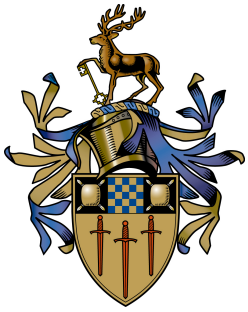
Using Reinforcement Learning to Design and Control Free-Flying Space Robots

Lucy Jackson

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey



Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

October 2022

Abstract

Free-flying space robots have the potential to revolutionise space exploration by facilitating a range of on-orbit operations. Whilst there have been some successful demonstrations of space robot technologies, the systems remain large with mission concepts limited to rendezvous with co-operative targets. However, the recent surge in small satellite technologies is changing the economics of space and downsizing a space robot is now viewed as a technologically feasible aim. Even with these advances, the design and control of a free-flying space robot is very challenging. Without a fixed base, the robot experiences large dynamic coupling effects where every motion causes counter-motions in other parts of the robot. These are influenced most heavily by the space robot's size and mass distribution as well as the employed control technique. As such, successful operation is heavily tied to both physical design and control. In fact, this coupling underpins all robotic systems since control and hardware design are intrinsically linked. This thesis therefore investigates the relationship between hardware and control, looking at how they impact each other and how this relationship can be exploited to improve performance. Although this work uses the On-Orbit Assembly (OOA) of a large aperture space telescope using free-flying space robots as a mission concept, the methods developed have applicability across the entire field of robotics.

There exists no documented design technique for any space robot. In fact, the dimensions and final design of successful missions are hard to find and never substantiated. This makes it hard to understand how robotic design impacts performance. This thesis presents a transparent design approach specific to free-flying space robots. A combined engineering approach is proposed, considering the mechanics and dynamics of the integrated system in order to present a design best suited for the OOA of a large aperture telescope. The feasibility of this space robot is evaluated in simulation and in-depth results are presented. This analysis supports the hypothesis that a small space robot is a viable solution for OOA and should be used as a starting point for the design of future systems.

As with the majority of robot design techniques, the initial hardware centered design approach in this thesis treats the control as immutable. Instead, simultaneous optimisation of hardware and control parameters is likely to improve overall performance. This is demonstrated by reasoning over both control and design in a Reinforcement Learning (RL) pipeline. The proposed method forms a complex differential path through a trajectory rollout, allowing a vast amount of information that was previously lost in the 'black-box' environment to be used. This means refinements can be made to both the morphology and control parameters simultaneously. The result is an efficient and versatile approach to holistic robot design. Performance improvements are seen with the space robot and a number of benchmark tasks compared to just performing hardware design optimisation.

While it is possible to modify robotic design in conjunction with control to improve performance, in some instances the design maybe changed with no option to modify the control algorithm. Data scarcity, brittle convergence and the gap between simulation & real world environments mean that most common RL approaches are subject to overfitting and fail to generalise to unseen environments. This means the replacement of parts with nonidentical components or the failure of sensors or joints will most likely render the control algorithm useless. Hardware-agnostic policies would mitigate this by allowing a single network to operate in a variety of test domains, where dynamics vary due to changes in robotic morphologies. This thesis utilises the idea

that learning to adapt a known and successful control policy is easier and more flexible than jointly learning numerous policies for different morphologies. It presents the idea of Hardware Agnostic RL. In this approach, two control polices are combined and varied embodiments are sampled using a novel adversarial loss function. This self-regulates morphologies based on their performance. The result is a final control policy that is robust to changes in the environment as well as degradation and failure of the robot.

Key words: Reinforcement Learning, Free-Flying Space Robot, On-Orbit Assembly, Robotic Design

Email: lucy.jackson.309@gmail.com

WWW: https://www.linkedin.com/in/lucy-jackson-9208bb189/

Acknowledgements

I would like to thank all of the supervisors I had during this PhD. Firstly, Dr. Simon Hadfield who took this project on under unusual circumstances and in doing so introduced me to the field of RL. Thank you for your continued support, from teaching me the fundamentals of machine learning when I was completely new to the field, to pushing me to achieve more than I ever imagined. Not only would this thesis not be what it is without your help and guidance, but truthfully it probably wouldn't exist. Second, Dr. Celyn Walters whose patience in helping me fix whatever machine or piece of software I managed to break was invaluable — as was teaching me how to use apostrophe's and commas. Finally, Prof. Mini Rai, thank you not only for your guidance and support during my first year as a researcher, but also for sparking my interest in the field of space robotics.

I would also like to thank everyone at CVSSP. Thank you for welcoming me with open arms, even after a 1 year delay. Thank you for providing endless laughs, debates and sometimes lunch breaks, be it in person or virtually.

This PhD was co-funded by Surrey Satellite Technology Ltd. I would like to thank Steve Eckersley for his help throughout and everyone on the Missions Concept team for making my placement so enjoyable.

Lastly I would like to thank my family. You got the pleasure of seeing me at my most stressed points — convinced I wouldn't finish or that something was not good enough. Thank you for listening to me rant (sometimes politely), usually ignoring it, and then telling me that it would be fine however many times I told you it wouldn't.

Contents

Nomenclature

AAST	Autonomously Assembled Space Telescope
ADR	Active Debris Removal
AI	Artificial Intelligence
AOCS	Attitude and Orbit Control Subsystem
API	Application Programming Interface
BP	Back-plane
CDF	Cumulative Distribution Functions
CMA-ES	Covariance Matrix Adaptation with Evolutionary Selection
CIRAS	Commercial Infrastructure for Robotic Assembly and Services
c.o.m	Center of Mass
COTS	Commerical Off-the-Shelf
CSA	Canadian Space Agency
DARPA	Defense Advanced Research Projects Agency
DEOS	Deutsche Orbitale Servicing Mission
DLR	German Aerospace Center
d.o.f.	Degrees of Freedom
DDPG	Deep Deterministic Policy Gradient
DPG	Deterministic Policy Gradient
DQN	Deep Q-Network
EPOpt	Ensemble Policy Optimization
ESA	European Space Agency

ETS-VII	Experimental Test Satellite VII
FoV	Field of View
FREND	Front-end Robotics Enabling Near-term Demonstration
GEO	Geosynchronous Orbit
HARL	Hardware Agnostic Reinforcement Learning
HARL-A	Hardware Agnostic Reinforcement Learning through Adversarial Selection
HCP	Hardware Conditioned Policies
HER	Hindsight Experience Replay
HST	Hubble Space Telescope
ISS	International Space Station
iSSI	Intelligent Space System Interface
JWST	James Webb Space Telescope
KL	Kullback–Leibler
LEO	Low Earth Orbit
MAML	Model-Agnostic Meta-Learning
MDP	Markov Decision Process
MOO	Multi-Objective Optimisation
MOST	Microvariability and Oscillations of Stars
MSE	Mean Squared Error
NASA	National Aeronautics and Space Administration
NFN	Normalising Flow Network
NN	Neural Network
ODE	Ordinary Differential Equation
OOA	On-Orbit Assembly
OOO	On-Orbit Operations
ORCHID	Optimisation of Robotic Control and Hardware in Design
PID	Proportional Integral Derivative
PM	Primary Mirror
PPO	Proximal Policy Optimisation

R&D	Research and Development
RAMST	Robotically Assembled Modular Space Telescope
RARL	Robust Adversarial Reinforcement Learning
RL	Reinforcement Learning
RNVP	Real-Valued Non-Volume Preserving
R.P.M	Revolutions per minute
RS	Random Selection
TALISMAN	Tendon-Actuated Lightweight In-Space Manipulator
Ta-DAH Design	Task-Driven Automated Hardware Design
TD	Temporal Difference
TRPO	Trust Region Policy Optimisation
SM	Secondary Mirror
SPIDER	Space Infrastructure Dexterous Robot
SRMS	Shuttle Remote Manipulator System
SSRMS	Space Station Remote Manipulator System
TRL	Technology Readiness Level
zero-g	Zero Gravity

Symbols

A number of subscript symbols are defined that are used throughout this thesis in combination with the other symbols. Two derivative symbols are also used in this work, both are defined here.

Subscript Symbols

m	Referring to the robotic manipulator
sc	Referring to the base spacecraft
scm	Referring to the coupling between the base and the mainpulator
d	Desired location
t	Referring to a time step
$*$	The optimal version of that parameter (used as a superscript)

Derivative Definitions

$\dot{}$	First derivative
$\ddot{}$	Second derivative

Introduced in Chapter 3

U	CubeSat unit
d_x	Width of base spacecraft
d_y	Depth of base spacecraft
d_z	Height of base spacecraft

m	Mass of a rigid body
N	d.o.f. of a system
$\sum B$	Reference frame fixed to the c.o.m of the base spacecraft
$\sum I$	Earth centered inertial reference frame
$\sum l$	Reference at frame the base of link
$\boldsymbol{\tau}$	Forces applied to space robot system
f_x	Linear force applied to the base spacecraft in the x dimension
f_y	Linear force applied to the base spacecraft in the y dimension
f_z	Linear force applied to the base spacecraft in the z dimension
τ_α	Torque applied to the base spacecraft about the x dimension
τ_β	Torque applied to the base spacecraft about the y dimension
τ_γ	Torque applied to the base spacecraft about the z dimension
τ_θ	Torque applied to link joint
D	Inertial properties of the system
C	Centrifugal and Coriolis properties of the system
q	State of system
x	Linear position of base w.r.t. x-axis of $\sum I$
y	Linear position of base w.r.t. y-axis of $\sum I$
z	Linear position of base w.r.t. z-axis of $\sum I$
α	Rotation of base about x-axis w.r.t. $\sum I$
β	Rotation of base about y-axis w.r.t. $\sum I$
γ	Rotation of base about z-axis w.r.t. $\sum I$
θ	Rotation of link w.r.t. $\sum l$

Introduced in Chapter 4

$\mathbb{C}$	Cost function
w	Weights
$\mathbb{F}$	Objective function

J	Total number of objective functions
$\mathbf{v}$	Design vector
$\mathcal{V}$	Design vector space
l	Scalar link length
t	Time step
$\mathbf{J}$	Jacobian matrix
$\mathbb{I}$	Inverse kinematic function
T	Total time for manipulator to execute desired task
$\overline{\mathbb{F}}$	Normalised objective function
$\mathbf{A}$	Wheel position within spacecraft
$\mathbf{h}^R$	Torque provided by reaction wheels in spacecraft
m_f	Mass of fuel
I_{sp}	Specific impulse of fuel

Introduced in Chapter 5

$\mathbf{a}$	Action taken by a policy or agent
r	Reward for taking an action from a given state
$\mathcal{P}$	Transition function
$\mathcal{M}$	Markov decision process
$\mathbf{s}$	State of an environment
$\mathbb{P}$	Probability function
$\mathcal{S}$	State space
$\mathcal{A}$	Action space
γ	Discount factor
$\tilde{\theta}$	Parameters of policy network
π	General neural network
R	Discounted return over an episode
Q	Q-function

$\mathbb{E}$	Expectation
V	Value function
$\tilde{\omega}$	Parameters of critic network
k_I	Integral constant for PID controller
k_P	Proportional constant for PID controller
k_D	Derivative constant for PID controller
$\mathbf{e}$	Error
μ	Mean
σ	Standard deviation
$\mathcal{R}$	Reward function
$\mathcal{J}_a$	Loss for actor network
ϵ	Clip parameter for PPO algorithm
A	Advantage function
$\mathcal{J}_c$	Loss for critic network
$\mathcal{N}$	Gaussian distribution
$\mathcal{J}_{PID}$	Loss for PID controller
$\mathcal{J}_{\mathbf{v}}$	Design parameter loss function
$\mathcal{J}_{\mathbf{v}d}$	Direct loss design parameter
$\mathcal{L}_a$	Learning rate for actor network
$\mathcal{L}_c$	Learning rate for critic network
$\mathcal{L}_v$	Learning rate for design vector
$\mathcal{L}_{PID}$	Learning rate for PID control parameters

Introduced in Chapter 6

$\tilde{\epsilon}$	Parameters of expert network
$\tilde{\psi}$	Parameters of modification network
$\tilde{\delta}$	Parameters of the adversary network
$^-$	Referring to the modification network

$\hat{}$	Referring to the expert network
$\circ$	Referring to the adversary network
l_p	Learning potential
$\mathcal{L}_{mod}$	Learning rate for the modification network
$\mathcal{L}_{NFN}$	Learning rate for the NFN network
$\mathbf{p}$	Target vector of numbers
$\mathbf{q}$	Original vector of numbers
f	Flow function
J	Jacobian
K	Total number of flows
k	Flow in question
D_{KL}	KL divergence
$\mathcal{J}_{adv}$	Loss for adversary network
Φ	Any nonlinear activation function
S	Scale function
T	Translation function
E_c	Episode count
$\mathring{N}$	Number of samples to train adversary network
$\mathbf{w}$	Weight applied to layer of NFN

List of Figures

List of Tables

Declaration

This thesis and the work to which it refers are the results of my own efforts. Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in the text, bibliography or in footnotes. This thesis has not been submitted in whole or in part for any other academic degree or professional qualification. I agree that the University has the right to submit my work to the plagiarism detection service TurnitinUK for originality checks. Whether or not drafts have been so-assessed, the University reserves the right to require an electronic version of the final document (as submitted) for assessment as above.

The work presented in this thesis is also present in the following manuscripts:

- Lucy Jackson, Chakravarthini M. Saaj, Asma Seddaoui, Calem Whiting, Steve Eckersley, and Simon Hadfield. Downsizing an orbital space robot: A dynamic system based evaluation. *Advances in Space Research*, 65(10):2247–2262, May 2020.
- Lucy Jackson, Celyn Walters, Steve Eckersley, Pete Senior, and Simon Hadfield. ORCHID: Optimisation of robotic control and hardware in design using reinforcement learning. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4911–4917, 27^{th} September - 1^{st} October, 2021. Virtual.
- Lucy Jackson, Celyn Walters, Steve Eckersley, and Simon Hadfield. HARL-A: Hardware agnostic reinforcement learning through adversarial selection. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3499–3505, 27^{th} September - 1^{st} October, 2021. Virtual.
- Lucy Jackson, Celyn Walters, Chakravarthini M. Rai, Steve Eckersley, and Simon Hadfield. Ta-DAH: Task driven automated hardware design of free-flying space robots. In *Proceedings of the 16^{th} International Conference on Space Robotics and Automation (ICSRA)*, 19^{th} - 20^{th} July, 2022. Virtual.

Signed:

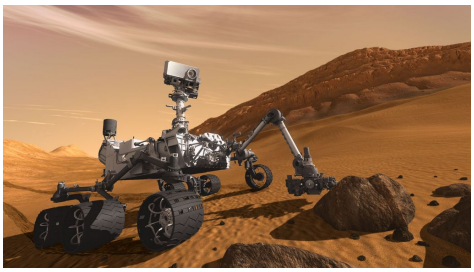
Date: 10.10.2022

Chapter 1

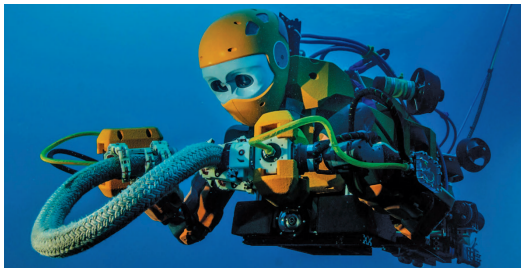
Introduction and Motivation

From the introduction of the first commercial robot in 1958 to the more recent development of surgical robots and personal assistant humanoids, many would argue that day-to-day life would not be the same without robotic technologies. Initially used to improve efficiency in farming, manufacturing and production via the automation of mundane and sometimes dangerous tasks, more industries are starting to invest in robotics. While very prominent in warehouses and medical settings, robots are less prevalent in situations that require more complex or sophisticated behaviors such as on the roads or interacting directly with humans. Success in these more dynamic environments involves the robot perceiving & making sense of its surroundings, planning a suitable motion to complete a task, executing the motion via the control of actuators and then monitoring its success to feedback into its plan. All of these task components are highly complex problems to solve individually, let alone end-to-end in a single system. While advances in computer vision have enabled intelligent perception, the emergence of Reinforcement Learning (RL) provides a framework to solve the problem of planning and executing actions. As such, it is becoming possible for robots to solve tasks of higher complexity.

Deep RL is an emerging Artificial Intelligence (AI) technique used to allow robots to learn from their environment through trial-and-error interactions. In recent years the field has seen huge improvements, notably, beating world-class players at strategic board games, playing video games to superhuman standards and controlling continuous robotic systems [149, 83, 86]. In the area of robotic control, policies learned via RL have started to replace more traditional techniques such as Proportional Integral Derivative (PID) and H∞ controllers, both of which



(a) NASA's Curiosity Rover on Mars. Image courtesy of NASA.



(b) Underwater avatar [68].

Figure 1.1: **Comparison of different robotic agents.** Robots are usually designed for specific tasks and as a result the hardware varies drastically based on the task at hand.

rely heavily on path planning algorithms. Unlike these traditional approaches, the use of RL algorithms removes the need for an off-line planning phase and implementation requires lower levels of domain knowledge.

In addition to control, the successful implementation of robotic systems relies on the physical structure and hardware of the robot itself. Given the large range of possible applications and operating conditions, there exists a large variety of bespoke robotic architectures. There are custom-built robots designed specifically for space & deep sea exploration, search & rescue missions and military operations. As an example, National Aeronautics and Space Administration's (NASA) Curiosity rover, shown in Figure 1.1a, was designed to explore Mars. It is modeled around a buggy design with six wheels mounted on two rocker-bogie suspension systems [160]. It hosts a number of perception sensors along with a rock vaporizing laser and different drills, allowing it to perform exploration tasks in-situ. Conversely, an underwater robotic avatar designed to explore marine life at depths of over $1\,\mathrm{km}$, takes a very different form. The avatar can be seen in Figure 1.1b. It has only thrusters as a form of propulsion and is modeled around the human body. It hosts two 7 Degrees of Freedom (d.o.f.) arms with hands capable of tactile sensing, allowing it to pick up items from the seabed. Even from this brief overview, it quickly becomes apparent that neither agent would have any level of success if put in the other's setting, even though the aim of both is to explore an extreme environment.

Original approaches to robotic design involved trial-and-error prototyping evaluated by quantifying the robot's performance in the target task. This results in individual solutions following a costly, time and resource intensive process. In an attempt to combat this, designs for different

agents with a similar or identical function can be modified and updated as opposed to restarting the Research and Development (R&D) process. This is what happened with the Mars rovers designed by NASA. To date, the agency has successfully operated 5 rovers on the planet. While the overall design of the rovers stayed consistent, modifications and improvements were made to different components *e.g.* the wheel diameter increased incrementally from the first to the last design. A similar design approach can be seen with different groups of medical robots. Almost all of those used for laparoscopic surgery take near identical forms but with different end effectors and d.o.f. This 'heritage' approach to robotic design can speed up the R&D process and lower costs by allowing for higher levels of technology transfer and 'learning from mistakes'. Yet, robotic design and R&D remains expensive and time-consuming. Even with prior knowledge, the development of the most recent rover took 11 years and cost upwards of $\$2.4$ million [76]. This thesis will therefore explore techniques to accelerate and streamline hardware design through automation. Automating the design process will aid in cutting the costs and lead times of robotic design projects by removing the need to manually produce, evaluate and test different design options. The automated design process developed in this thesis should encode and leverage prior knowledge in a compact and meaningful way in order to facilitate information transfer. In addition to improving the R&D process, there is potential for the successful implementation of an automated design technique to improve the overall performance by finding a design that would not have been discovered via systematic human iteration.

Just as important as the correct physical design for a robot's successful operation is the employment of the correct control technique. A well-designed robot is useless if it is not capable of actuating itself, or being actuated in a manner that will facilitate the successful accomplishment of tasks. It quickly becomes apparent that successful operation of any robotic system is heavily tied to both its physical design and control. Unfortunately, in practice, it is impossible to decouple what constitutes a good hardware design from the control policy used. This is because of the intrinsically coupled physical laws that underpin motion. A biological example of the interconnectedness of mechanical design and motor skills can be seen in human athletes. Sprinters aim to develop stronger hip muscles and a technique that involves higher frequency, shorter strides, while long distance runners require stronger ankle muscles and longer strides at a lower frequency [45]. Most approaches to hardware design optimisation generally adopt a simplistic control scheme since it is inefficient and time-consuming to fine-tune a controller or

train an RL agent for each design iteration. This means that once the hardware design is finalised and a more specialised control policy is developed, the previously chosen hardware may no longer be optimal. As a result the design process could be improved if a combined approach was taken whereby the physical design and control of the agent are reasoned over simultaneously. This thesis explores ways of co-optimising the control and design of a robotic agent. Such a process would remove the need to iterate between physical design and control, improving the final performance while also providing a more time efficient and less costly design process.

Although a co-optimised design will likely function better than one designed in a disjointed manner, the result is a control scheme suitable for use with only that robot. During training an RL agent implicitly learns the kino-dynamics of the exact system on which it was trained. Since the kinematics and dynamics of a system are functions of the physical hardware, it means that any changes to the robot's form will induce a change in this model. As such, hardware changes are likely to render a trained RL agent or other rudimentary control technique ineffective. Taxing data requirements, brittle convergence and hyperparameter sensitivity in the RL domain makes the training of agents a volatile and long process. Training a new agent for each iteration of robotic hardware therefore requires excessive additional time and compute. To combat this, the control of multiple agents with differing kino-dynamics using a single control policy will be investigated. Policies capable of operating on a range of different robots would increase the longevity of systems due to robustness to dynamic changes as a result of robot degradation or failure.

