trajectory of a robot completing the task using the 2D-0SP-V1 controller. The initial orientation of the robot is -180 °.

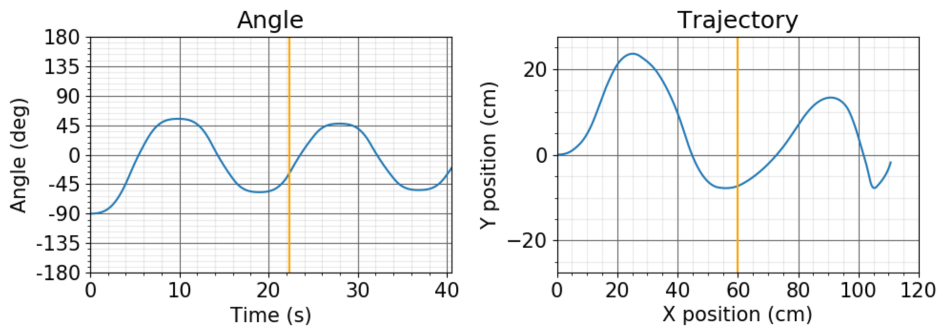


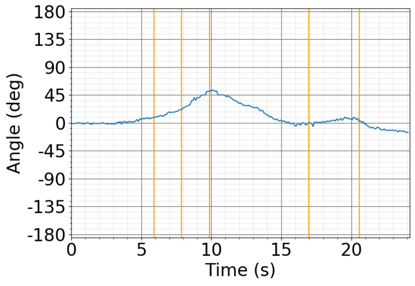
Fig. F.4 Theoretical angle variations and trajectory of a robot completing the task using the 2D-0SP-V1 controller. The initial orientation of the robot is -90 °.

meaning the, in the best case scenario, the edge of the module reaches over 30 cm, which is already over the limit of the water tank used.. Additionally, the data for the angular velocity was retrieved using a rotation based on the rotation policy of the `2D-1SP` controller, which generates a higher torque than it is possible for the `2D-0SP-V1` controller. This means that the rotation would be slower, leading to a higher maximum Y position. as a result, the water tank is not large enough, along the Y axis, to allow this controller to perform unimpeded.

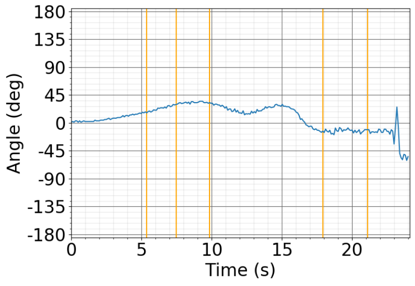
Appendix G

Full Experimental results

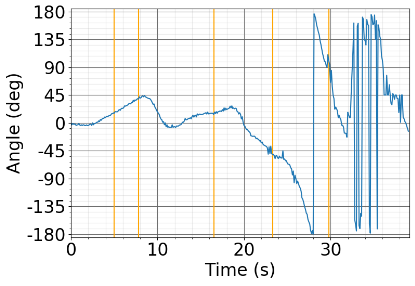
In Section 5.3.3 we presented the results, in terms of angle variation over time, for an example trial of each controller, using each initial orientation. In here, we present the results for the remaining trials of each controller, grouping them by initial orientation.



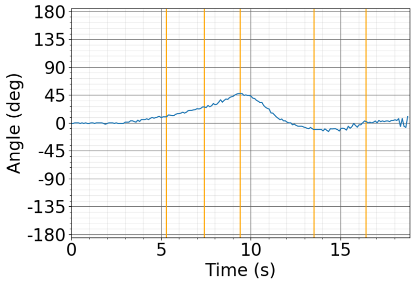
(a)



(b)



(c)



(d)

Fig. G.1 Angle changes over trial duration for the remaining 2D-1SP controller trials, using an initial orientation of 0°. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the x axis.