To summarise, it is the aim of this thesis to explore a number of avenues related to the interconnectedness of a robot's hardware and control. This includes approaches for automated design, simultaneous reasoning over hardware design & control and approaches to control that are robust to changes in hardware.

1.1 Motivating Use Case

The work presented in this thesis is inspired by the use of space robots to carry out the On-Orbit Assembly (OOA) of a large aperture space telescope. In this thesis a space robot is defined as a robotic manipulator attached to a base spacecraft, a representation of which can be seen in Figure 1.2. Note that the manipulator can have any number of d.o.f.

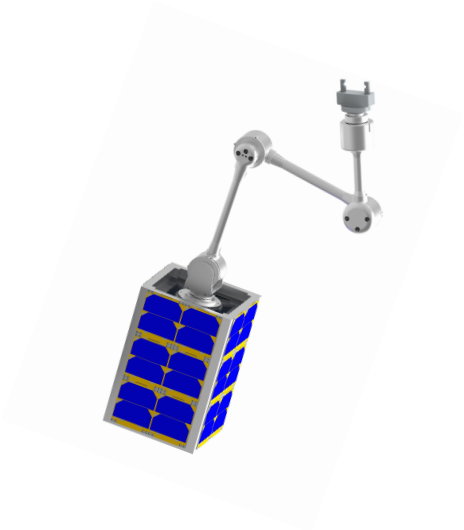


Figure 1.2: **Space Robot.** As defined in this thesis a space robot consists of a manipulator of any number of d.o.f. mounted on a base spacecraft of any size [62].

Until recently OOA has been limited to large national missions, such as the assembly of The International Space Station (ISS), with bespoke components and oversized robotics. However, a new space era is approaching in which low-cost autonomous robotic assembly systems are commercially viable and therefore applicable to smaller missions and a wider market. Current technologies in the field are reaching their physical limits and the advantages of designing and launching a larger space telescope are extensive. Not only will the imaging distance and the resolution of images improve, but it will demonstrate that it is possible to build large structures in space using OOA. Initially telescopes reached a maximum size due to an inability to manufacture monolithic mirrors above a certain diameter. In response to this, deployable segmented mirrors and other structures were developed — such as the James Webb Space Telescope (JWST). However, even in its stowed configuration the JWST exceeds the size of any vacuum chamber on Earth, meaning the usual pre-flight tests could not be carried out. Instead, its deployment in-situ had to be monitored extensively and modified as engineers saw, for the first time, how it reacts in the super-cold, Zero Gravity (zero-g) environment. This design already pushes the feasible size limit for a deployable telescope in space and new technologies must be developed.

The tasks required to assemble a structure in space are extensive and will likely require a multitude of highly dexterous space robots, known as a swarm. The high level of dexterity

required means that the robots must be small enough to achieve fine accurate motion and compliant enough to handle small and fragile components, yet also be large enough to manipulate heavy payloads. In addition to this, systems should be capable of manoeuvering relative to the telescope without collision and with high precision. In practice, it is therefore likely that a number of different sized spacecraft will be needed, where each spacecraft in the swarm is optimal for a different task.

Each space robot must be designed for its specific task while also being able to overcome the effect of dynamic coupling. This phenomenon is the result of the combination of the conservation of momentum and microgravity operating conditions. It means that any controlled or uncontrolled motion of the arm will induce a destabilising force on the base spacecraft and vice versa. This effect increases as the difference in inertia between the manipulator and the base spacecraft increases. If the manipulator is downsized too much it will lack dexterity and reach, resulting in a small workspace and diminished capability. However, a bigger arm will need a larger base in order to ensure system stability, ultimately leading to higher launch costs and lower mission flexibility. Pivotal to solving this problem is understanding the coupled, highly non-linear dynamics of the arm and the base spacecraft and how it can best be designed and controlled. This use case therefore provides strong motivation for the investigation of the interconnectedness of control and hardware design presented in this thesis.

1.2 Objectives

The overall aim of this thesis is to investigate the relationship between robotic hardware and control and how each impacts performance. The challenges and context discussed in the previous section provide a number of distinct objectives and aims for this thesis:

1. To explore the automation of hardware optimisation techniques as a way of improving the performance of a small space robot carrying out OOA.
2. To develop an automated design methodology that will reason over hardware and software simultaneously, rather than using a disconnected two stage approach.
3. To develop RL task-based control solutions for generalisation to different robotic hardware.

It is important to note that the work contained in this thesis has general applicability in the wider field of robotics, not only the target OOA application. While the first contribution addresses this use case directly, there is a running theme of evaluating the other developed techniques across multiple scenarios. It is hoped that this work has interest for robotic design and control within the space sector and beyond.

1.3 Contributions

In order to address the defined objectives a number of distinct and interrelated pieces of work are presented. Chapter 3 provides context to this work by defining the architecture of a modular space telescope that is suited to OOA. It also outlines a suitable mission concept and details on the space robot simulator used throughout this work. Specifics on the exact telescope design can be found in the appendix.

Chapter 4 presents Task-Driven Automated Hardware Design (Ta-DAH Design) [66] — an automated design technique for a space robot, thus addressing Objective 1. This approach uses Multi-Objective Optimisation (MOO) to determine the optimal sizing of a robotic manipulator, based on a dynamic system evaluation [63]. A systematic approach is then employed to size the base spacecraft. Throughout, designs are evaluated with a simplistic control scheme. This results in some designs having lower performance than expected. As mentioned previously, high performance can only be expected if the hardware and software are designed simultaneously.

Chapter 5 addresses this problem, and in doing so, Objective 2. It presents Optimisation of Robotic Control and Hardware in Design (ORCHID) [64] — a technique that simultaneously optimises a robot's hardware and control policy. This is achieved by utilising differentiable dynamic simulators to increase the amount of information that can be leveraged during training. However, as with other RL architectures, ORCHID overfits to the single optimised design. While this gives desirable behavior in the short-term, it means that any changes to robotic hardware render the control policy useless.

As a solution, chapter 6 presents Hardware Agnostic Reinforcement Learning through Adversarial Selection (HARL-A) [65], which addresses Objective 3. HARL-A is an RL architecture that allows for a more general control policy to be learned. It is specifically tailored to learn a

policy over changing robotic embodiment or a number of failure modes. This provides a level of redundancy and robustness to finalised robotic systems.

Chapter 7 provides a summary of the thesis findings and suggests a number of directions for future work in the field. In summary, the main aim of this thesis is to investigate the relationship between the physical design and control of a range of robots.

Chapter 2

Literature Review

This chapter introduces existing methods and topics related to the work in this thesis. Initially different space telescope missions and architectures are outlined, as well as past missions involving OOA and the robotic technologies used. Then both classic and more modern design approaches for robotic manipulators are discussed, including those that involve RL. The chapter ends by covering different approaches to achieve robust, hardware-agnostic control.

2.1 On-Orbit Assembly Using Robotic Spacecraft

2.1.1 Optical Space Telescopes

The first optical telescope in space was Hipparcos — a scientific satellite destined for Geosynchronous Orbit (GEO), launched by European Space Agency (ESA) in 1989 [120]. It was a reflective Schmidt telescope with a spherical primary mirror measuring $29\,\mathrm{cm}$ [172]. This mission was considered successful having obtained results about 1.5 to 2 times better than the original aims [79, 120].

One year after the launch of Hipparcos, the Hubble Space Telescope (HST) launched — a joint venture between NASA and ESA. The Primary Mirror (PM) in the HST had a diameter of $2.4\,\mathrm{m}$ making it much larger than the previous ESA mission, and one of the largest mirrors in space [99]. Yet the performance of the HST was lower than expected as it suffered from spherical aberration. The PM was equipped with 24 force actuators in two concentric circles, designed to flex the

mirror's surface to account for this aberration. However, the actuators were undersized compared to the mirror diameter and did not have sufficient range to correct the large and unexpected difference in focal length [24]. This reduction in performance highlighted one of the biggest challenges when designing and launching large mirrors for space telescopes. Following this, it wasn't until 2003 that another space telescope was launched. This was Microvariability and Oscillations of Stars (MOST), developed by The Canadian Space Agency (CSA) [8]. Classed as a microsatellite at less than $50\,\mathrm{kg}$, no new large mirror technologies were required, and instead this mission addresses challenges at the opposite end of the spectrum [26]. Following this a number of smaller systems became operational. However, as with MOST these all involved much smaller mirrors and are therefore not discussed as they hold little relevance to the context of this thesis.

The Kepler space telescope, which launched in 2009, is the next telescope of interest in the context of this work. It uses a reflective Cassegrain-Schmidt design, with a primary mirror of $1.4\,\mathrm{m}$ in diameter, making it the second-largest mirror of any optical space-based observatory after the HST [19]. Although launching almost 20 years apart, the Kepler Telescope's PM and the HST's PM are manufactured from the same material and are of very similar geometry [10]. Of interest is that no new phasing technologies were launched to account for the performance reduction seen by the HST. Instead, the size of the mirror was reduced. The optical space telescope design in this thesis utilises similar sized mirrors to those used on the Kepler Telescope.

The JWST was the first of a new generation of mirror concepts, being the first to employ a segmented PM [10]. The motivation for this approach was the need to launch a PM of $6\,\mathrm{m}$ in diameter in a launch vehicle fairing of $4.5\,\mathrm{m}$. Although not strictly an optical telescope since imaging will occur in the infrared wavelengths, the deployable design is of great interest. The PM is made of 18 hexagonal beryllium segments, and its Secondary Mirror (SM) sits at the end of a tripod-like structure [108]. Each segment is $1.32\,\mathrm{m}$ flat-to-flat, making them more similar in size to the mirrors used on the Kepler Telescope, and much smaller than those on the HST [52]. In order to ensure these segments act as a single mirror once deployed, each is equipped with a 7 d.o.f. actuator — 6 d.o.f. for rigid body motion and an additional d.o.f. for the radius of curvature [52]. Having launched in 2021 the JWST is the largest telescope currently in-orbit. This segmented mirror design hosts some issues, most of which are as a result of needing to accurately position the segments relative to each other. However, at present it is the most suitable design for the OOA of large mirrors [104, 106]. The PM is split across three different structures,

two of which are deployed once the JWST reaches its desired location. This requires a high number of automatic motor-driven motions for the telescope to be fully operational, with only built-in redundancy as failure back-up. As such, the mission concept presented in Section 3 proposes the use of free-flying space robots to assemble the mirror components. Not only does this limit the number of mechanisms required in the design, but also allows for replacement if any module or component were to fail.

2.1.2 Past and Planned Robotic Missions

The first robotic system used in space was the Shuttle Remote Manipulator System (SRMS), which was deployed from the cargo bay of the Space Shuttle in 1981 and aided in the assembly of the ISS [50]. It was a $15.2\,\mathrm{m}$ long, robotic manipulator, with 6 d.o.f. [129]. Since then more bespoke robotic manipulators have been developed. ESA have developed the European Robotic Arm which operates on the Russian segment of the ISS, and the CSA have upgraded the SRMS, now named the Space Station Remote Manipulator System (SSRMS) [20, 137].

Free-flying robotic manipulators were introduced as a more versatile solution to carry out tasks on-orbit. The first technology demonstration mission was in 1997, when the Experimental Test Satellite VII (ETS-VII) launched. The manipulator in this mission had 6 d.o.f. and a reach of $2\,\mathrm{m}$. It was mounted on a $2.5\,\mathrm{t}$ chaser satellite that was able to rendezvous with a cooperative target spacecraft [111]. This mission came to an end when time delays in the control system meant the reaction forces, as a result of the dynamic coupling, could not be counteracted quickly enough causing the system to become unstable [119]. Improvements in control theory and sensing capability meant this was not an issue for the Orbital Express mission. This was a joint effort between Boeing and Defense Advanced Research Projects Agency (DARPA). Launched in 2007, it was designed to validate the feasibility of rendezvous and autonomous refueling [100]. This mission proved much more successful than its precursor, although the hardware remained large, with the base spacecraft weighing $1\,\mathrm{t}$ [51].

Following this, technology demonstrations of free-flying space robots capable of operating in-orbit remain in the conceptual stage. German Aerospace Center (DLR) developed the Deutsche Orbitale Servicing Mission (DEOS), which aimed to demonstrate the feasibility of maintenance and servicing, but never progressed beyond the preliminary design definition stage [128]. This program was renamed as the Front-end Robotics Enabling Near-term Demonstration (FREND),

with the aim of servicing satellites not built to enable robotic interaction [39]. This technology has been fully tested at laboratory level but is yet to launch [50]. The manipulator developed in the FREND concept has since been suggested for use in a range of other servicing missions, including DARPA's PHOENIX mission and NASA's OSAM-1 (formerly Restore-L) [105, 127]. The OSAM-1 spacecraft will host an additional payload named Space Infrastructure Dexterous Robot (SPIDER). This payload includes a lightweight $5\,\mathrm{m}$ robotic arm, designed to assemble 7 elements of a communication antenna [1]. The OSAM-1 mission was due to launch in 2020. NASA also developed the Tendon-Actuated Lightweight In-Space Manipulator (TALISMAN) in house, which was the first system designed exclusively for OOA [44]. This hardware was the cornerstone for NASA's Commercial Infrastructure for Robotic Assembly and Services (CIRAS) project, developed to demonstrate the ability of space-based robotic assembly by attaching and detaching solar arrays [21, 132]. However, most of these technologies remain large, with the FREND manipulator weighing $78\,\mathrm{kg}$, and the TALISMAN manipulator weighing $36.2\,\mathrm{kg}$ [39, 44]. In addition to this, none of the literature expands on how or why design choices were made, making the transfer of knowledge hard.

With the increasing appeal of both OOA and Active Debris Removal (ADR) industries beyond government organisations are starting to invest in robotic space technologies. The Kraken manipulator is a commercial project developed by Tethers Unlimited, it is a $4.2\,\mathrm{kg}$, 7 d.o.f. manipulator [163]. No further information in relation to the performance metrics or operating conditions of the system are available and it is unclear if development has moved beyond the mission concept. Although, Tethers Unlimited are on the payload team for the OSAM-1 mission, so it is possible that there is a technology cross over between the two mainpulators. Airbus are also developing the VISPA arm which is designed to be reconfigurable and versatile, with the aim of performing a wide range of On-Orbit Operations (OOO). The VISPA arm has a reach of $1.8\,\mathrm{m}$ and an estimated mass of $14.5\,\mathrm{kg}$ [91].

Although there is a clear commercial demand and a certain degree of Technology Readiness Level (TRL) for a small space robot mission there currently exists no immediate planned technology demonstrations. The implementation of a downsized system is a challenging task with problems arising from scaling down both the manipulator and base spacecraft. The trade-off when downsizing the manipulator is to reduce its mass and power consumption, while still maintaining a sizable dexterous workspace and high payload capacity. The preservation of the arm's dexterity is paramount since the tasks required for the assembly of the telescope

presented in Section 3 involve fragile components. The design approach presented in Section 4 addresses these issues by encoding the dynamic model of the system and solving a complex cost function using MOO. In doing so, it outlines the full design procedure for a small space robot, something that is not currently addressed in the literature. Instead, work in the field focuses on the presentation of the finalised designs, as is summarised in this section.

2.1.3 Robotic Technologies Specific to Telescope Architectures

There are only a handful of mission concepts that explore and outline the robotic architectures necessary to facilitate the OOA of large aperture space telescopes. Such proposals exist only in the literature with no investment for later phase development.

The earliest study of such a mission concept presented Autonomously Assembled Space Telescope (AAST), a $10\,\mathrm{m}$ Cassegrain focus telescope [14]. This telescope uses hexagonal mirror segments of up to $2.5\,\mathrm{m}$ in size. It highlights the development of suitable robotics as the priority for the next stage of the study [13]. However further work is not available and it is unclear if the feasibility study was taken further. Similar to the AAST study, Feinberg *et al.* present a modular telescope to be assembled on-orbit [48]. This design hosts a larger PM at $20\,\mathrm{m}$, but again, does not address what robotic technology would be appropriate for the assembly process.

The first study including both telescope design and robotic technology presents Robotically Assembled Modular Space Telescope (RAMST). Their work outlines a general purpose, tethered hexapod robot that can assemble a telescope with a $100\,\mathrm{m}$ spherical primary mirror [78]. While the study demonstrates the use of force-control with a stereo vision system to perform some the of required tasks on Earth, it does this with only a dual-arm system [78]. Conversely, the proposed robotic agent suffers from complexity issues, with 6 legs each having 7 d.o.f. that need to be individually controlled by numerous mechanisms in an extreme space environment.

Another approach to robotic OOA involves fixed based robots. Roa *et al.* present such a robot, which is designed to assemble hexagonal mirror tiles [133]. Unlike the architecture used in the RAMST concept, the study specifies more details on how the telescope components have been designed specifically for OOA. While the work focuses on the assembly of the primary mirror, details on the additional assembly tasks (SM and structural components) are missing, as is an in-depth look at the operation and control of the robotic arm used in the assembly process.

Following the suggestion of a fixed base agent Nair *et al.* suggest the use of a $12.5\,\mathrm{m}$, 5 d.o.f. end-over-end walker to assemble a $25\,\mathrm{m}$ aperture telescope [104]. They outline the required movement patterns of the walker. This arm remains fixed to the base spacecraft throughout operation and was inspired by the European Robotic Arm which operates on the ISS [20]. The main advantage of this architecture is the low complexity and higher mobility compared to a fixed base robot. However, the use of a free-flying robot, such as that presented in Section 3 would remove the requirement for an arm of this length since the satellite could re-position itself relative to the payload. Another issue that a free-flying robot would mitigate is a potential obstruction of either a sensor's Field of View (FoV) or the FoV of the actual telescope. This is a design aspect that would need to be very carefully considered when locating docking points with an end-over-end walker or the robot itself in the case of a fixed base system. The main drawback of the free-flying space robot is the complexity of both the hardware design and the control. The problem of designing such a system is addressed in two different ways in this work. Section 4 explores a hardware centered design approach to small free-flying space robots, while Section 5 presents a system driven by deep RL to learn the best physical design and control policy pair.

2.1.4 Space Robot Control

The control of a space robot is a well explored topic with methods generally fitting into two main categories — free-floating and free-flying. Free-floating control is when the Attitude and Orbit Control Subsystem (AOCS) is not used on the base spacecraft. While this is fuel-efficient, it results in an undefined workspace and dynamic singularities [147]. Conversely with a free-flying space robot, the base is fully stabilised during arm motion. A number of well-established approaches exist for both types of control dependent on the specific use case [98, 147, 152, 161]. However, these approaches rely first on an offline trajectory planning stage prior to online tracking. This makes implementation slow and solutions lack versatility.

The use of deep learning will aid in integrating the path planning stage into the control algorithm, producing more robust solutions. As such, this thesis explores the use of RL to control the space robot. Hovell and Ulrich implement such a method by using RL in 2D rendezvous problems, named Deep Guidance [59]. Their network learns the correct velocity commands and feeds these to a conventional controller to allow for rendezvous between a chaser and a target. Wu

et al. employ a similar approach and use RL to learn to plan feasible trajectories for a space robot, which they demonstrate with both fixed and moving targets [173]. Hao *et al.* also use deep learning to look more specifically at how OOA could be achieved using a manipulator [54]. They present a ground-based hardware in the loop robotic demonstrator that simulates in-orbit manipulation. They use computer vision and AI to estimate the pose and position of the target object. Again, this work does not utilise RL to control the manipulator itself. Further work by Hao *et al.* looks at the control of the manipulator but this is achieved by learning optimal trajectories off-line by encoding trajectories as a Gaussian probabilistic distribution [55].

2.2 Classic Approaches to Robotic Hardware Design

Finding the optimal geometry of a robotic manipulator is one of the most complicated problems in robot kinematic design [74]. The area of terrestrial robotic hardware design optimisation has been extensively researched and there exists numerous varied solutions to the design problem. Contrary to this, the process of designing a free-flying space robot is under-explored and undocumented. As such, this section outlines a number of important terrestrial design solutions that can be utilised and manipulated to design a space robot.

Approaches to terrestrial design broadly fit into two groups, *task-based* and *performance-based* hardware optimisation. Task-based approaches aim to find a set of design parameters that best satisfy task requirements [70]. Whereas, with performance-based control, the robotic designs are evaluated on simple tasks where the emphasis is on ensuring that a high level of general performance can be achieved.

2.2.1 Performance-Based Hardware Optimisation

The very first robots were all designed via intuition and experience requiring experts with extensive domain knowledge. The majority of methods used to evaluate these early designs involved looking at where the manipulator could reach and how easy it was to change the position and orientation of the end-effector [177]. These initial approaches lead to the definition of a quantitative measure of these two qualities — named workspace and manipulability [178, 75]. Kucuk and Bingul optimsed the design of a 3 d.o.f. manipulator using a combination of both of these performance parameters [73]. They first maximise the robot's workspace, given mechanical

constraints and then evaluate its performance. This was done by ensuring that the kinematic and dynamic characteristics are satisfactory using a combination of the manipulability measure and condition number. Li and Dai implement a similar method where they use the workspace of a manipulator to determine its reachable points [82]. These points are then evaluated to optimise the link lengths of a three-link manipulator for a given set of trajectories.

Manipulability is derived from the Jacobian Matrix. This matrix is fundamental in understanding the motion of the end-effector of a robotic manipulator and underpins a lot of early design approaches. Stucco *et al.* use the Jacobian to evaluate the performance of robots designed for different applications, ensuring that each is well conditioned to limit singularities [154]. They present an approach to bespoke robot design by integrating application specific requirements into a general performance function. This performance function is constrained by the desired workspace. However, most Jacobian based performance indices suffer from very significant limitations driven by scale dependence, non-homogeneity and dimensional dependence and therefore its direct use is now usually avoided [117].

While this type of design approach values good general performance and well-conditioned systems, there is less focus on the performance on any one task. As such, task-based approaches have started to appear. These generally build on the performance-based approaches by using the same metrics but evaluating success and fine tuning designs on specific tasks.

2.2.2 Task-Based Hardware Optimisation

One of the first approaches to task-based hardware design constructed the framework known as 'Progressive Design'. This breaks the full design task was down into kinematic design, planning, and control [69]. The kinematic design was solved by optimising design parameters to fit the task specification, in this case, a reachability constraint, singularity constraint, and joint angle constraint. Park *et al.* built on the 'Progressive Design' approach by introducing the Grid Method to improve the efficiency of the kinematic design stage [115]. They apply this optimisation technique to the design of a 2 d.o.f. planar manipulator for a nuclear power plant, where the task again provides a number of optimisation constraints.

Al-Dois *et al.* formed a multi-objective weighted function consisting of the task-time and joint torques to optimise the robot's trajectory, link lengths and mass [3]. They used this to optimise

a 3 d.o.f. manipulator subject to kinematic and dynamic constraints based on the required task — arc or spot welding.

Kivela *et al.* moved away from the traditional task-based optimisation approach by arbitrarily setting the robot parameters and cycling through a number of task points to calculate the total error for the given configuration [70]. They found the optimal link lengths for a redundant 6 d.o.f. manipulator by finding the configuration that lead to the minimal total error between the end effector position and a number of task points. As with all task-based approaches, this requires an in-depth knowledge of the tasks the robot needs to carry out.

Carlone and Pinciroli proposed a design method that instead approaches from a subsystem level, therefore requiring less specific task knowledge [25]. They look at the optimal design of a robot subject to cost constraints and performance maximisation. They reduced the problem into one involving non-continuous variables. This was done by parameterising the design into a number of modules (*i.e*. motors, sensing, frames), each with a performance and cost value. Designs were then formed using a stack of modules constrained by overall cost.

While a range of solutions exist, by far the most popular approach to the task-based optimisation of terrestrial robots is to use a multi-objective cost function [101, 116, 138, 181]. Each selects a variation of performance and kinematic design parameters specific to the desired task. The design parameters are then optimised using the MOO framework. A summary of common terrestrial robot performance parameters can be found in [117]. In the approach presented in Section 4, the space robot manipulator is optimised using MOO. To this end, a set of objective functions will be defined based on the performance parameters pivotal to the successful operation of the space robot.

2.3 Multi-Objective Optimisation

A MOO problem is defined as the process of optimising systematically and simultaneously a collection of conflicting objective functions [33]. The issue when solving MOO problems comes from the fact that the search space becomes partially ordered and Pareto fronts are not necessarily convex, meaning solutions may be locally optimal as oppose to globally optimal [2]. There is rarely a single global optimal solution to such a problem and instead a number of Pareto-optimal solutions exist [93]. A Pareto-optimal solution is a solution that is not worse in

any of the objectives and better in at least one [2]. An entire set of such optimal points is then known as the Pareto optimal set, and can be joined to make the Pareto optimal front. The Pareto front is very hard to define for a multi-objective problem. Extensive research has been done into MOO methods, with a number of summaries available in the literature [93, 47, 67, 34]. Methods to solve these problems can be categorised into three groups, those that require information as an input (*priori articulation*), those that require the user to select from a number of optimal designs (*posteriori articulation*) and those that require no user input (*no articulation of preferences*) [93].

2.3.1 Priori Articulation

Priori articulation methods allow a user to specify preferences in terms of goals or relative importance of the different objectives. One of the first approaches to solving MOO problems is the global criterion method. The aim is to minimise the sum of all objective functions/goals thereby finding a decision vector that is as close to the optimal solution as possible [32]. However, this method quickly came under scrutiny for not providing truly optimal results and being reliant on knowledge of a near optimal decision vector [174].

The weighted sum method builds on the idea that it is unlikely for a single solution to optimise all objective functions simultaneously [92]. First introduced in 1963 by Zadeh, it has been prominent in the field ever since [180]. This method follows a similar approach to the global criterion method but objective functions each have a weighting allowing for consideration of user preferences [93]. General use of this approach provides an acceptable Pareto optimal solution, and the weighted sum method is presented as a tool in the literature [92]. The drawback of this method is that it reduces a multi-dimensional problem to a single objective, meaning the complexity of the decision space may not be fully captured.

The weighted min-max method can also be used but this involves the calculation of a Utopian objective vector [93, 27]. This vector is calculated by subtracting a small positive scalar from the ideal objective function, which itself represents the lower bound of the Pareto optimal set; therefore making the Utopian vector unattainable in the feasible region [95]. The process of calculating both of these vectors is not only computationally expensive, but requires prior knowledge; it is therefore avoided. The Lexicographic method can be used to solve MOO problems. A pre-defined order is used to solve all objective functions in order of importance

[153]. This method is computationally expensive and was shown to not satisfy typical optimality conditions [130].

2.3.2 Posteriori Articulation

Methods with posteriori articulation allow the user to view options in the Pareto optimal set before making a decision, meaning there is no need to consider which objective function is of more or less importance. Das and Dennis presented the Normal Boundary Intersection Method that finds several Pareto optimal points for general nonlinear MOO problems [37]. As a collection, these points capture a number of different trade-offs within the problem, however this was later shown to skip some optimal points [93]. Following this, Messac *et al.* developed the Normal Constraint Method [94]. This improved on the Normal Boundary Intersection Method by only producing Pareto optimal points.

2.3.3 No Articulation of Preferences

In the instance in which no user preferences can be defined, no articulation of preference methods are employed. These methods are mostly simplifications of Priori articulation methods with the exclusion of decision parameters [103]. One such example is to use the weighted-sum approach where all weights are set to 1 [93].

The modified classic design approach presented in Section 4 uses the Priori articulation method, more specifically the weighted sum approach. This is because the required input information can be inferred from the tasks outlined in Section 3. Methods with posteriori articulation are also utilised in order to select which d.o.f. system is desirable. Any other posteriori selections are avoided since they rely on intricate knowledge of how each design parameter will affect system performance. This is often infeasible due to the undocumented, non-linear, non-holonomic behavior of the system.

2.4 Reinforcement Learning

RL is the process by which an agent learns a behavior through trial and error interactions in a dynamic environment. Early approaches to solving RL problems include Temporal Differ-

ence (TD) learning, Monte Carlo methods and Dynamic Programming [156]. However, these approaches generally lack scalability and are limited to low dimensional problems. The advent of deep learning and Neural Networks (NNs) has significantly impacted the field, and most popular RL methods now use NNs as powerful function approximators — known as deep RL. Deep RL algorithms can be split into three main groups, *Q-learning* methods, *policy gradient* methods and *actor-critic* methods.

2.4.1 Q-Learning

With Q-learning the aim is to learn or estimate how good any state-action pair is and then select actions accordingly. The field of deep RL was really set in motion with the introduction of Deep Q-Network (DQN) [97]. This algorithm was the first example of an artificial agent that was capable of learning to succeed at a range of tasks from minimal inputs (only pixel and game scores), using the same network architecture and training algorithm.

DQN was improved upon via the introduction of Double DQN, in which a second state-action value approximator was introduced to prevent the over estimation seen with DQN [167]. Both of these methods use a replay buffer to store experiences that is then sampled from during training. A number of techniques have been introduced that improve on this sampling method. Prioritised experience replay ranks these experiences and means more important ones can be sampled more frequently [141]. Hindsight Experience Replay (HER) modifies inputs in the replay buffer such that they represent a successful tuple, therefore helping with the problem of insufficient exploration [5].

2.4.2 Policy Gradient Methods

Policy gradient methods work by directly learning the optimal policy that will lead to success in a task. From evolutionary strategies to direct searches, a broad range of approaches to finding the optimal policy exist [83]. The REINFORCE rule underpins the first widely used policy gradient methods [171]. It was used to learn stochastic policies that were capable of tracking images and capturing objects [143, 175]. Silver *et al.* proposed the Deterministic Policy Gradient (DPG) as a method that allowed for learning of continuous action spaces [150]. This outperformed stochastic policy methods in a number of benchmark problems. However, this algorithm sometimes suffered from overfitting and unstable updates.

Schulman *et al.* then introduced the idea of trust regions and presented Trust Region Policy Optimisation (TRPO) [144]. This approach constrained the Kullback–Leibler divergence between the new and old policy to prevent large policy updates and ensure monotonic improvement of the stochastic policy. Proximal Policy Optimisation (PPO) followed the development of TRPO, showing similar results but with much simpler implementation [145]. It was shown to outperform current online policy gradient methods and to find a balance between sample complexity and train time.

2.4.3 Actor-Critic Methods

Actor-critic methods involve learning both a value function and a policy, therefore combining the advantages of policy search and Q-learning methods. This makes them a sub-set of policy methods [7]. Note that a system learns the value of a state and the Q value of a state-action pair. A2C and A3C were introduced by Mnih *et al.* in 2016, named advantage actor-critic and asynchronous actor-critic respectively [96]. For Atari games, A3C was shown to run much faster than DQN but final performance was no better [83].

Lillicrap *et al.* advanced DPG by combining it with DQN, named Deep Deterministic Policy Gradient (DDPG), therefore making it an actor-critic method [86]. They showed success in 20 different environments and outperformed planned control algorithms that had access to the dynamics of each system. Actor-critic methods is are now widely employed with the introduction of a value network to methods such as PPO and TRPO showing higher levels of performance.

2.5 Simultaneous Design of Hardware and Control

Co-optimisation of morphology and control has been of interest for decades in the field of robotics with approaches falling into three broad categories — *evolutionary algorithms*, *nonlinear programming* and *machine learning*.

2.5.1 Nonlinear programming

Nonlinear programming approaches use a full set of dynamic equations to solve for both the morphology and control. Park *et al.* first presented the idea of optimising a two link manipulator along with PID control parameters [114]. They solved the problem using a form of gradient descent and showed how the settling time of the system could be drastically reduced. Since this a number of different approaches to the co-optimisation of design and control using nonlinear programming have been presented, although the overarching techniques remain the same [126, 4, 81, 121].

More recently, Avila Belbute-Peres *et al.* proposed an approach to directly optimise robotic morphologies and solve simple control problems [38]. They achieve this by redefining systems using differentiable mathematical concepts on a case-by-case basis (similar methods are shown in [84, 40]). While the idea of this work is to integrate such blocks into an RL pipeline, this is not demonstrated in their work, and instead, a form of nonlinear programming is used to demonstrate the applicability of their environments. However, use of their method is limited since they require differentiable reward functions, which are typically sparse step functions based on the final state of a system and almost universally hold no information about the robotic morphology.

While the results of nonlinear programming approaches are somewhat promising, they rely on explicit knowledge of system dynamics and how these relate to the physical parameters. This becomes non-trivial for any complex system and is impossible to implement in the more common model-free scenario.

2.5.2 Evolutionary Computation

The idea of evolutionary computation for the co-optimisation of design and control was first proposed in 1994 by Sims [151]. This work, inspired by biological evolution, maintained a population of agents all with different morphologies and sensor placements. Over time these were mutated, and their performance evaluated until no mutation would further improve performance. In general, the field of evolutionary computation has received more attention compared to nonlinear programming, and is now a common approach for solving the co-optimisation problem [9, 110, 170]. While the results are promising these methods are computationally

expensive, data inefficient and encoding dependent. Hornby *et al.* addressed the problem of encoding, presenting a method that allowed for more complex agents to be expressed [58]. At each mutation stage they 'build' an agent, allowing it to reuse modules or add new ones — this is referred to as generative representation. In addition to this, these approaches are generally limited by the mutation of control parameters. As standard, each design and control pair is mutated once, and it is unlikely that this mutated controller will perform well on the new design. As such, it is typical for evolutionary approaches to fall into local minima.

More recently, ideas have combined evolutionary methods with an inner control loop usually optimised via machine learning. Wang *et al.* use the idea of evolutionary computation and formulate the automatic robot design problem as a graph search problem using neural graph evolution[169]. They use a graph NN as the control policy and set each design parameter as a node on a graph. They then search and mutate this graph to discover more optimal designs. Throughout policy training they allow the sharing of weights to increase efficiency and performance. Their method is capable of designing new configured robots as well as fine-tuning human-engineered results. However, due to the nature of their approach, environments must be re-defined and the computational demand is high.

Hejna *et al.* propose a combination of evolutionary algorithms and machine learning, named task agnostic morphology evolution [61]. While not strictly co-optimisation of control and hardware since pre-defined motion primitives are used, performance of their algorithm shows good results compared to those trained with task specific algorithms. They evaluate performance-based on the robot's ability to reach diverse states and the causality of their actions. The causality metric is the fitness function used in the evolutionary optimisation loop. However, performance evaluation is unrepresentative of real world tasks and is only tested in situations where the predefined motions can be used *e.g.* walking in a given direction.

2.5.3 Machine Learning

Ha bridged the gap between evolutionary computation and RL by using a population based policy gradient method to sample the optimal robot-agent pair [53]. Although their method yielded a higher reward in fewer training iterations when compared to the 'out-of-the-box' agent, the method is computationally expensive since a population of designs must be maintained throughout training. Schaff *et al.* propose a similar approach where they treat the parameterised

design as an additional input to the control policy [139]. Throughout training they maintain a distribution of designs and continually shift this distribution towards higher performing regions of the design space. Fundamental to the success of their method is that their control policy maintains generalisation over morphologies in the distribution — a widely documented challenge in the field of RL [89]. Schaff *et al.*'s later approach for more general agent design suffers from the same limitation of needing to maintain performance over a range of morphologies [140].

Luck *et al.* noted the need to maintain a distribution of designs as a limitation and instead decompose the problem as a bi-level optimisation [88]. They alternate optimising the morphology and the control policy. By learning Q-values over the design distribution and using this to pre-evaluate agent performance prior to testing, they reduced the size of design distribution that must be maintained. This method is reliant on learning Q-values over a wide distribution of robotic designs, or risk missing out on the optimal design due to poor function approximation.

More recently, HWasp was presented. In this approach the effects of the robotic hardware needing to be optimised are re-defined separately from the rest of the environment as a differentiable computational graph [30]. The control policy and hardware policy are then optimised in a standard RL framework. This approach lacks generality since it requires a redefinition of the action space, unique to each problem, as well as the inclusion of bespoke computations for the graph implementation. HWasp also suffers from an inability to fully optimise changes in morphology or link dimensions. Instead, modifications are made to kinematic structure (inclusion of a mass damper system at either end of a link) in order to allow for the necessary modifications to the action space to be implemented. The approach proposed in Section 5 does not suffer from such limitations and the physical dimensions of robots can be optimised directly using back-propagation.

Bolland *et al.* employ a similar method to HWasp by parameterising the systems reward function, in addition to the transition function and the policy [18]. They assume that all are differentiable with respect to their parameters. They solve the co-optimisation problem by iteratively approximating the gradient of the expected return via Monte Carlo methods, and taking gradient ascent steps in the environment and policy space. In contrast, the method proposed in Section 5 does not require a differentiable reward function, which as discussed previously can be considered a limiting requirement — shown by the simple task settings in which Bolland *et al.* evaluate their work [18].

2.6 Robust Reinforcement Learning for Control

One issue in the field of RL is a lack of generality to other robotic embodiments. Lundell *et al.* show how a system trained to execute the ball in cup task, via RL, for a specific string length is unsuccessful for any other string length [89]. This issue falls under the overarching issue of generalisation or robust RL. Generalisation looks at the ability of a robot to operate successfully in an environment beyond that which it was trained in [112]. In the context of this thesis, the changes in environment of interest are triggered by a change in embodiment.

2.6.1 Domain Randomisation

Training via domain randomisation has been shown to improve the ability of a policy to operate in a range of target environments. Domain randomisation has had most success in the field of sim-to-real transfer, whereby agents are trained in simulation and then successfully deployed on a physical robot [159, 136, 28, 118]. Tan *et al.* train a quadruped robot on randomised environments, showing that this produces a more robust policy [157]. In a similar approach, Stulp *et al.* show how domain randomisation can make an agent robust to state estimation uncertainties [155]. However, success of any domain randomisation approach is dependent on the correct knowledge of acceptable randomisation range otherwise and agent may never learn a useful policy [179]. Dai *et al.* show a comparison between different RL agents and how they perform under visual domain randomisation [36]. They show how specific algorithms are not necessary and most RL algorithms show increased robustness when trained in this manner.

2.6.2 Multiple Policies

The idea of using multiple policies to achieve a level of generalisation falls most closely to evolutionary methods and therefore exhibits similar limitations. Yu *et al.* propose using multiple policies with varying behaviors, where each is optimised for a particular dynamic vector [179]. At the search stage, the highest performing policy is selected using co-variance matrix adaptation. Devin *et al.* also explore the idea of having multiple trained networks that they draw from based on the task at hand [41]. They decompose neural network policies into 'task-specific' and 'robot-specific' modules, where 'task-specific' modules are shared across tasks and 'robot-specific' modules across different robots. The recombination of these modules after training allows for

different tasks to be carried out on different robots. However this requires training across options simultaneously, and tasks cannot be learned in sequence.

2.6.3 Meta-Learning

Meta-learning is the concept of learning to learn and allows the system to learn to adapt to new situations [142]. By nature, these techniques all require some experience in the target environment, therefore not displaying zero-shot transfer. The RL^2 algorithm is an example of such a technique. This algorithm exploits a recurrent neural network set up with value and policy functions [46]. The trajectory at each layer, along with the current state, is used as the input to the next layer. RL^2 works with a model-free RL algorithm. Despite this, multiple tests of this algorithm have shown that it is no better at generalising than standard RL algorithms [46, 112].

Following this work, Finn *et al.* developed the Model-Agnostic Meta-Learning (MAML) algorithm [49]. This algorithm identifies sensitive parameters that allow for a small number of gradient updates to facilitate fast learning on a new task. Early evaluation of this algorithm shows that it can adapt to new environments much better than standard RL algorithms, only if changes do not affect the dynamics of the system. Tests were carried out by varying the goal velocity and direction of a half-cheetah system.

Anne *et al.* address the limitation that meta-learning is only used for multi-task learning and present a meta-learning control policy that can achieve fast adaptation to changing dynamic conditions [6]. They train a context vector that parameterises the physical state of the robot along with the initial weights of the controller. Finding the optimal controller weights for all context vectors means that as the dynamic situation of the robot varies the controller is able to quickly adapt.

2.6.4 Adversarial Learning

Adversarial learning is a popular framework used to train robust RL control policies. It works on the basis of modeling environmental changes as adversarial disturbances. Mandlekar *et al.* show how policies can be trained using adversarial examples to generate robust control for use on real systems [90]. However, all evaluation tests are carried out on the same robotic system

since the aim is to make agents robust to malicious attacks. Rajeswaran *et al.* proposed the Ensemble Policy Optimization (EPOpt) algorithm [125]. This dual-step approach consists of policy optimisation using samples from a batch of tasks whereby the distribution of tasks is updated using the lowest performing experiences. This algorithm was designed to be robust to perturbations, where changes in the agent's environment were modeled as such disturbances. This was achieved by varying the torso mass and friction damping in a 2D hopper task. It was shown that the EPOpt trained policy was able to generalise to a range of hopper models carrying out the same task each time. However, further research into this algorithm showed that its performance, in both interpolation and extrapolation, was comparable to that of standard deep RL algorithms [112].

Pinto *et al.* propose an approach where the system is robust to changes in the dynamics, named Robust Adversarial Reinforcement Learning (RARL) [122]. They modeled uncertainties as an adversarial agent that disturbs the system. They evaluate the generalisation of the learned policy by training it with certain mass and friction values and test it with different values. Their algorithm proved more robust than the random baseline policy in a third of all test cases, leaving much room for improvement.

Shioya *et al.* propose two modifications to the RARL architecture, the first introduces a penalty term to the adversary's loss function for sampling training examples that fall far from the current state [148]. The second utilises sampling from a memory bank of adversary networks. Both of these methods aimed to tackle the problem outlined by Bansal *et al.* , whose experiments showed that always using the hardest environment can hinder the training of the agent [11]. The use of a Learning potential (l_p) in Section 6 accounts for Bansal *et al.*'s findings.

2.6.5 Other Approaches

Tasse *et al.* employ logical composition within an RL framework to train an agent that can determine if it can solve a new task with existing knowledge or if a new skill must be learned [158]. If the latter, their method generates an estimate of the optimal policy in order to speed up learning. This guarantees a level of generalisation over an unknown task distribution, however no generalisation exists over changing robotic morphologies. By nature this also does not employ zero-shot transfer.