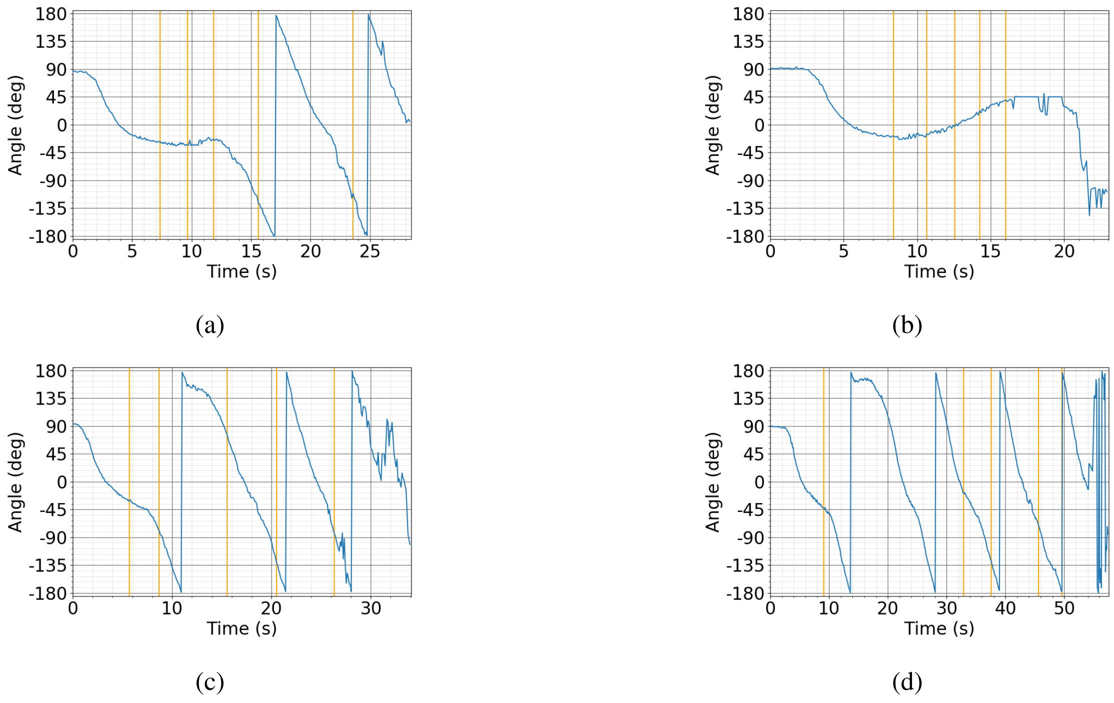
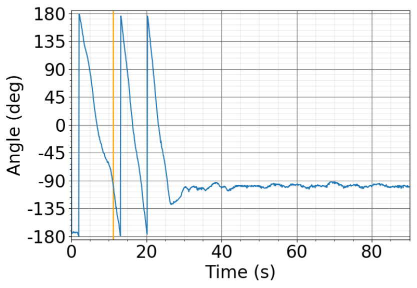
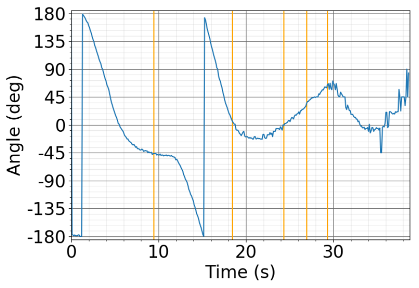


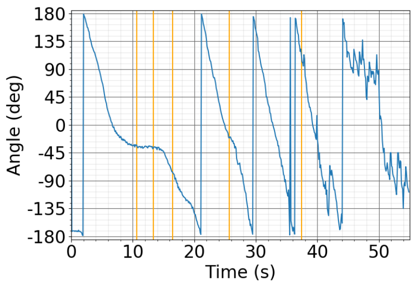
Fig. G.2 Angle changes over trial duration for the remaining 2D-1SP controller trials, using an initial orientation of 90°. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.



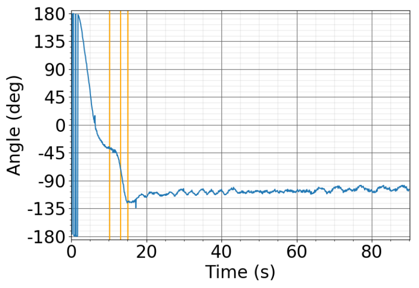
(a)



(b)

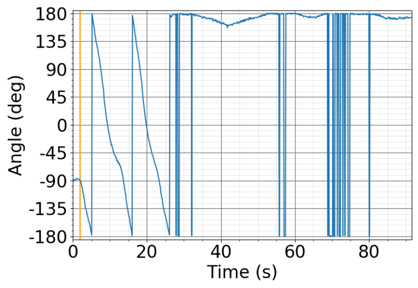


(c)