Chen *et al.* propose a method that is considered current state-of-the-art in the context of hardware-agnostic policies [29]. They augment a physical parameter vector with the environmental observation and use this as the agent's observed state. During training the physical parameter vector is randomly selected from a discrete distribution of 7 possibilities. Testing is then carried out on these 7 possibilities plus 1 more. While they report strong results compared to DDPG with HER, the scope of operation of their agent is limited by the training distribution. On top of this, some policies are fine-tuned in the test environment. While this is not as laborious as re-training, zero-shot transfer is required for policies to be truly hardware-agnostic. They also show that their explicit method falls short in situations when the task policy is dependent on agent dynamics.

The method for training hardware-agnostic policies presented in Section 6 builds on the approach suggested by Chen *et al.* [29]. It improves upon the random sampling of embodiments during training via the introduction of an intelligent, adversarially inspired network. It learns a distribution over embodiments on which the system can learn but currently has low performance. The use of a Normalising Flow Network (NFN) allows this distribution to be learned and sampled from without prior knowledge of its complexity or shape [71]. Additionally, the system in Section 6 is set-up to utilise 'expert' knowledge allowing the system to adapt already learned behavior as opposed to trying to distill multiple different experts.

2.7 Summary

This chapter first introduced a wide variety of space telescope architectures and space robot technologies. While the recent launch of the JWST was a vast leap for on-orbit mirror technologies, more innovative ideas need to be developed in order to launch even large space telescopes. This is apparent in the push to start designing modular telescopes that can be assembled on-orbit. There exists a number of different approaches, although little work looks into the design of a telescope suitable for OOA with space robots or the design of the space robot itself.

This chapter then introduced methods for jointly learning the control and morphology of robotic systems. While there exist a number of different approaches, most do not involve actual co-optimisation with the morphology and control being optimised in an iterative manner. Methods presented in this thesis overcome this by allowing for simultaneous reasoning over hardware

and control. In addition, the majority of current methods require high computational demand and are not easy to implement.

Finally, this chapter investigated current approaches to robust control under changing environmental dynamics. While many solutions exist to robust control under varying tasks and other parameters, few focus on varying dynamics, with even fewer displaying zero-shot transfer. The work in this thesis is the first to provide this robustness and zero-shot generalisation to changing environments.

Chapter 3

Mission Concepts for the On-Orbit Assembly of a Large Aperture Telescope

A well designed free-flying space robot will have applications in a wide range of missions, including aiding in the assembly of large structures, servicing of active satellites and ADR. While the scope for such a system is vast, only one concept is investigated in this thesis. However, the findings of this research can be applied to other on-orbit missions and feasibility studies. The selected mission looks at the OOA of a large aperture modular telescope using free-flying space robots. The mission is expanded and investigated in this chapter in order to provide quantifiable requirements for the control and design of the spacecraft.

This chapter starts by providing details of the overall mission concept, and the corresponding tasks that are required to achieve successful OOA. It then outlines how the simulator used to model the tasks and space robot were defined.

3.1 Mission Concept

The assembly mission will take place in GEO. This orbit has been selected for a number of reasons, all of which stem from the fact that the effects of atmospheric density and gravity are negligible at this altitude. Firstly, zero-g conditions are easier to model compared to

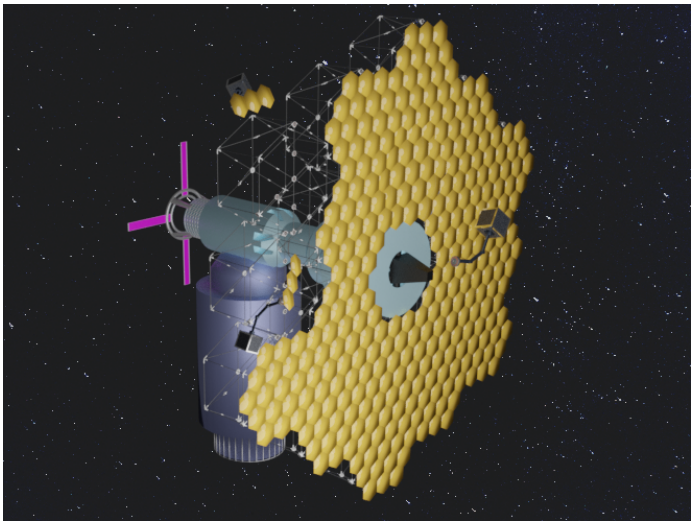


Figure 3.1: **Telescope being assembled on-orbit.** The docked transport satellite is attached to the main telescope hub. Also visible are a number of different space robots carrying out assembly tasks. The wire type structure is the assembled BP and the yellow parts are PM segments. The pink appendages at the rear are solar panels.

microgravity, making the dynamic model of the system more accurate. Secondly, the lack of atmosphere means that drag on the space robot and telescope components is also negligible. As a result, orbital maintenance manoeuvres will not be required and components should remain at a constant altitude for an extended period of time. This limits the fuel requirements of the space robot and overall complexity of the mission. In addition to this, in GEO the telescope and space robot will experience only very short eclipse periods which will improve power generation capability.

In GEO the telescope will always be observable from the same ground station. While teleoperation will not be used, observability is still important to ensure that real-time monitoring and aborts can take place. These attributes would not be guaranteed if the telescope were to be assembled in Low Earth Orbit (LEO). The disadvantage of the assembly taking place in GEO is the increased fuel requirements. However, in an attempt to combat this, the space robots will be on a ‘piggy-back’ launch, as will other components where possible.

3.1.1 Telescope Architecture

The telescope is designed using as many available technologies and previous missions as possible in order to maximise TRL and lower costs. Therefore, the design is a combination of previous work with a number of small modifications. As such, only a summary is provided here and a detailed design case is presented in Appendix A. The telescope is built around a main hub which will be the first component of the system to launch. The hub itself does not require any assembly. Instead, three main parts require assembly or deployment and these are the drivers of this mission concept:

1. **Segmented PM assembly**: The PM is the main light gathering surface on a reflective telescope. Since resolution is dependent on how much light the telescope can collect, the size of this mirror governs the overall performance of the telescope.

2. **Modular Back-plane (BP)**: This is the structure that supports the PM modules and acts as the contact between the PM and the main telescope housing.

3. **SM**: This is a smaller mirror that is used to redirect and refocus the light reflected from the PM.

These components are shown in Figure 3.1. Following the launch of the main telescope hub, the BP is the first structure to be assembled. The BP modules are attached in two rings around the hub — named the inner and outer ring. Each BP module is made of struts and nodes that deploy into a hexagonal structure. These modules make a main structure onto which the PM segments are mounted. The inner ring can be seen in Figure 3.2, and the outer ring attaches directly to this in a concentric manner. Also visible in Figure 3.2 are a number of Intelligent Space System Interface (iSSI) modules — a standard interface used to connect components. The use of a standard interface is paramount to the assembly process in order to minimise complexity and demand on the space robot. All components connect using this hardware and grasping of each is achieved via the same interface. As such the robotic manipulator needs only to know how to grasp and connect a single interface, reducing complexity in the end effector architecture and control. The robotic manipulator's end effector is one of these interfaces. This particular standard interface was chosen since it provides strong mechanical properties and hosts a self-alignment mechanism. This removes the burden on the space robot to mechanically

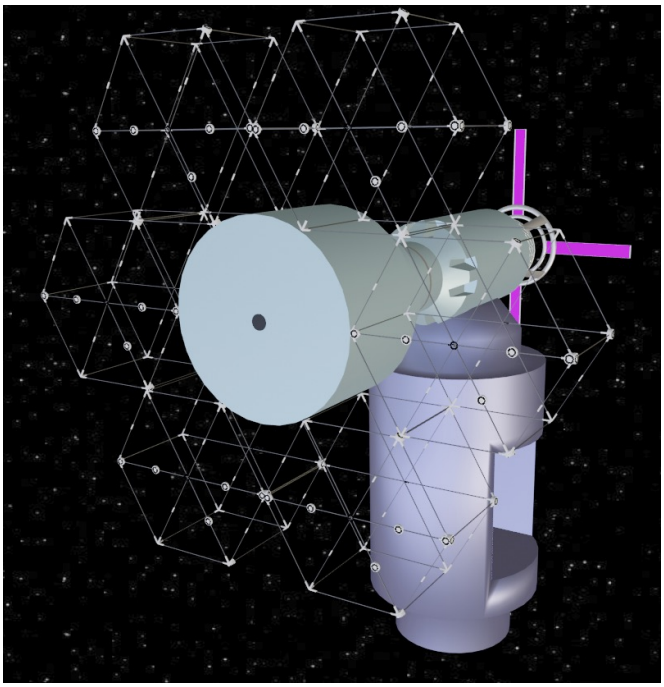


Figure 3.2: **Assembly of inner BP ring.** This shows the inner BP ring attached to the main telescope hub (light blue). The iSSI modules are shown as grey dots and the storage satellite is shown in purple.

latch components, an operation that is known to be complex. The BP modules will be launched on a transport satellite in their stowed configuration This satellite will dock with the main telescope hub. While this adds complexity to the storage satellite it means that the space robot is able to operate between two bodies with no relative motion. The space robot will grasp and then transport each of the BP modules to their correct location then aid and monitor their deployment. Note that the BP modules themselves have no level of intelligence and are not able to communicate.

Following the assembly of the BP structure, a second transport satellite containing the PM segments will launch. Like the first transport satellite this will dock to the main hub. The space robot then attaches each PM module in-place over the fully assembled BP structure. Upon completion this satellite will un-dock, leaving scope for the delivery of spare parts to the telescope when needed. Once all the PM segments have been attached, each mirror will undergo phasing. This ensures that the final telescope can take accurate and high quality images. This part of the assembly does not require the space robot. Finally, the SM will be deployed. This mirror extends beyond the PM structure and as with the phasing does not require assembly.

At this stage there is no specific timeline for the mission. Delays are possible between the launch of the storage satellites. However, upon docking, the space robot should begin and finish the assembly as soon as possible. The choice of GEO means that both the telescope and space robots can remain in a 'dormant' orbit while waiting for the launch of the next components with little fuel expenditure.

3.1.2 Assembly Missions

A number of different missions are required to achieve the full telescope assembly. This work employs a novel end-mission design approach whereby the full assembly process is broken down into a number of missions. These are then further broken down into sub-missions, which are parameterised by different payloads and success requirements. The advantage of designing the mission in this innovative way is that the robot can complete the full telescope assembly through a combination of several sub-missions. This minimises the complexity of the control, as optimisation/training can be performed on easily achievable and well-defined manoeuvres. Each sub-mission is made of a *requirement*, a *payload* and a *task*. The different options for these components are shown in Table 3.1. The *tasks* defined here correspond to the desired motion. When intelligent control is employed this motion is learnt and therefore is not explicitly encoded. However, in the absence of an intelligent controller, trajectories are provided that capture the desired motion for each task. Each trajectory is defined by a start and end system configuration and a high order polynomial is used to determine the intermediate joint angles. This overall motion is summarised in Table 3.2, the exact joint values are included in Table 3.3 for 3 options of d.o.f. While these are arbitrary estimations they act as a good basis for optimisation. The full telescope assembly can then be carried out by manipulating sets of $task + payload + requirement$ or in the case of the pre-defined trajectories, $task + payload$. This is because the $requirements$ are encoded in the trajectories and instead performance is evaluated by assessing how well the robot was able to achieve the desired motion.

Figure 3.3 shows how these smaller, lower level sub-missions are combined to allow for full telescope assembly. What is not captured in the $task + payload + requirements$ is the starting conditions of the robot, payload and other components. These are instead captured in the missions themselves, where a mission here refers to a number of sub-missions in series. The three missions highlighted in Figure 3.3 are the inner BP assembly mission, outer BP assembly

Table 3.1: **Sub-mission components.** In order to assemble the proposed telescope a number of $tasks$, success $requirements$ and $payloads$ are defined. These can be combined in different ways to make a sub-mission.

Tasks	Payloads	Requirements
Fine manipulation	PM module	Connect iSSI module
Grasp	BP module	Disconnect iSSI module
Relocation	Nothing	Move within any specified distance of a target location

Table 3.2: **Simplified OOA tasks.** Defined here are the motions that correspond to the $tasks$ outlined in Table 3.1 .

Task	Desired motion
Fine manipulation	A small end effector motion, starting with the arm in its 'zero' position and moving away from its Center of Mass (c.o.m). The base should remain stable throughout.
Grasp	Large manipulation of the arm with a stable base. The arm should start away from its 'zero' position and move back towards this.
Relocate	No movement of the arm with a large motion of the base. The arm should be kept in its initial position throughout base motion.

mission and the PM assembly mission — where each represents the assembly of a single module in a single location. In practice multiple of these missions would occur with slightly varying start and finish states. Since the motion of the robot will be the same in each case only the outlined missions are evaluated in this work.

3.2 Space Robot Dynamic Model

In order to investigate the behavior of the space robot an accurate model of the dynamic system is needed. As prescribed by the nature of work in this thesis, this model must be parameterised by the space robot's physical hardware. This makes the implementation different to that already available in the literature. An in-depth derivation is not included here since it is an amalgamation of prior work tailored to this particular use case in this work. Instead, an overview of the required

Table 3.3: **Trajectories for OOA.** The final poseof each d.o.f. of the system for a number of different tasks are given here. First three terms of the start and final pose are in m while the rest are in $^\circ$.

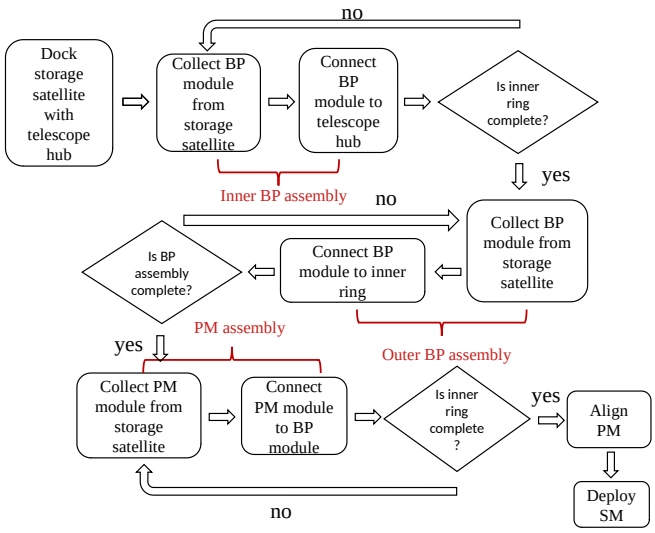
	Task	Start Pose	Final Pose
5 d.o.f.	Fine manipulation	$0,0,0,0,0,0,0,0,0,0,0$	$0,0,0,0,0,0,-5,5,-20,-30,90$
	Grasp	$0,0,0,0,0,0,5,-5,0,-45,-90$	$0,0,0,0,0,0,0,0,0,0,0$
	Relocate	$0,0,0,0,0,0,0,0,0,0,0$	$1,-1,1,0,0,0,0,0,0,0,0$
6 d.o.f.	Fine manipulation	$0,0,0,0,0,0,0,0,0,0,0,0$	$0,0,0,0,0,0,-5,5,-20,-30,20,90$
	Grasp	$0,0,0,0,0,0,5,-5,0,-45,-20,-90$	$0,0,0,0,0,0,0,0,0,0,0,0$
	Relocate	$0,0,0,0,0,0,0,0,0,0,0,0$	$1,-1,1,0,0,0,0,0,0,0,0,0$
7 d.o.f.	Fine manipulation	$0,0,0,0,0,0,0,0,0,0,0,0,0$	$0,0,0,0,0,0,-5,5,-20,-30,20,5,90$
	Grasp	$0,0,0,0,0,0,5,-5,0,-45,-20,-30,-90$	$0,0,0,0,0,0,0,0,0,0,0,0,0$
	Relocate	$0,0,0,0,0,0,0,0,0,0,0,0,0$	$1,-1,1,0,0,0,0,0,0,0,0,0,0$

equations is given. The final equations used match those presented by Seddaoui *et al.* [146]. These equations are valid based on the following assumptions:

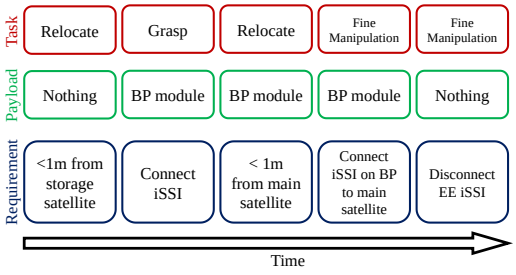
- Each link acts as a rigid body.
- Either payload capture hasn't occurred, or it has occurred and the payload is modeled as an additional rigid body at the end of the manipulator. Contact dynamics are not considered.
- The spacecraft has suitable sensing capability.
- When the arm is not actuated, the system will orbit with the expected characteristics of a standard rigid body.
- Arm actuation time is short, so orbital effects are negligible during operation.

3.2.1 Spacecraft Definition

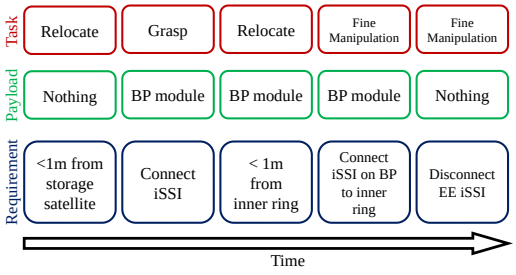
As defined in Section 1.1, a space robot is a robotic manipulator attached to a base spacecraft. The robotic arm is made up of a number of different links, with joints connecting successive



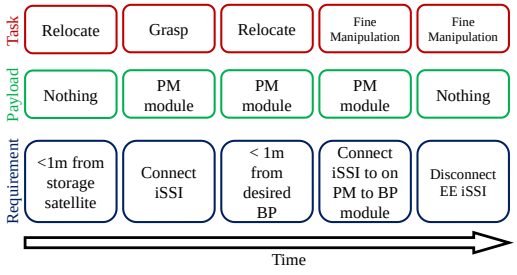
(a) Full Assembly Mission



(b) Inner BP Assembly Mission



(c) Outer BP Assembly Mission



(d) PM Assembly Mission

Figure 3.3: **Novel breakdown of missions for OOA.** Different combinations of $task + payload + requirement$ are needed to achieve all the sub-missions required for the full telescope assembly. This shows a breakdown of how these are combined in order to achieve assembly of the inner and outer BP rings and the PM.

Table 3.4: **Standardised form factor sizes and dimensions.** The size of the base spacecraft is defined using the CubeSat standard. The form factor is the number of units used to make the base spacecraft. The dimensions are given here, where d_x, d_y and d_z are the width, depth and height of the base and m_{sc}, its mass. A mass range is given since this is hardware dependent.

Form Factor	$3U$	$6U$	$12U$	$18U$	$24U$	$27U$
m_{sc} (kg)	$4 < ... \leq 8$	$8 < ... \leq 16$	$16 < ... \leq 24$	$24 < ... \leq 30$	$30 < ... \leq 35$	$35 < ... \leq 40$
d_x, d_y, d_z (m)	$0.1, 0.1, 0.3$	$0.2, 0.1, 0.3$	$0.2, 0.2, 0.3$	$0.3, 0.2, 0.3$	$0.4, 0.2, 0.3$	$0.3, 0.3, 0.3$

links with an end effector as the last link. The base spacecraft will take the form of a box which can house a number of required sub-systems.

Throughout this thesis, the base spacecraft is defined using the CubeSat standard, where satellites are made from a number of units (U). Units are $0.1\,\mathrm{m}\times\ 0.1\,\mathrm{m}\times\ 0.1\,\mathrm{m}$, each of which is estimated to weigh $1.33\,\mathrm{kg}$ [123]. Table 3.4 summarises the dimensions and mass of a number of different form factors, where d_x is the width of the base, d_y is the depth and d_z is the height, all in m. m_{sc} is the mass of the base in kg. A mass range is given since the exact mass depends on the selected subsystems. For all form factors, the base spacecraft has N_{sc} = 6, where N_{sc} is the total number of d.o.f. of the base. Rotations and translations are measured by the movement of the body fixed reference frame ($\sum B$) w.r.t. the inertial reference frame ($\sum I$). Figure 3.4 shows the discussed reference frames.

The robotic manipulator can have any number of d.o.f., where the total number is denoted as N_m. N_m can take any value from 0 to ∞, although $N_m = 6$ is sufficient for full arm dexterity. Any additional d.o.f. will add complexity and redundancy into the system. The total d.o.f. of the system (N) is therefore,

$$N = N_{sc} + N_m. \tag{3.1}$$

The manipulator is made of a number of links, with each link referred to in turn as i. Each has its own reference frame at the joint ($\sum l_i$) which is used to define the rotation of that d.o.f. or link — these are shown in Figure 3.4.

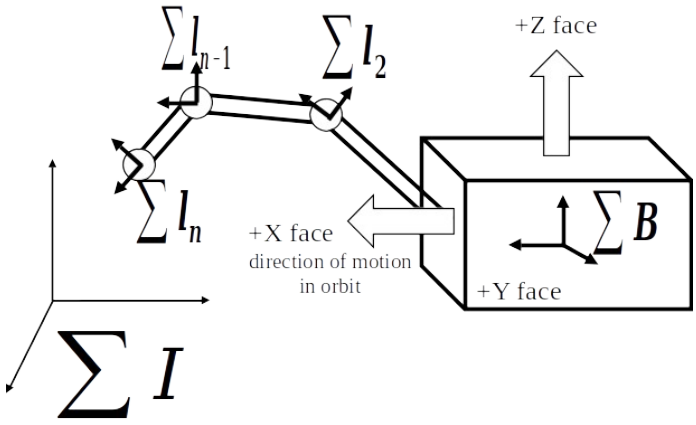


Figure 3.4: **Reference frames used with the space robot.** The base position is defined as a movement of $\sum B$. $\sum I$ is the inertial reference frame. The position of each link is described by the rotation of $\sum l_i$, which is fixed to the base of the corresponding link. Also shown here is the definition of the base spacecraft faces.

3.2.2 Dynamic Coupling

When operating in-situ, the space robot will experience dynamic coupling. This is a result of the microgravity operating conditions and the conservation of linear and angular momentum [31]. It means that any controlled or uncontrolled movement of the manipulator will induce a force in the base spacecraft and vice versa. If this induced force is large enough it will cause the base spacecraft to deviate from its desired position. It is possible for the AOCS to counteract this, but this is reliant on the subsystem being sized correctly as well as having accurate predictions of the induced motion that needs to be accounted for. This is a particular issue for downsized space robots as this mechanism is more prominent when the mass and inertia of the manipulator and payload are not negligible in comparison to the base [176].

3.2.3 Force Definition

The equation used to calculate the general forces in the system is defined as follows:

$$\boldsymbol{\tau} = \mathbf{D}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} \tag{3.2}$$

where,

$$\mathbf{q} = \begin{bmatrix} x & y & z & \alpha & \beta & \gamma & \theta_i \cdots \theta_{N_m} \end{bmatrix}^T. \tag{3.3}$$

$\mathbf{q} \in \mathbb{R}^N$ is the state vector of the system, making $\dot{\mathbf{q}} \in \mathbb{R}^N$ the corresponding velocities and $\ddot{\mathbf{q}} \in \mathbb{R}^N$ the accelerations. In Equation 3.3, the first 3 terms represent the linear position of the base (x, y, z) and the next 3 terms represent the attitude of the base (α, β, γ), all w.r.t. $\sum I$. The next N_m terms in the state vector represent the displacement of each joint of the robotic manipulator w.r.t. to that link's reference frame — $\sum l_i$.

The $\mathbf{D} \in \mathbb{R}^N$ matrix represents the inertial properties of the system:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{sc} & \mathbf{D}_{scm} \\ \mathbf{D}_{scm}{}^T & \mathbf{D}_m \end{bmatrix}, \tag{3.4}$$

where the subscript $_m$ relates terms to the manipulator, $_{sc}$ to the spacecraft and $_{scm}$ to the coupling between the base and manipulator. The same subscripts apply to the $\mathbf{C} \in \mathbb{R}^N$ matrix although this matrix represents the Coriolis and Centrifugal properties:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{sc} & \mathbf{C}_{scm} \\ \mathbf{C}_{scm}{}^T & \mathbf{C}_m \end{bmatrix}. \tag{3.5}$$

$\boldsymbol{\tau}$ is defined as the forces acting on each d.o.f. of the system,

$$\boldsymbol{\tau} = \begin{bmatrix} f_x & f_y & f_z & \tau_\alpha & \tau_\beta & \tau_\gamma & \tau_{\theta i} \cdots \tau_{\theta N_m} \end{bmatrix}^T, \tag{3.6}$$

where each element corresponds to that in state vector $\mathbf{q}$, making all but the first three elements torques.

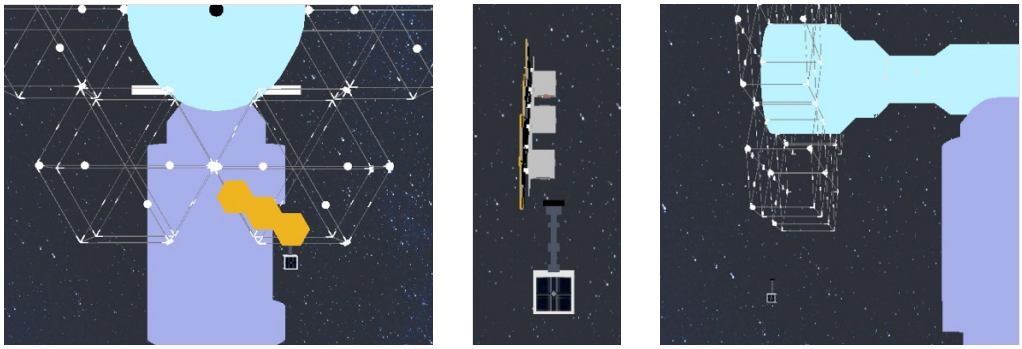
In order to control the space robot, it is possible to apply a force to any of the N d.o.f. In order to determine the resultant motion of this applied force, and therefore the behavior of the space robot under control, Equation 3.2 must be solved for $\ddot{\mathbf{q}}$,

$$\ddot{\mathbf{q}} = \mathbf{D}^T \left(\boldsymbol{\tau} - \mathbf{C}\dot{\mathbf{q}} \right) \tag{3.7}$$

this is done either using a system of Ordinary Differential Equations (ODEs) or Simulink.

3.2.4 Implementation

In order to successfully model the behavior of the space robot the dynamic model is integrated into a semi-realistic simulator. The simulator is used throughout this thesis, both in an RL pipeline and as a standalone system. A Simulink implementation is used when individual trajectories are used to quantify tasks, and a Python implementation is utilised when an RL pipeline is employed in order to exploit auto-differentiation.



(a) Connect PM module (b) Grasp PM module (c) Relocate for outer BP

Figure 3.5: **Example of tasks in python simulator.** Demonstration of render function in python space robot simulator. Three different task and payload combinations are shown.

Simulink Implementation

Simulink was selected for use since it is well suited to solve systems of differential equations, such as Equation 3.7. The dynamics were encoded as a block diagram with the space robot hardware parameters, control forces and start & final positions as inputs. The block diagram was set up to calculate $\ddot{\mathbf{q}}$ which was then integrated twice to give $\dot{\mathbf{q}}$ and $\mathbf{q}$ — making up the state vector. The starting point for integration was taken as the value from the previous timestep, or the start state when $t = 1$. Each simulation was run over 200 timesteps, with integration occurring over a dynamic range of 1 to 0.001 timesteps, decided internally at run-time. No render function was used and instead, introspection was done via extraction and plotting of intermediate states. Due to the nature of trajectory modeling in this simulator, no collisions need to be handled.

Python Implementation

The dynamic system was also implemented in Python. This system was modelled around the OpenAI gym Application Programming Interface (API) and uses a differentiable ODE solver to solve Equation 3.7 [23]. Integration occurs over a fixed timestep of 0.01 using the Runge-Kutta 4^{th} order method. Episodes have a maximum run time of 200 timesteps. The simulator is set up to include the different tasks outlined in Section 3.1.2. The Pandas3D physics engine is used to model collisions between payloads and the space robot as well as handling the task

completion requirements. The Pandas3D package also provides a render function that allows for introspection of each episode, an example can be seen in Figure 3.5. Photorealistic rendering is not needed since training occurs via the state space rather than the pixel space. The state space is made up of the $\mathbf{q}$, $\dot{\mathbf{q}}$, the payload's location in $\sum I$, its size, and its mass — giving a dimension of $2N+10$. The action space has dimensions N, allowing a control over all d.o.f.

A reward scheme is required in this environment in order to facilitate RL training. The same shaped reward scheme is used for all tasks. The per-timestep reward is the negative of the magnitude of the positional error between the end effector and the target location. A final reward of $+1000$ is given for successful completion of the task with various penalties for undesirable behavior. This includes a penalty of -30 for exceeding the limits of the actuators mounted on the arm, any collisions, and going over the allowable time limit. The magnitude of the final base velocity is given as a penalty in the final timestep in order to encourage the system to complete the task in a controlled manner. As such, the maximum achievable reward depends on the starting position of the space robot relative to the target payload.

Chapter 4

Task Driven Automated Hardware Design

In this chapter, Ta-DAH Design is presented — a method to improve the performance of a space robot via hardware optimisation. It also combats the slow speed of hardware design via the implementation of an automated design approach — thus addressing Objective 1. The proposed solution utilises a novel cost function in an iterative, automated, task aware pipeline. The design approach is driven by simulation-based statistics, therefore leveraging prior knowledge and facilitating the faster output of optimal designs.

The approach is applied to the task of optimising a small space robot for the OOA of the large aperture telescope outlined in Section 3.1. The motivation for designing a smaller space robot is to provide a low cost alternative to the large, bespoke systems currently available. It is not a replacement for such systems, but instead a solution to the problems that cannot be solved with a larger system. This work provides new insight into the physical implications of the dynamic coupling effect, something which is yet to be covered in the literature, although its theory has been investigated extensively by researchers. While it is possible to tailor this automated design process to a number of different applications, this is outside the scope of this chapter.

This chapter first defines the problem area, before outlining the automated design process used. Analysis of Ta-DAH Design is provided as well as the optimal designs for the OOA tasks. The chapter ends with a conclusion that motivates the next chapter of this work.

4.1 Problem Definition

While the benefits of downsizing a space robot are numerous, it is a challenging task with problems arising from scaling down both the manipulator and base spacecraft, as discussed in Section 1.1. The trade-off when downsizing the manipulator is to reduce its mass and power consumption while still maintaining a sizable dexterous workspace and high payload capacity. Reducing the mass of the arm will lower launch costs, and reducing its power consumption will limit demand on the base spacecraft. The preservation of the arm's dexterity is paramount since some assembly tasks involve fragile components and the connection and disconnection of joints is an arduous task; yet to be carried out on-orbit.

4.2 System Requirements

A number of system requirements are defined to ensure the optimisation process captures the desired attributes, these are summarised in Table 4.1. This table does not encapsulate all subsystem level requirements, but those that apply in the context of this dynamic, task based design approach. For example, capability for intelligent perception or up-link and down-link rates do not affect the dynamic interaction between the arm and base and are not considered at this time.

4.3 Methodology

Ta-DAH Design for space robot hardware optimisation is a two-step process. It first addresses the size of the manipulator and then the size of the base. Figure 4.1 shows the full process. The size of the manipulator is determined using a constrained weighted sum optimisation approach, which is solved in two separate steps. First, a coarse exhaustive search identifies the most promising region of the design space and then the Nelder-Mead Simplex search method is used to refine the design [77]. Constraints are set by the system requirements and target tasks. The size of the base spacecraft is optimised to ensure that the system can operate successfully. This is done by mounting the chosen manipulator on the smallest form factor and evaluating its performance. The size of the base spacecraft is increased until a number of conditions are met. This design approach puts emphasis on outputting the smallest operational system possible.

Table 4.1: **System requirements.** The system requirements for the full space robot system. These drive the design optimisation process. FR denotes a functional requirement. PR denotes performance requirement and DR a design requirement.

Requirement	Description
FR_01	System will be able to independently manipulate a payload.
FR_02	System will operate in the free-flying mode.
PR_01	System will be able to manipulate the BP and PM modules.
PR_02	End effector will apply forces greater than $50N$.
PR_03	Manipulator will have a length of $< 1m$.
PR_04	End effector will have accuracy of at least $\pm$ 2mm
DR_01	Base spacecraft mass will not exceed $100kg$.
DR_02	Manipulator will weigh less than $10kg$.

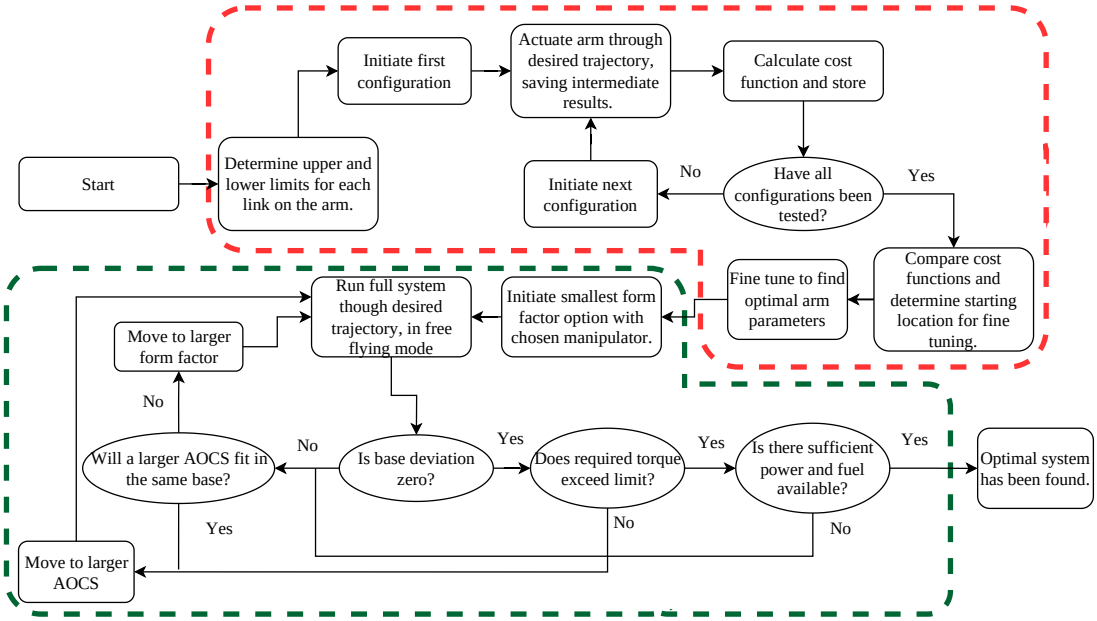
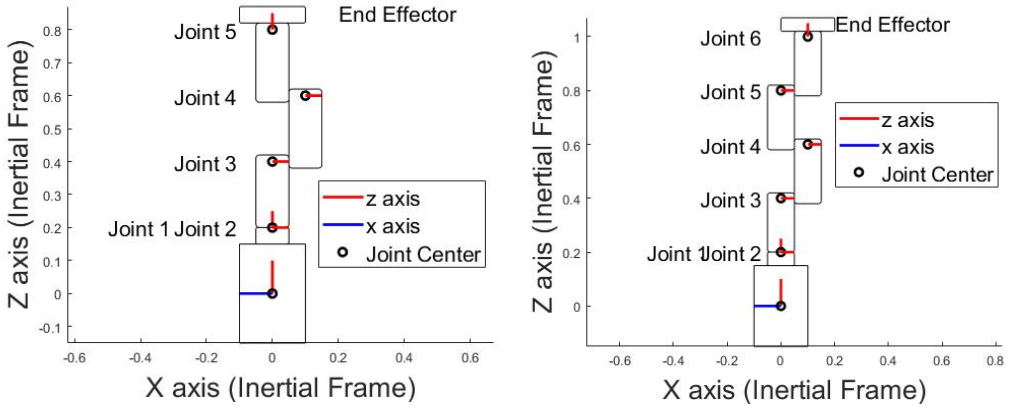


Figure 4.1: **Ta-DAH Design methodology.** The full optimisation process for the space robot is shown here. Highlighted in red is the section that relates to the optimisation of the arm while the green part highlights the base optimisation procedure. Each part is expanded and explained in the following section.



(a) 6 d.o.f. system

(b) 7 d.o.f. system

Figure 4.2: **Joint configurations.** Different d.o.f. manipulators are considered in this work, however the overall schematic remains the same. Joint 1 and 2 act at the same location to replicate a single joint having 2 d.o.f. Additional Joints are added after Joint N_m-1. All Joints other than 1 and N_m rotate about their z-axis.

A PID controller is used in the Ta-DAH Design pipeline. This controller works by outputting the required forces to reduce the error ($\mathbf{e}$) between the current state of the system ($\mathbf{q}$) and the desired state of the system ($\mathbf{q}_d$),

$$\boldsymbol{\tau}_t = k_P \mathbf{e}_t + k_I \int \mathbf{e}_t + k_D \frac{d\mathbf{e}_t}{dt}. \tag{4.1}$$

It is constrained by the tuple (k_I, k_D, k_P). In order to implement this controller a target trajectory is needed. $\mathbf{q}_d$ is given in Table 3.3. In order to optimise the system for each specific task, the space robot is actuated through the corresponding trajectory and the resulting motion and forces are used for evaluation. The same controller is used throughout.

4.3.1 Arm Optimisation

The principle of the weighted sum MOO problem is to determine a set of parameters in order to minimise a cost function ($\mathbb{C}$), where

$$\mathbb{C} = \sum_{j=1}^{J} \mathbf{w}_j \mathbb{F}_j \left(\mathbf{v}\right). \tag{4.2}$$

Here, $\mathbf{v} \in \mathcal{V}$ is a design vector from the design space, $\mathbb{F}_j$ represents each objective function, and $\mathbf{w}_j$ is the weight given to the corresponding function. J is the total number of objective

Table 4.2: **Definition of selected objective functions.**

	Definition
$\mathbb{F}_1\ (\mathbf{v})$	Reach of the Manipulator
$\mathbb{F}_2\ (\mathbf{v}, \mathbf{q}(t))$	Manipulability of the manipulator configuration
$\mathbb{F}_3\ (\mathbf{v}, \mathbf{q}(t))$	Error between the CoM of the end effector and the desired location
$\mathbb{F}_4\ (\mathbf{v}, \mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$	Forces and torques applied to the base during arm actuation
$\mathbb{F}_5\ (\mathbf{v}, \mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$	Torque required at each joint to move the arm at a given velocity and acceleration

functions used. At this stage in the Ta-DAH Design pipeline the design vector is defined as,

$$\mathbf{v} = \begin{bmatrix} l_1 & l_2 & \dots & l_{N_m} \end{bmatrix}^T, \tag{4.3}$$

where $l_{1..N_m}$ represents the length of each link in the manipulator. The same joint configuration is adapted for use with a manipulator of any d.o.f., as seen in Figure 4.2. Regardless of how many d.o.f. the manipulator has, joint 1 and 2 operate at the same location, mimicking $l_1 = 0$ and providing rotation about two axes at this point. The other joints provide rotation about that link's z-axis and additional d.o.f. are added as joints prior to Joint N-1. Changes are made directly in the system model to account for rotations between the link frames themselves, *e.g.* between joint 1 and 2. Additional d.o.f. are not added by the end effector due to the use of a standard interface — details can be found in Appendix A.

The objective functions selected for use with Ta-DAH Design represent fundamental parameters necessary for successful operation of the space robot for specific tasks. All the functions are summarised in Table 4.2 and are discussed and formalised in turn.

Objective 1

Objective function 1 relates to the system's pseudo-workspace. Workspace is not directly considered since, due to their nature, space robots have only an 'attitude confined' workspace. This becomes a function of path history, and therefore the selected orbit [162]. This means orbit corrections could be used to satisfy workspace demands. As a result, workspace is a 'soft' constraint. The overall reach of the manipulator is fundamental instead, since this constrains close proximity and rendezvous capability for any given orbit. $\mathbb{F}_1(\mathbf{v})$ is the sum of the link

lengths, which in this case is the sum of the design vector components,

$$\mathbb{F}_1(\mathbf{v}) = \sum_{i=1}^{N_m} l_i. \tag{4.4}$$

Objective 2

Manipulability is a measure of the capability of a robot to execute a specific task in a given configuration and is a function of the dynamics and kinematics of the robot. This property governs objective function 2. Manipulability is a way to quantify the ability of a manipulator to move and orientate its end effector within a given workspace. This quantifies the dexterity and prevalence of singularities within the workspace. $\mathbb{F}_2(\mathbf{v}, \mathbf{q}(t))$ is defined as

$$\mathbb{F}_2(\mathbf{v}, \mathbf{q}(t)) = \sqrt{det\left[\mathbf{J}(\mathbf{v}, \mathbf{q}(t))\,\mathbf{J}(\mathbf{v}, \mathbf{q}(t))^T\right]}. \tag{4.5}$$

$\mathbf{J}$ is the Jacobian of the manipulator and $\mathbf{q}$ is the state vector of the system, defined in Section 3.2.3. $\mathbb{F}_2(\mathbf{v}, \mathbf{q}(t))$ is a local metric, meaning it is posture and therefore time dependent. In each case this will be calculated for the final position of the manipulator.