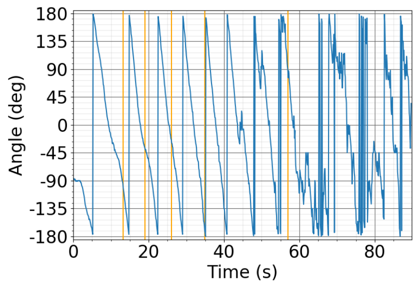


(d)

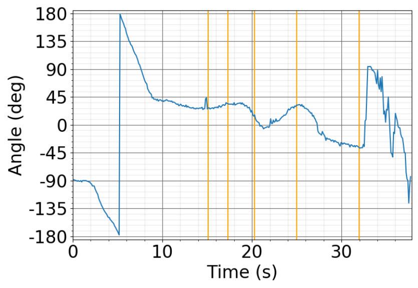
Fig. G.3 Angle changes over trial duration for the remaining 2D-1SP controller trials, using an initial orientation of 180°. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.



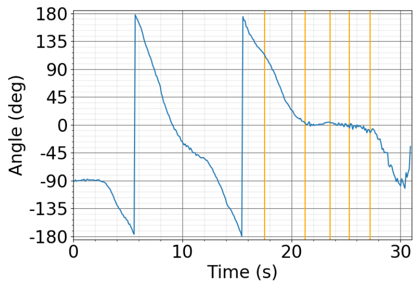
(a)



(b)

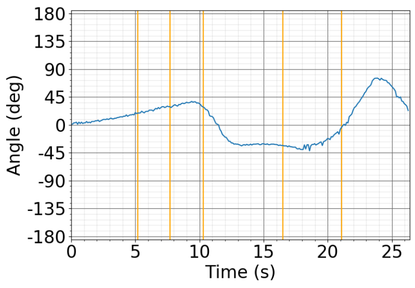


(c)

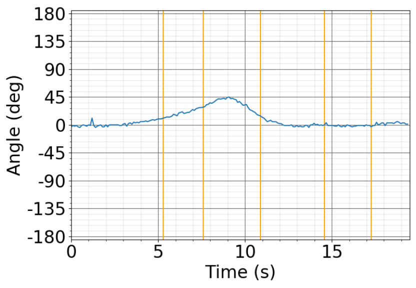


(d)

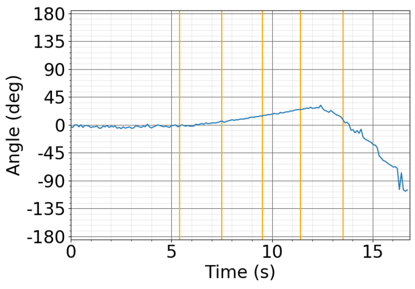
Fig. G.4 Angle changes over trial duration for the remaining 2D-1SP controller trials, using an initial orientation of -90 °. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.



(a)

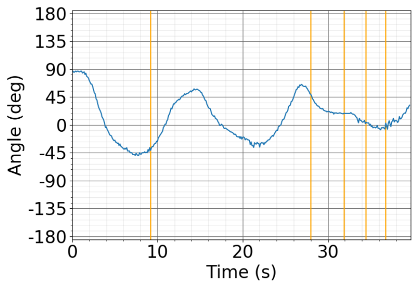


(b)

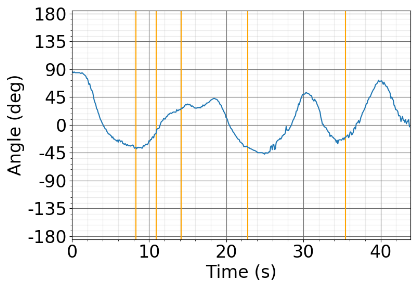


(c)

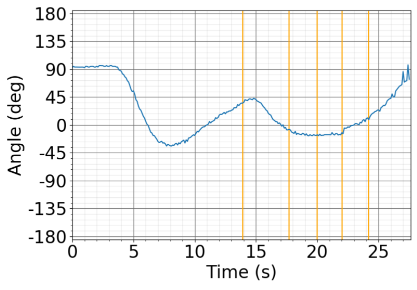
Fig. G.5 Angle changes over trial duration for the remaining 2D-2SP controller trials, using an initial orientation of 0 °. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.