Objective 3

The error between the c.o.m of the end effector and the desired location, defined by the task, is objective 3:

$$\mathbb{F}_3(\mathbf{v}, \mathbf{q}(t)) = \mathbb{I}\left(\mathbf{v}, \mathbf{q}_d(t)_{6:N_m}\right) - \mathbb{I}\left(\mathbf{v}, \mathbf{q}(t)_{6:N_m}\right). \tag{4.6}$$

The subscript $_{6:N_m}$ denotes that only the forces applied to the arm are used here. $\mathbb{I}$ is used to represent the inverse kinematics of the system and is used with the desired and actual arm configuration. t is the time step in the simulation. Again, this is a local not global metric and the objective function is time dependent. This is avoided as it adds another level of dimensionality to an already complex problem. Instead, $\mathbb{F}_3(\mathbf{v}, \mathbf{q}(t))$ is redefined to capture the worst case value for the given trajectory,

$$\mathbb{F}_3(\mathbf{v}, \mathbf{q}(t)) = \max_t \left[\mathbb{I}\left(\mathbf{v}, \mathbf{q}_d(t)_{6:N_m}\right) - \mathbb{I}\left(\mathbf{v}, \mathbf{q}(t)_{6:N_m}\right)\right]. \tag{4.7}$$

Objective 4

Pivotal to the performance of the space robot is its ability to remain stable throughout operation. The effect of dynamic coupling can be quantified by looking at the torques and forces applied to the base spacecraft during arm actuation. This will be used as objective function 4 which is defined as

$$\mathbb{F}_4\left(\mathbf{v}, \mathbf{q}\left(t\right), \dot{\mathbf{q}}\left(t\right), \ddot{\mathbf{q}}\left(t\right)\right) = \left[\mathbf{D}\left(\mathbf{v},\mathbf{q}(t)\right)\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{v},\mathbf{q}(t), \dot{\mathbf{q}}(t)\right)\dot{\mathbf{q}}\right]_{1:6}. \tag{4.8}$$

$\mathbb{F}_4$ is a local metric and must be modified to account for the total effect of the dynamic coupling throughout the manipulator's trajectory:

$$\mathbb{F}_4\left(\mathbf{v}, \mathbf{q}\left(t\right), \dot{\mathbf{q}}\left(t\right), \ddot{\mathbf{q}}\left(t\right)\right) = \sum_{t=0}^{T}\left[\mathbf{D}\left(\mathbf{v},\mathbf{q}(t)\right)\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{v},\mathbf{q}(t), \dot{\mathbf{q}}(t)\right)\dot{\mathbf{q}}\right]_{1:6}, \tag{4.9}$$

where T is the total time for that task.

Objective 5

Due to the nature of the satellite, body-mounted solar panels will be used. Combining this with the idea of making the system as small as possible presents the issue that power will be a finite resource. Power consumption is therefore considered when defining the objective functions. Objective function 5 is defined as the power required to actuate the arm and its payload,

$$\mathbb{F}_5\left(\mathbf{v}, \mathbf{q}\left(t\right), \dot{\mathbf{q}}\left(t\right), \ddot{\mathbf{q}}\left(t\right)\right) = \left[\mathbf{D}\left(\mathbf{v},\mathbf{q}(t)\right)\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{v},\mathbf{q}(t), \dot{\mathbf{q}}(t)\right)\dot{\mathbf{q}}\right]_{6:N_m}. \tag{4.10}$$

Again, this is time-dependent and is modified to account for the total power required throughout actuation;

$$\mathbb{F}_5\left(\mathbf{v}, \mathbf{q}\left(t\right), \dot{\mathbf{q}}\left(t\right), \ddot{\mathbf{q}}\left(t\right)\right) = \sum_{t=0}^{T}\left[\mathbf{D}\left(\mathbf{v},\mathbf{q}(t)\right)\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{v},\mathbf{q}(t), \dot{\mathbf{q}}(t)\right)\dot{\mathbf{q}}\right]_{6:N_m}. \tag{4.11}$$

Final Cost Function

It is necessary to normalise each of the objective functions. This is done prior to the summation to ensure they are non-dimensional. If $\overline{\mathbb{F}}_j$ represents the normalised function then,

$$\overline{\mathbb{F}}_j = \frac{\mathbb{F}_j}{|\,\mathbb{F}_{j\,\max}\,|}. \tag{4.12}$$

The maximum value $\left(\overline{\mathbb{F}}_{j\,\max}\right)$ corresponds to the maximum value for any configuration in the testing range. This produces an upper bound of 1, with no lower bound, also acting to scale the objectives and provide equal weighting. A normal weighted sum cost function contains a number of functions to minimise. However, in this case, some of the objective functions quantify characteristics to maximise, such as the manipulability and reach. As a result, these objective functions are added to the final cost function. The final cost function for use in the optimisation is therefore defined as

$$\mathbb{C} = -\mathbf{w}_1\overline{\mathbb{F}}_1 - \mathbf{w}_2\overline{\mathbb{F}}_2 + \mathbf{w}_3\overline{\mathbb{F}}_3 + \mathbf{w}_4\overline{\mathbb{F}}_4 + \mathbf{w}_5\overline{\mathbb{F}}_5. \tag{4.13}$$

Setting all $\mathbf{w}_{j=1}$ will give all the objective functions the same governance over the results due to the normalisation. As such, this work does not alter the weightings, however it should be known that these can be tuned if a designer is looking for any particular quality from the system. For example, setting $\mathbf{w}_2 = 2$ will output a more dexterous robot of a bigger size for the same task. At this stage it is impossible to optimise over d.o.f. and instead this must specified manually. It is suggested that this is achieved by optimising the design of 5 to 7 d.o.f. manipulator systems and comparing final performance.

4.3.2 Base Spacecraft Optimisation

The previous section illustrates how the optimisation of terrestrial robotic manipulators can be adapted for use with a space robot manipulator. However, no similar technologies exist in relation to the base spacecraft, and as such, no techniques are available for modification. Instead, a new approach is taken in the Ta-DAH Design pipeline, whereby the smallest size of the system that is capable of supporting the arm during successful operation is determined. The form factor can take a number of discrete sizes, unlike the manipulator which inhabits a continuous design space. The options considered in this work are shown in Section 3.2.1.

Ta-DAH Design starts with the smallest form factor and evaluates its performance with the task optimal manipulator. If this meets a number of conditions then it is optimal, but if that base cannot support the arm then a bigger one is selected and performance is re-evaluated. This process continues until an optimal system has been found. This requires a clear definition of the requirements that make the system operational. Since the aim of this work is to design a system for a given task, not to optimise a full mission concept, a number of conditions are excluded

Table 4.3: **Definition of selected conditions.** Summary of the conditions used to optimise the base spacecraft in the Ta-DAH Design pipeline.

Condition	Description
1	Zero deviation during free-flying mode
2	Sufficient power is available to run all the required subsystems
3	Sufficient fuel is avaliable to perform arm manouvers

here, for example up-link and down-link capability. It has been determined that the spacecraft will be operational if all the conditions in Table 4.3 are met, these are driven by the system requirements.

Condition 1

Condition 1 states that the AOCS should maintain zero deviation along the desired trajectory. Throughout arm actuation the AOCS must apply the equal and opposite force induced by the dynamic coupling in addition to the required control forces for base motion. A larger manipulator relative to the base will increase the dynamic coupling and therefore the demand on the subsystem. In addition to this, a smaller base will be unable to house a more powerful AOCS. This condition will be checked by analysing the required torque for zero deviation and comparing it to what is available from the appropriately sized AOCS.

The torque required from the AOCS can be calculated based on the configuration of the reaction wheels in the spacecraft and the force needed to rectify the position of the base. If matrix $\mathbf{A}$ dictates the reaction wheel position in the spacecraft, the torque requirement for each reaction wheel $\left(\mathbf{h}^R\right)$ can be calculated using

$$\mathbf{h}^R = \mathbf{A} \cdot \begin{bmatrix} \tau_\alpha & \tau_\beta & \tau_\gamma \end{bmatrix}^T. \tag{4.14}$$

If one reaction wheel is mounted in each axis matrix $\mathbf{A}$ is defined as

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.15}$$

Table 4.4: **Properties of Reaction Wheels.** A number of COTS reaction wheels have been selected. Each have different torque and size properties [15, 107, 22, 57, 134].

System	Max Torque ($\mathrm{mN\,m}$)	Max Speed (R.P.M)	Required Form Factor
UniSat-7	3.7	7000	12
SatBus 4RW0	5.9	6500	12
RSI 12-75	90	7500	24
HoneyBee Robotics Microsat CMG array	112	12000	24
Bradford RW	265	4000	27

and solving Equation 4.14 gives

$$\mathbf{h}^R = \begin{bmatrix} \tau_\alpha & \tau_\beta & \tau_\gamma \end{bmatrix}^T. \tag{4.16}$$

Equation 4.16 is then used to determine the required torque for zero-deviation. This is compared to available hardware in order to determine if condition 1 can be satisfied with the current manipulator and base spacecraft pair. A number of AOCS subsystems are proposed for use, these are summarised in Table 4.4. Commerical Off-the-Shelf (COTS) components are selected in order to keep manufacturing costs to a minimum. The torque quoted can be applied in each of the three axes.

Condition 2

The second condition requires the spacecraft to generate sufficient power to operate. The satellite will generate power using body mounted solar panels since deployable panels would limit the space in which the system could operate. The power available for each of the different form factors was calculated using the method outlined in [35]. It is assumed that the panels will be mounted on the $+/-Y$, $-X$ and the $+Z$ face, where the Y faces are perpendicular to the velocity vector and the $+Z$ face is space-facing — as seen in Section 3.2.1. This leaves the forward-facing panel $(+X)$ free to mount the arm and the Earth-facing panel $(-Z)$ free to mount antennae and other required hardware. The spacecraft is assumed to be in a GEO at around $35\,786\,\mathrm{km}$. The corresponding orbital average powers can be found in Table 4.5, these are compared to the power required from the AOCS, manipulator joints and linear thrusters to

Table 4.5: **Power generated by different form factors.** Average orbital power output by each of the form factors.

Form Factor	Average Power (W)
$3U$	4.76
$6U$	8.97
$12U$	9.52
$18U$	13.7
$24U$	17.9
$27U$	14.28

achieve the defined tasks. In practice the power draw will be considerably higher than this from running additional subsystems, however an on-board battery will supplement the power supply.

Condition 3

This condition specifies that the system must have sufficient fuel to facilitate zero-deviation from the desired trajectory for the given task. This requires the propulsion system to output sufficient thrust. I_{sp} is the specific impulse of the fuel being used in $\mathrm{N\,s\,kg^{-1}}$ and m_f is the mass of fuel in kg that must be expelled.

$$m_f = \frac{t \sum_{c=1}^{3} \boldsymbol{\tau}_c}{I_{sp}} \tag{4.17}$$

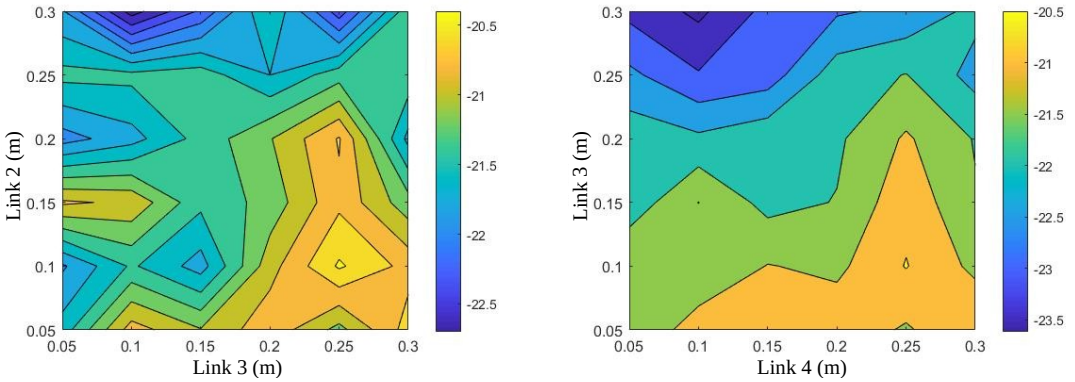
Equation 4.17 is then used to determine the required mass of fuel. A solid fuel is not suitable for use as it cannot be throttled and therefore a variable control force cannot be applied. Instead, a liquid or gas fuel will be used as this can be throttled and re-started. Electric propulsion is also not considered due to the already high power demands and long 'burn' times. A cold gas system will be used since it is small and simple to implement. Information on the linear thrusters considered for use during optimisation can be found in Table 4.6.

4.4 Results and Discussion

Prior to optimising the system for the OOA of a large aperture telescope using Ta-DAH Design, a number of simulations were run to generate and analyse statistical, quantifiable information

Table 4.6: **Linear Thrusters.** Comparison between cold gas linear thrusters. Some values are approximated [166, 164, 165].

	I_{sp} $\left(\mathrm{N\,s\,kg^{-1}}\right)$	Wet mass (g)	Max Power Consumption (W)
NASA's C-Pod Micro	40	1244	5
CuSP	70	690	11
AFRL Propulsion Unit	70	1005	15



(a) Link 2 and 3 varied

(b) Link 3 and 4 varied

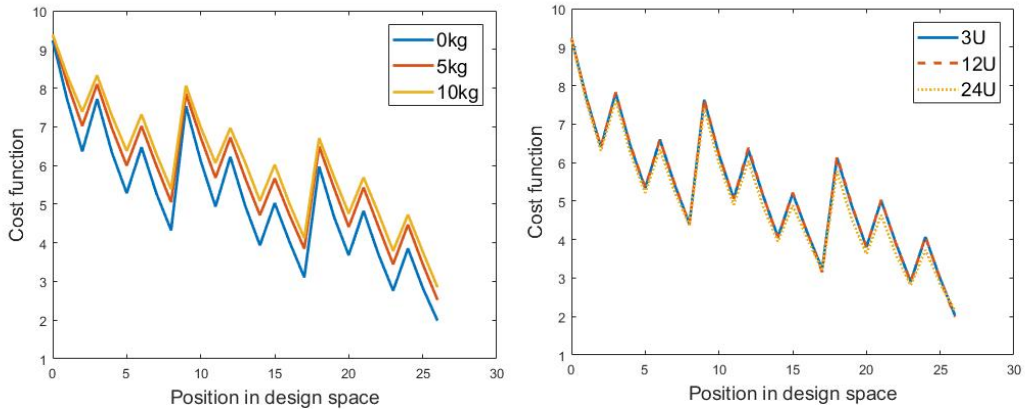
Figure 4.3: **Impact of link length on cost function.** Contour plot showing impact of varying link lengths on the cost function. In both cases the link not in question is held at a constant.

on the implication of different sized space robots. The aim of this is to provide usable metrics and outcomes for future space robotic missions.

4.4.1 Arm Optimisation

Exhaustive Search of Design Space

In order to investigate the shape of the design space, initially the dimensionality of the problem was reduced. The cost function was evaluated over two varying inputs in the design space. Figure 4.3 shows the emergence of both local and global minima, at differing locations, demonstrating the complexity of the design space.

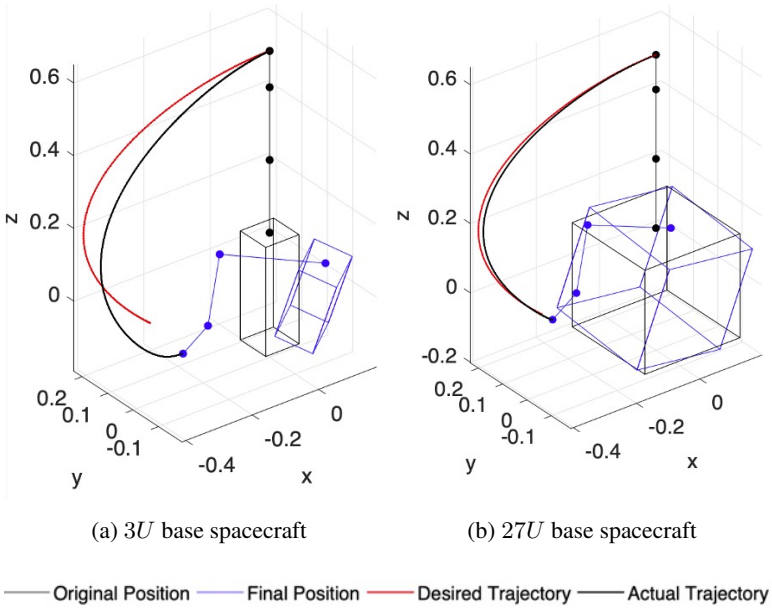


(a) Varying payload mass

(b) Varying base spacecraft size

Figure 4.4: **Cost function for specific manipulator.** Representation of the local minimas in the design space for a 5 d.o.f. manipulator with varying payload mass and base spacecraft size. The position in the design space is arbitrarily assigned for the figure axis.

An exhaustive coarse grid search was carried out for a range of feasible options in the design space and the cost function was plotted (Figure 4.4a). This is not the proposed solution for solving the problem but is used to visualise the position of local and global minima. Each link was evaluated from $0.1\,\mathrm{m}$ to $0.3\,\mathrm{m}$ at $0.05\,\mathrm{m}$ increments. It can be seen that there are a number of local minima in the cost function, these occur around the point at which link 2 is at its maximum. This search was carried out for 100 randomly generated trajectories, and it was found that, for a 5 d.o.f. system, 75% of the trajectories displayed local minima where link 2 was at its maximum. For a 6 d.o.f. system this was 64% and for 7 d.o.f., 58%. The same exhaustive search was carried out for arms mounted on different sized bases and with different payload masses. It was found that this had no effect on the location of the local minima (Figure 4.4b). The continuity in the position of these local minima in the design space is exploited in the Ta-DAH Design pipeline. In the first design stage, link 2 is fixed to its maximum length and a coarse grid search of the rest of the design space is carried out. The maximum length of link 2 is determined using the maximum reach (as defined in Table 4.1) divided by the number of links in the manipulator. The result of this initial search locates the starting point for the fine-tuning phase of the Ta-DAH Design process, which optimises the link lengths around the 'best' local minimum for that task. This two-stage approach means that the solution is more likely to be globally optimal — it also reduces computation time.



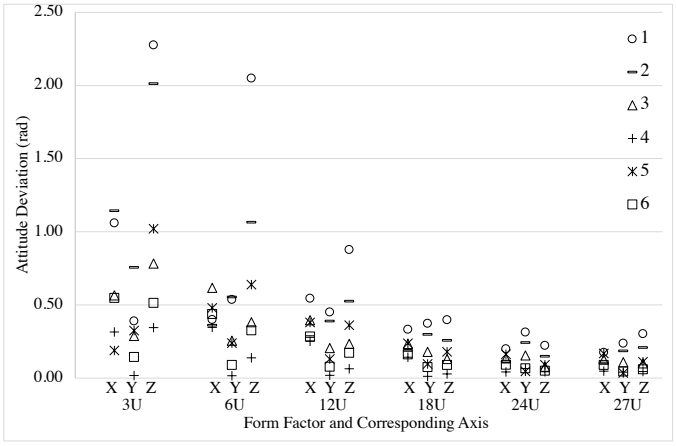
(a) $3U$ base spacecraft (b) $27U$ base spacecraft

Figure 4.5: **Dynamic coupling for different sized spacecraft.** Error in arm trajectory and the resulting base spacecraft deviation is shown here. This occurs if the dynamic coupling is not counteracted in the control system.

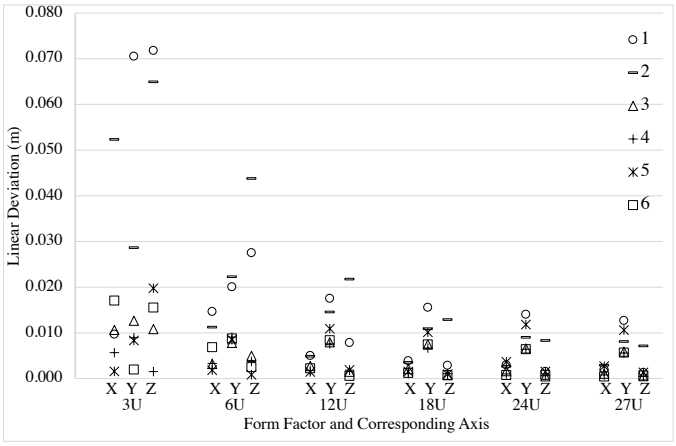
4.4.2 Base Spacecraft Optimisation

Prior to optimising a full system for the OOA of the large aperture space telescope, in the Ta-DAH Design pipeline, the behavior of the base spacecraft as a result of arm actuation was investigated. Of interest are the forces induced in the base, which are quantified by looking at the base's change in position, as a result of arm actuation. The other area of interest is the forces and torques required to counteract these induced forces since this will govern the system's stability.

The initial hypothesis tested was if increasing the size of the base spacecraft, in comparison to the arm, decreases the impact of dynamic coupling. An arbitrarily sized arm was mounted onto a $3U$ and then a $27U$ spacecraft and actuated through a single trajectory with the base operating in the free-floating mode. The final position of the base and the arm was used to illustrate the magnitude of the dynamic coupling — shown in Figure 4.5, The base deviation and resulting positional end effector error is larger when the base is smaller, this is because of the decreased arm to base inertial ratio.



(a) Attitude



(b) Translation

Figure 4.6: **Behavior of different sized base spacecraft in the free-floating operation mode.** Maximum base deviation for a range of form factors in the free-floating operation mode for each of the 6 random trajectories.

Knowing that a smaller base will experience larger deviation under the same manipulator motion, this relationship was further investigated using an arbitrary 5 d.o.f. manipulator, where $l_1 = 0\,\mathrm{m}$, $l_2 = 0.2\,\mathrm{m}$, $l_3 = 0.2\,\mathrm{m}$ and $l_4 = 0.1\,\mathrm{m}$. The simulation was run with each form factor for 6 random trajectories, where the arm is controller with a PID controller and no AOCS is used for base control. The maximum linear and attitude deviation was extracted, the results are shown in Figure 4.6. These graphs show the modulus of the maximum deviation value for the entire trajectory. Both the linear and rotary deviation decrease in an exponential manner with increasing form factor, as expected, due to the change in inertial ratio. Although the exact

numbers from this analysis are dependent on the size of the chosen manipulator, the trend holds for any small space robot. The exponential decay means that beyond a certain ratio, the effect of dynamic coupling becomes negligible — something that is utilised by the large robotic systems already in-orbit.