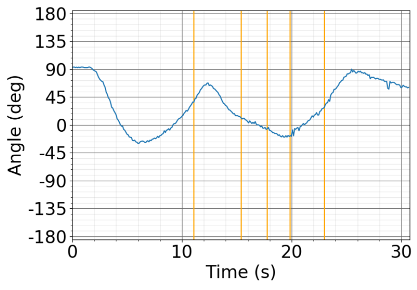
(a)



(b)

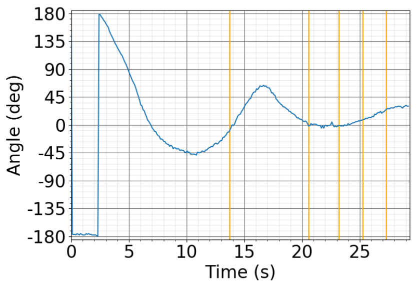


(c)

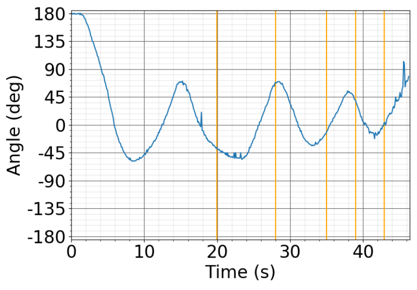


(d)

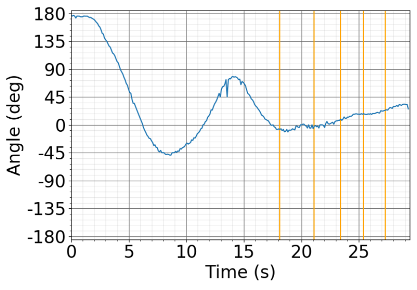
Fig. G.6 Angle changes over trial duration for the remaining 2D-2SP controller trials, using an initial orientation of 90°. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the x axis.



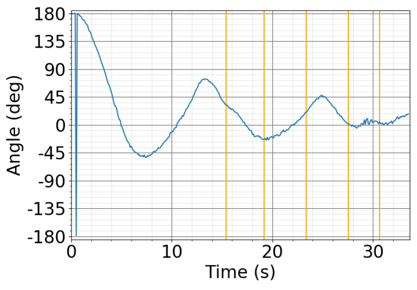
(a)



(b)

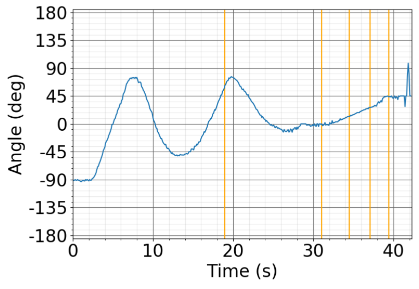


(c)

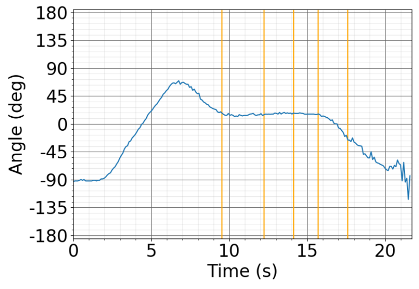


(d)

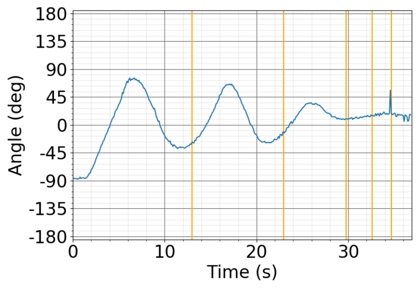
Fig. G.7 Angle changes over trial duration for the remaining 2D-2SP controller trials, using an initial orientation of 180 °. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.



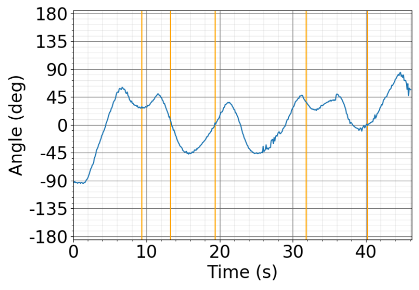
(a)



(b)



(c)



(d)

Fig. G.8 Angle changes over trial duration for the remaining 2D-2SP controller trials, using an initial orientation of -90 °. The orange vertical lines represent the times at which the robot reached 10, 20, 30 ,40, 50 and 60 cm in translation, along the *x* axis.