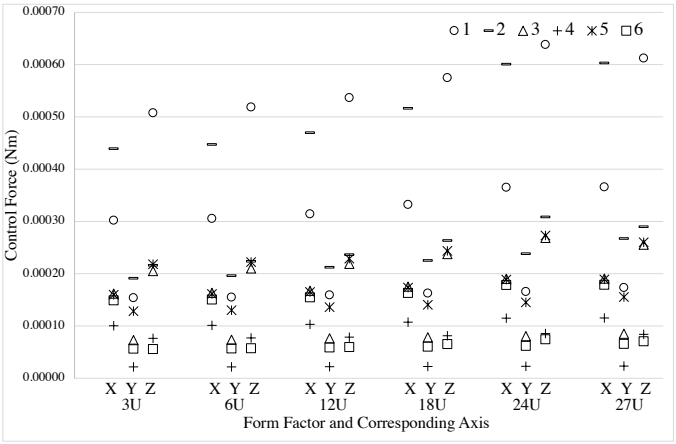
Although the effects of dynamic coupling are larger with a smaller base, it is possible that this does not translate into needing larger forces to rectify the effects. This is because first principles state that a system with smaller inertia will require less force to move. In order to evaluate this performance metric the spacecraft is operated with a PID controller aiming to maintain the base's position. It does this by applying the forces required to counteract the disturbance caused by the dynamic coupling — the same forces that trigger the behavior quantified in Figure 4.6. The same size arm and 6 trajectories used in Section 4.4.1 are used again. The maximum required forces to preserve this position, can be seen in Figure 4.7. For all options of form factor, across all missions, the resultant base deviation was approximately zero, and very small control torques are required.

The control torques increase only very slightly with form factor. This is counter to the deviation needed to be overcome. It can be concluded that a base with higher inertia needs higher control forces than a smaller one to move the same amount — in keeping with fundamentals. The design implication of this is that increasing the base spacecraft does not reduce the required control forces, although it does reduce the effects of the dynamic coupling.

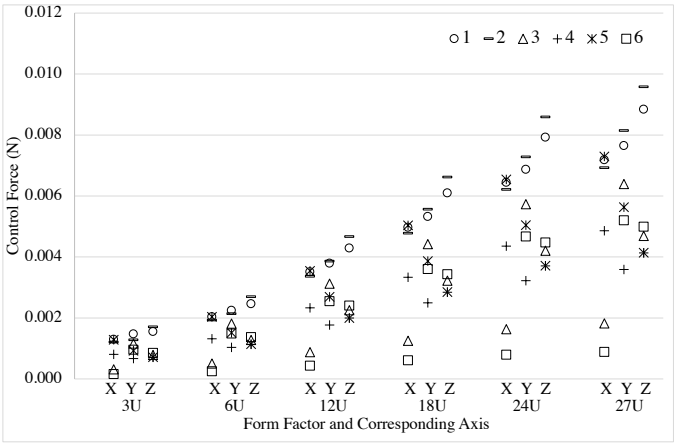
Conversely, the linear control forces are seen to increase with form factor. It was expected that these forces remain constant/decrease with increasing form factor since the linear deviation needing to be corrected falls. However, this is not the case, and in practice the larger base requires much higher forces to achieve the same motion compared to a smaller craft. This means that increasing the size of the base spacecraft to counteract the effects of dynamic coupling will put a higher demand on the linear control subsystem. It is therefore a trade-off between trying to limit the base disturbance due to arm manipulation and reducing the AOCS requirements.

4.4.3 Design of Space Robot for On-Orbit Telescope Assembly

Following the exploration of the design space of the space robot with 6 random trajectories, Ta-DAH Design was used to optimise a space robot for the OOA of a large aperture telescope. Since the current set-up does not allow for optimisation over d.o.f., three different options are



(a) Attitude



(b) Translation

Figure 4.7: **Control forces and torques required in the free-flying operation mode.** Maximum control forces required for a range of form factors in the free-flying operation mode for each of the 6 random trajectories.

analysed, these are a 5, 6 and 7 d.o.f. manipulator, mounted on a 6 d.o.f. base, giving a system with 11, 12 or 13 d.o.f.

As stated in Section 3.1.2, three different missions are required for the full assembly, these are the inner BP, outer BP and PM assembly — each of which is made up of $task + payload$ pairs. It is necessary to design a single space robot to achieve each of the three missions. This was done by actuating the robot through all the desired $task + payload$ sub-missions within a mission in Ta-DAH Design as opposed to just through a single trajectory. The cost function is summed across all $task + payload$ pairs. The 7 d.o.f. manipulator was not stable for any

Table 4.7: **Starting points for Ta-DAH Design.** Optimal arm configurations following the coarse grid search (denoted by 'Pre') are given for the 5 and 6 d.o.f. manipulators. These are then further optimised using the Nelder-Mead Simplex search method with the results quoted as 'Final'. All lengths are in m.

Task	d.o.f.		Link Length (m)				Total Reach (m)	Final Cost
			2	3	4	5		
Inner BP	5	Pre	0.3	0.25	0.05	-	0.6	436
		Final	**0.298**	**0.252**	**0.052**	-	**0.602**	**424**
	6	Pre	0.25	0.15	0.2	0.15	0.75	6685
		Final	**0.246**	**0.158**	**0.21**	**0.151**	**0.765**	**1673**
Outer BP	5	Pre	0.3	0.25	0.05	-	0.6	436
		Final	**0.298**	**0.252**	**0.052**	-	**0.602**	**424**
	6	Pre	0.25	0.15	0.2	0.15	0.75	6685
		Final	**0.246**	**0.158**	**0.21**	**0.151**	**0.765**	**1673**
PM	5	Pre	0.3	0.15	0.05	-	0.5	500
		Final	**0.315**	**0.151**	**0.049**	-	**0.615**	**439**
	6	Pre	0.25	0.15	0.15	0.15	0.7	4270
		Final	**0.263**	**0.151**	**0.151**	**0.139**	**0.704**	**1475**

mission, this was because the linear PID controller was not capable of functioning with such a complex highly non-linear system. This manifested in a highly erratic motion with a large deviation from the desired trajectory. This system is therefore discounted and not discussed or evaluated further. The starting designs for the final optimisation and the final designs are shown in Table 4.7, along with the corresponding final cost of each design. Figure 4.8 shows a schematic of each of the final designs. For all systems, the final optimisation step drastically reduced the cost of each design, with the smallest improvement in the 5 d.o.f. system when carrying out either task involving the BP. This mission is one with the absolute lowest cost anyway, this is followed by the same system with the PM assembly mission. For all missions, the 5 d.o.f. manipulator has a much lower final cost than the 6 d.o.f. system. For all missions

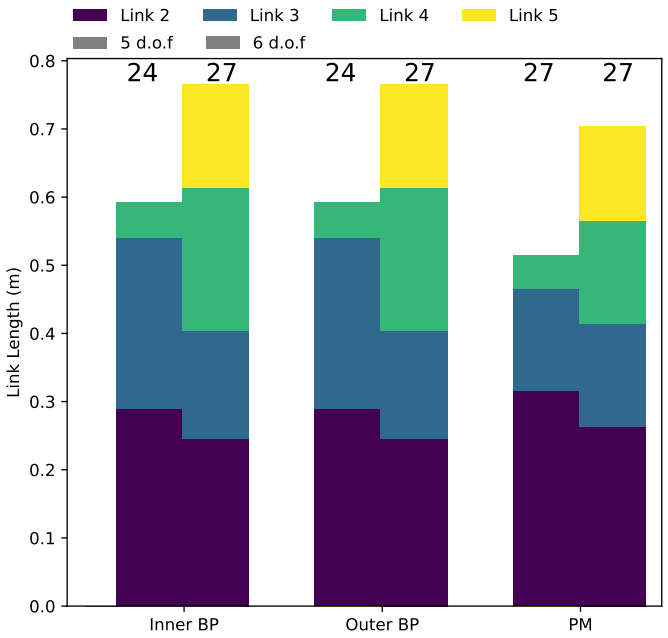


Figure 4.8: **Optimal Space robot designs.** Schematic of the optimal space robot designs for each of the three missions required for the OOA of the large aperture telescope. The number above each schematic is the optimal form factor.

the selected reaction wheel was the HoneyBee robotics system and the selected linear propulsion system was the AFRL prop unit.

Figure 4.8 shows that for all missions the 6 d.o.f. manipulator is the longest and always requires the largest form factor — $27U$. The 5 d.o.f. arm requires a smaller form factor for the inner and outer BP assembly, where a $24U$ base is optimal. The larger form factor is optimal with the heavier payload (PM module) and with the manipulators with higher d.o.f. This is because both the extra joints and the larger payload increase the dynamic coupling effects. As seen in Figure 4.6, the effect of this increase is minimised by having a larger base. However, as explained previously, increasing the size of the base increases the required control forces, even though the effect of the dynamic coupling is reduced. Table 4.8 shows that the power required for control is in fact, higher with the larger spacecraft for the same mission and d.o.f. system. Table 4.8 shows drastically higher power requirements compared to what is available from any of the proposed form factors. Although this was set as a condition for base optimisation, it was found that for these missions, none of the form factors could provide sufficient power.

Table 4.8: **Power requirements for sub-missionss.** The power required for each task within the sub-mission for the optimal design of all systems.

	Sub-task	Relocate		Grasp		Relocate		Fine Manipulation	
		Power (W)	Fuel (kg)	Power (W)	Fuel (kg)	Power (W)	Fuel (kg)	Power (W)	Fuel (kg)
Inner BP	*Payload*	*None*		*None*		*BP*		*BP*	
	5 d.o.f.	297	**5.9**	94	0.3	297	**4.2**	73	0.2
	6 d.o.f.	**441**	2.1	**26418**	**16.5**	**441**	2.4	**309650**	**10.63**
Outer BP	*Payload*	*None*		*None*		*BP*		*BP*	
	5 d.o.f.	297	**5.9**	94	0.3	297	**4.2**	73	0.2
	6 d.o.f.	**441**	2.1	**26418**	**16.5**	**441**	2.4	**309650**	**10.63**
PM	*Payload*	*None*		*None*		*PM*		*PM*	
	5 d.o.f.	297	**6.3**	89	0.2	297	**4.5**	272	0.18
	6 d.o.f.	**441**	2.5	**33034**	**11.9**	**442**	1.83	**755890**	**10.59**

As such this condition was removed and the form factor was optimised independently of this. Instead, a battery will be used to supplement power generation during these missions. Even with this modification, it can be seen that for all missions with the 6 d.o.f. manipulator, the power requirements are orders of magnitude larger than with the 5 d.o.f. arm. This is due to the control system trying to counteract instabilities caused by the increase in complexity due the extra joint and longer reach. It is unclear at this stage if this could be mitigated through controller refinement, such as that in Section 5. In contrast to this, it can be seen that the fuel requirements are, on average, higher across the missions with the lower d.o.f. arm. This is not an issue since the selected linear thruster has sufficient capability for the mission.

Across both the 5 and 6 d.o.f. arms, for the same d.o.f., each manipulator is shorter when a heavier payload is used *i.e*. the PM module. This is because with a shorter arm, the overall inertia when transporting a payload is lower, which reduces the effect of the dynamic coupling — one driver of the cost function. For all missions, with this controller the optimal design is the lower d.o.f. system — 11 d.o.f. There exists two optimal systems, one when the BP module is in

question and one when the PM module is. These designs were selected based on the drastically lower costs across all missions.

4.5 Conclusion

This chapter presented Ta-DAH Design, an automated design process tailored to optimise the hardware of free-flying space robots. Designing a space robot is a challenging problem with many dimensions not present in terrestrial robotic design. However, the suitable design of such a system will have huge benefits in the industry via the automation of previously unattainable OOO, including OOA, ADR and other servicing missions.

The design process presented in this work uses a weighted cost function to optimise the link lengths of a robotic manipulator and then iterates over pre-defined form factors to optimise the size of the base spacecraft. The objective functions used to size the manipulator quantify a number of important performance parameters and are all defined and expanded in this work. The base spacecraft is sized by ensuring that the on-board subsystems can perform the required maneuvers for the given tasks. Ta-DAH Design successfully optimised different space robots for use with the large space telescope design outlined in Appendix A. It was determined that a 11 d.o.f. system $(6$ d.o.f. base $+ 5$ d.o.f. manipulator$)$ should be used for all missions where the base sizing and link lengths vary based on the task at hand. This system is optimal for the controller and tasks defined in this section.

While the technique presented in this work hosts a number of benefits, mainly streamlining the design process of a complex robotic system and showing that hardware optimisation can increase overall performance, some improvements could be made. The system requires in-depth knowledge in relation to the mission scenario. This is because a top level mission, such as grasping a BP module, must be broken down into parameters accepted by the optimisation pipeline. In addition to this, all designs are evaluated using a PID controller. The 12 d.o.f. system was discounted in part due to the excessive power draw and instabilities. However, it is impossible to determine if this is due to a sub-optimal controller or poor physical design. While control parameters could be re-tuned for each design and task pair, this is a time-consuming process with no easy solution. Instead, this problem motivates the control approach in the next chapter, where RL is used to co-optimise the control and hardware of the system simultaneously.

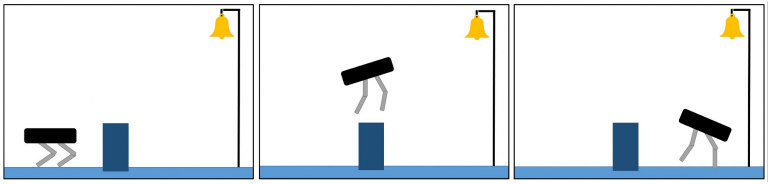
In addition to this, the disjointed manner in which Ta-DAH Design approaches the optimistion of the base and manipulator is likely to impact the quality of the final design. This again provides motivation for holistic design approach presented the next chapter.

Chapter 5

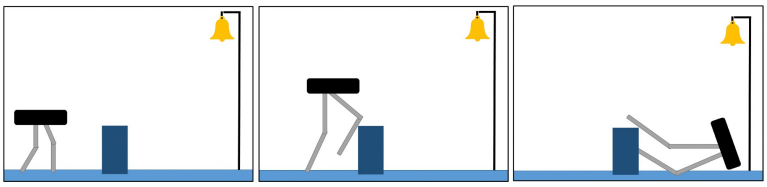
Co-Optimisation of Hardware and Software using Reinforcement Learning

The space robot designed in the previous chapter highlighted the implications of not optimising a control scheme for each specific hardware design. As discussed in Section 2, this decoupled approach to designing robotic systems is standard in the majority of design techniques. This leads to suboptimal results as the successful performance of any agent in a given environment is dependent on both its physical design and control capabilities. In some cases a poor mechanical design may result in an impossible control problem, such as a walker with legs too short to reach a target, see Figure 5.1a. In this case, no amount of training can overcome a fundamentally flawed hardware design. In less extreme cases, designs may be inefficient, with excess material or joints, leading to high energy consumption and/or poor movement patterns. In the opposite scenario, optimising only the hardware of a robotic agent with a simplistic control algorithm is unlikely to lead to optimal performance, as shown in Figure 5.1b. It is this aspect that impacted the final designs using the Ta-DAH Design process.

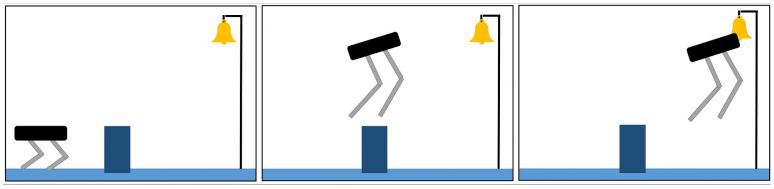
This chapter addresses this design disconnect. It presents Optimisation of Robotic Control and Hardware in Design (ORCHID) — an RL pipeline that simultaneously co-optimises the hardware and software of a robotic agent. The result is an efficient, versatile approach to holistic robot design, that brings the final system nearer to optimality.



(a) Agent changing its control policy to try and complete a task.



(b) Agent changes its morphology, rendering the control policy useless.



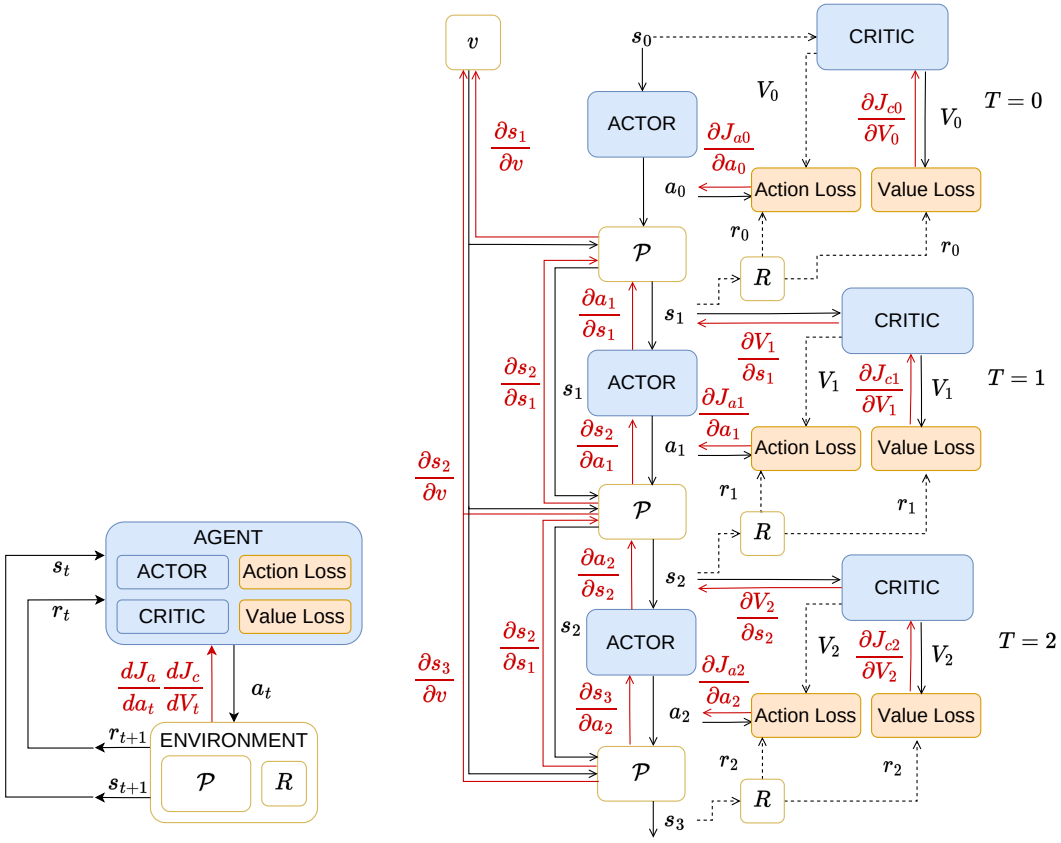
(c) Agent is able to complete the task by changing both its morphology and control policy.

Figure 5.1: **Different options for optimising control and morphology.**

This chapter first outlines the methodology of ORCHID. Before giving the results from a number of experiments demonstrating the advantages of this pipeline. A detailed look at the design of the space robot for OOA is also given before the chapter is concluded.

5.1 Methodology

ORCHID is a unified approach to hardware and software co-optimisation, where a single RL pipeline optimises both the morphology and control policy of a robot simultaneously. This method takes inspiration from the concept that fundamentally underpins all forms of deep learning – hierarchical partial differentiation. The approach works by forming a complex differential path through a trajectory rollout, meaning a vast amount of information can be leveraged. In typical RL pipelines this information is lost in the 'black-box' environment.



(a) Standard RL training loop

(b) ORCHID training loop

Figure 5.2: **Comparison of ORCHID and standard RL gradient flow.** In both images the forward pass is shown in black and the backward pass in red. Non-differentiable paths are shown with a dotted line.

Figure 5.2a shows a standard RL feedback loop for training of an actor-critic architecture based on experiences in an environment for all time steps in T. Visible in red is the simple differential path required for network updates to improve agent performance. This is dependent only on the action taken $(\mathbf{a}_t)$ at time t and the subsequent reward (r_t), leaving the agent with little information to exploit during updates.

Figure 5.2b shows three time steps in an environment following ORCHID. The use of a differentiable transition function $(\mathcal{P})$ hugely expands the capability of the system, via the introduction of a complex differential path throughout the rollout. From this, it becomes apparent how information on the impact/quality of a change to the morphology parameters will

impact performance — these parameters are defined as $\mathbf{v}$, as seen in Section 4. Exploiting this information makes it possible to quantify the coupled relationship between physical design and control of a robot-agent pair. During training updates are made to $\mathbf{v}$ and/or the agent parameters in a way that best improves overall performance. The exact robotic morphology parameterisation vector is environment specific and is sized by the number of designable parameters, *i.e.* if the mass and width of each leg on a quadrouped robot is to be optimised: $\mathbf{v} \in \mathbb{R}^{8\times1}$.

5.1.1 Reinforcement Learning Formulation

Deep RL is a branch of AI that utilises neural networks to understand and exploit unstructured environments. A neural network acts as a control policy that dictates what actions a robot should take in the environment. The network learns to map input states to optimal actions. In this work, agent is used to describe the full RL system. For an actor-critic method this consists of a policy and an additional network named the critic which is discussed later. Throughout training the policy and critic are optimised using experiences collected by the agent in the environment.

The agent takes $\mathbf{a}_t$, based on the policy and state $(\mathbf{s}_t)$, for each t. This changes the state of the environment to $\mathbf{s}_{t+1}$, the state includes the agent's physical location. r_t is received at each time step based on how good the action was from that state. These experiences are processed to determine a loss which is used to update the policy via back propagation. The policy aims to maximise the reward over a sequence of steps.

The standard convention for solving an RL problem in a fully observable environment is to formulate it as an Markov Decision Process (MDP) — $\mathcal{M}$. The Markov property states that the agents next state $\left(\mathbf{s}_{t+1}\right)$ is dependent only on its current immediate state and not everything that came before it, such that a state is only Markov if,

$$\mathbb{P}\left[\mathbf{s}_{t+1}|\mathbf{s}_t\right] = \mathbb{P}\left[\mathbf{s}_{t+1}|\mathbf{s}_1, ..., \mathbf{s}_t\right]. \tag{5.1}$$

An MDP is then an environment where every state displays the Markov property and state transitions can occur either randomly or via an agent taking an action. This means that in every state there is some probability that the state will change without the agent taking an action. An MDP is represented by the tuple: $\mathcal{M}$=$(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where $\mathbf{s} \in \mathcal{S}$ is any state in the state space and $\mathbf{a} \in \mathcal{A}$ is an action in the continuous action space. $\mathcal{P}\left(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t\right)$ is the state transition function that represents the dynamics of the environment, dictating how each state changes

to the next. γ is the reward discount rate. This is used to give more distant rewards a lower weighting than immediate rewards.

ORCHID uses the PPO algorithm which is an actor-critic method. This approach uses two networks during training − an actor network and a critic network. The actor network is the control policy, parameterised by $\tilde{\theta}$ which is defined as a number of weights and biases. The actor maps s_t at time t to a_t, denoted as: $\pi_{\tilde{\theta}}(\mathbf{a}_t|\mathbf{s}_t)$. The optimal policy, given by the optimal values of $\tilde{\theta} = \tilde{\theta}^*$, will maximise the expected return (R), over experiences induced by the control policy. Note that optimality cannot be proven in the case of RL and here optimal policy is used to mean better than any baseline. Where,

$$R\left(\mathbf{a}_T, \mathbf{s}_T\right) = \sum_{t=0}^{T} \gamma^t r\left(\mathbf{a}_t, \mathbf{s}_t\right), \tag{5.2}$$

Here $r(\mathbf{a}_t, \mathbf{s}_t)$ is the expected reward of taking $\mathbf{a}_t$ from state $\mathbf{s}_t$. This is approximated by the Q-value $(Q\left(\mathbf{a}_t, \mathbf{s}_t\right))$, where Q is calculated as,

$$Q\left(\mathbf{a}_T, \mathbf{s}_T\right) = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}}}\left[\sum_{t=0}^{T} \gamma^t r\left(\mathbf{a}_t, \mathbf{s}_t\right)\right]. \tag{5.3}$$

The optimal values of $\tilde{\theta} = \tilde{\theta}^*$ are estimated by solving,

$$\tilde{\theta}^* = \arg\max_{\tilde{\theta}} \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}}} R\left(\mathbf{a}, \mathbf{s}\right). \tag{5.4}$$

Solving this equation can be done in a number of different ways, depending on the method used to sample **s** and **a**. With PPO, experiences are collected in the environment and these are used to calculate an average R. This is used to calculate a loss function and then back propagation is used to optimse $\tilde{\theta}$.

The role of the critic is to learn the value of each state. The critic is also an NN and is parameterised by $\tilde{\omega}$. This network takes input $\mathbf{s}_t$ and outputs the quality/value $(V\left(\mathbf{s}\right))$ of that state: $\pi_{\tilde{\omega}}(V_t|\mathbf{s}_t)$. The optimal values of $\tilde{\omega} = \tilde{\omega}^*$ are determined by solving:

$$\tilde{\omega}^* = \arg\max_{\tilde{\omega}} \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}},V\sim\pi_{\tilde{\omega}}} \left(R\left(\mathbf{a}, \mathbf{s}\right) - V\left(\mathbf{s}\right)\right)^2. \tag{5.5}$$

The learned value of each state is used to stabilise learning.

5.1.2 Parametric Control Optimisation

Although implemented with PPO in this work, ORCHID is agnostic to the control policy architecture — given it is differentiable and can be parameterised. To demonstrate this, experiments are carried out with two different control policies. The first is a classical PID controller, where the policy is defined as:

$$\mathbf{a}_t = k_P \mathbf{e}_t + k_I \int \mathbf{e}_t + k_D \frac{d\mathbf{e}_t}{dt}, \tag{5.6}$$

and parameterised by (k_I, k_D, k_P), where $\mathbf{e}_t$ is the positional error *i.e.* the difference between $\mathbf{s}_t$ and a reference vector. This tuple acts as the control policy and a neural network is not used in this setting, instead k_I, k_D, k_P are optimised.

The second controller is a deep multi-layer-perception control network parameterized by $\tilde{\theta}$. All networks used in this chapter have 3 hidden layers, each with 64 nodes. The input and output dimensions of each are dependent on the environment and use case of the network. The input dimension of the actor is governed by the dimension of the state space, and the output by the dimensions of the action space. Contrary to this, the input dimension of the critic is the same as the actor, while the output will always be 1. The outputs of the actor network are stochastic with the network predicting a mean (μ) and standard deviation (σ) for each action, this distribution is sampled from at run time. The Tanh activation function is used on all but the output layer to ensure non-linearities between layers. Outputs corresponding to σ are clamped to a minimum of 0.001.

5.1.3 Losses

Throughout training, different losses are used to optimise the various trainable parameters. These losses quantify the performance of the system.

PPO

The PPO algorithm works by ensuring that policy updates are stable and do not cause the new policy to move too far from the old policy. It does this by utilising the advantage function (A),

$$A_t\left(\mathbf{a}_t, \mathbf{s}_t\right) = Q\left(\mathbf{a}_t, \mathbf{s}_t\right) - V\left(\mathbf{s}_t\right), \tag{5.7}$$

and hyperparameter ϵ, which clips the size of an allowable update. The use of the advantage during actor updates reduces variance and acts to normalise batch data during training in a way that stabilises learning and prevents exploding gradients. The loss function used to constrain $\tilde{\theta}$ is therefore,

$$\mathcal{J}_a\left(\tilde{\theta}\right) = -\mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}}}\left[\frac{\pi_{\tilde{\theta}t}(\mathbf{a}|\mathbf{s})}{\pi_{\tilde{\theta}t+1}(\mathbf{a}|\mathbf{s})}A\left(\mathbf{a},\mathbf{s}\right), clip\left(\frac{\pi_{\tilde{\theta}t}(\mathbf{a}|\mathbf{s})}{\pi_{\tilde{\theta}t+1}(\mathbf{a}|\mathbf{s})}, 1-\epsilon, 1+\epsilon\right)A\left(\mathbf{a},\mathbf{s}\right)\right]. \tag{5.8}$$

The critic network is updated using the Mean Squared Error (MSE) between its output for a given state ($V(\mathbf{s})$) and the expected return for that same state. The critic loss ($\mathcal{J}_c$) is defined as,

$$\mathcal{J}_c\left(\tilde{\omega}\right) = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}}}\left(\left[R\left(\mathbf{a},\mathbf{s}\right) - \pi_{\tilde{\omega}}\left(\mathbf{s}\right)\right]^2\right). \tag{5.9}$$

This network therefore learns to output the same expected return as gained from actual experiences in the environment.

PID

The values for k_I, k_D and k_P are sampled from a distribution $\left(k_I \sim \mathcal{N}(k_{I\mu}, k_{I\sigma})\right)$ at the start of each episode. A full episode is run with the sampled values and the final reward is calculated. The loss function for the PID controller ($\mathcal{J}_{PID}$) is then a function of the probability of the selected constants given their distribution, and the resulting episode reward,

$$\mathcal{J}_{PID}(K) = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim K} A\left(\mathbf{a},\mathbf{s}\right), \tag{5.10}$$

where K = [$k_{I\mu}$, $k_{I\sigma}$, $k_{D\mu}$, $k_{D\sigma}$, $k_{P\mu}$, $k_{P\sigma}$] which are the parameters to be optimised. With this controller a critic network is still required. This uses the loss defined in Equation 5.9 with the actions drawn from K as oppose to $\pi_{\tilde{\theta}}$.

Design Parameter

ORCHID aims to modify the design parameter such that if the policy were to take the same action from the same state the resultant state with the new embodiment will give a higher reward than pre-modification. The quality of a state is quantified via $V(\mathbf{s})$. The design parameter loss ($\mathcal{J}_{\mathbf{v}}$) is therefore an average of the sum of the state value over an episode,

$$\mathcal{J}_{\mathbf{v}}\left(\mathbf{v}\right) = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P}}\left(\frac{\sum_{t=0}^{T} V\left(\mathbf{s}_t\right)}{T}\right). \tag{5.11}$$

Since $V(\mathbf{s})$ is a learned parameter, a 'direct' loss ($\mathcal{J}_{\mathbf{v}d}$) is also introduced that captures the true reward of an episode. This ensures that a delay in learning by the critic network does not hinder optimisation of the design parameter. As with $\mathcal{J}_{PID}$ the direct loss utilises the fact that the design parameters are sampled from a distribution where,

$$\mathcal{J}_{\mathbf{v}d}(\mathbf{v}) = \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\theta}}} \left(\text{logprob}(\mathbf{v})R(\mathbf{a},\mathbf{s})\right). \tag{5.12}$$

Note that this equation is for the PPO controller. For the PID controller actions are instead drawn from K. Since a distribution over designs is used during optimisation, the mean is taken as the final optimal design.

5.1.4 Parameter Co-Optimisation

Optimising parameters $\tilde{\theta}$, $\mathbf{v}$ and $\tilde{\omega}$ is nontrivial. Both are highly nonlinear with numerous complex inter-related functions. To deal with this, the entire system is formulated as a chain of partial derivatives — excluding the reward function. Different compositions of these partial derivatives allow the system to calculate the direction in which each parameter should be updated. Updates are done in a way to minimise or maximise the respective losses defined in Section 5.1.3.

The introduction of $\mathbf{v}$ into the optimisation results in a number of new partial derivative terms, compared to the standard formulation: $\frac{\partial \mathbf{s}}{\partial \mathbf{v}}$, $\frac{\partial \mathbf{s}}{\partial \mathbf{a}}$, $\frac{\partial V}{\partial \mathbf{s}}$ and $\frac{\partial \mathbf{s}_t}{\partial \mathbf{s}_{t-1}}$. $\frac{\partial \mathbf{s}}{\partial \mathbf{a}}$ and $\frac{\partial V}{\partial \mathbf{s}}$ can be solved explicitly from the forward pass, while $\frac{\partial \mathbf{s}}{\partial \mathbf{v}}$ and $\frac{\partial \mathbf{s}_t}{\partial \mathbf{s}_{t-1}}$ require the use of a differentiable transition function. In order to implement this, the environment is split into two components — $\mathcal{P}$ and a reward function $\mathcal{R}$, shown in Figure 5.2. This modification does not affect the optimisation of $\tilde{\theta}$ or $\tilde{\omega}$ as the partial derivative paths remain the same:

$$\frac{\partial \mathcal{J}_a}{\partial \tilde{\theta}} = \frac{\partial \mathcal{J}_a}{\partial \mathbf{a}} \frac{\partial \mathbf{a}}{\partial \tilde{\theta}}, \tag{5.13}$$

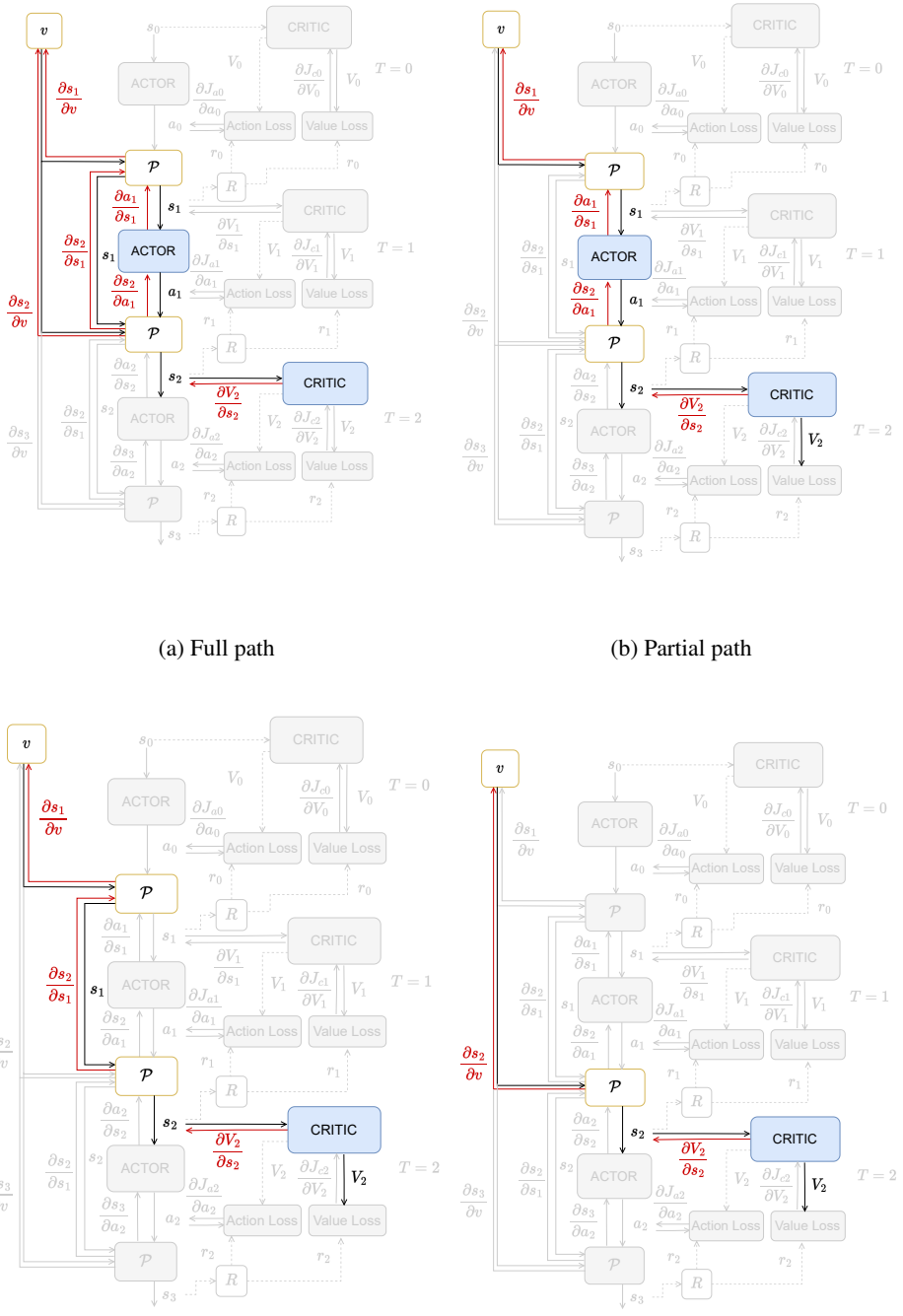
$$\frac{\partial \mathcal{J}_c}{\partial \tilde{\omega}} = \frac{\partial \mathcal{J}_c}{\partial V} \frac{\partial V}{\partial \tilde{\omega}}, \tag{5.14}$$

and for k_I, k_P and k_D,

$$\frac{\partial \mathcal{J}_{PID}}{\partial (k_I, k_P, k_D)} = \frac{\partial \mathcal{J}_{PID}}{\partial \mathbf{a}} \frac{\partial \mathbf{a}}{\partial (k_I, k_P, k_D)}. \tag{5.15}$$

A similar path can be formed to update the design vector using both losses for an isolated time step,

$$\frac{\partial \mathcal{J}_{\mathbf{v}}}{\partial \mathbf{v}} + \frac{\partial \mathcal{J}_{\mathbf{v}d}}{\partial \mathbf{v}} = \frac{\partial \mathcal{J}_{\mathbf{v}}}{\partial \mathbf{s}} \frac{\partial \mathbf{s}}{\partial \mathbf{v}} + \frac{\partial \mathcal{J}_{\mathbf{v}d}}{\partial \mathbf{v}}. \tag{5.16}$$



(a) Full path

(b) Partial path

(c) Partial path

(d) Partial path

Figure 5.3: **Partial derivatives for v when $t = 2$.**

However, $\forall t \geq 2$, there exists a complicated and unique partial derivative sum. Taking $t = 2$ as

an example, there are now 3 different information flows, as illustrated in Figure 5.3. The full chain of derivatives is therefore,

$$\frac{\partial \mathcal{J}_{\mathbf{v}2}}{\partial \mathbf{v}} + \frac{\partial \mathcal{J}_{\mathbf{v}d2}}{\partial \mathbf{v}} = \frac{\partial V_2}{\partial \mathbf{s}} \left(\frac{\partial \mathbf{s}_2}{\partial \mathbf{v}} + \frac{\partial \mathbf{s}_2}{\partial \mathbf{a}} \frac{\partial \mathbf{a}_1}{\partial \mathbf{s}} \frac{\partial \mathbf{s}_2}{\partial \mathbf{v}} + \frac{\partial \mathbf{s}_2}{\partial \mathbf{s}} \frac{\partial \mathbf{s}_1}{\partial \mathbf{v}} \right) + \frac{\partial \mathcal{J}_{\mathbf{v}2}}{\partial \mathbf{v}} \tag{5.17}$$

Indeed the number of routes for gradient flow grows exponentially with the length of the rollout. This creates huge computational demand and drastically slows down the training of the system. Instead, it is possible to limit the gradient flow between time steps. For example applying Equation 5.16 independently at all timesteps would ensure that information flows across a single timestep. Meanwhile applying Equation 5.17 independently at every timestep would limit the system to 2 steps of information flow. The use of different derivatives is explored.

5.1.5 Differentiable State Transitions

ORCHID requires that the environment must have a differentiable transition function as detailed above. Although this could theoretically preclude certain environments, the advancement of differentiable simulators means behaviors such as collisions, grasping and soft deformable structures can now be modelled using fully differentiable functions [60, 84]. All environments in this thesis have been modified to have a differentiable $\mathcal{P}$ following the method outlined in [40]. ORCHID does not require the reward function be differentiable and as such this is not modified. Usually $\mathcal{P}$ is defined as:

$$\mathcal{P}\left(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t\right), \tag{5.18}$$

however, after modification, it becomes a function of $\mathbf{v}$,

$$\mathcal{P}\left(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t, \mathbf{v}\right). \tag{5.19}$$

This is captured in the definition of the environments used to evaluate ORCHID in this work.

5.1.6 Implementation

During robotic hardware design it is normal to have bounds on the allowable values of parameters. These may be due to physical limitations, such as non-negative weights and lengths or known environmental stipulations such as storage or operating facility sizes. ORCHID implements

such constraints by passing parameters through a smooth differentiable bounding function such as CELU [12],

$$CELU(v,\alpha) = \begin{cases} v, & \text{if } v \geq 1 \\ \alpha\ exp\left(\frac{v}{\alpha}\right) - 1, & \text{otherwise} \end{cases}. \tag{5.20}$$

Depending on which controller is required a different combination of losses are used to update the control parameters and design vector. Algorithm 1 shows the training loop when the PPO controller is used, where $\mathcal{L}_a$, $\mathcal{L}_c$ and $\mathcal{L}_v$ are the learning rates for the optimisation of $\tilde{\theta}$, $\tilde{\omega}$ and $\mathbf{v}$ respectively. In this implementation the following losses are used: $\mathcal{J}_{\mathbf{v}}$, $\mathcal{J}_{\mathbf{v}d}$, $\mathcal{J}_c$ and $\mathcal{J}_a$. In this set-up, o denotes the total number of updates to the system, p the number of steps to carry out per update, q the number of iterations over one batch of data. The same values were used for all experiments in this chapter, with $o = 4000$, $p = 2500$ and $q = 4$. Two different learning rates were used where $\mathcal{L}_a$ $\mathcal{L}_c = 7e-4$ and $\mathcal{L}_v$= $7e-2$.

Algorithm 2 shows the training loop when the PID controller is used, where $\mathcal{L}_{PID}$ is the learning rate of the PID parameters which is also $7e-4$. The required losses in this implementation are $\mathcal{J}_{PID}$, $\mathcal{J}_c$, $\mathcal{J}_{\mathbf{v}}$ and $\mathcal{J}_{\mathbf{v}d}$.

5.2 Experiments and Results

The generality of ORCHID is demonstrated by evaluating performance across four control problems. Three of these are OpenAI gym environments [23]. Unless stated otherwise motor specifications and material choice/density are kept the same as the default. In all cases, updates to the morphology maintained the symmetry of parts and the overall configuration of the robots. CartPole and Pendulum are demonstrated with both the PPO and PID controllers. The other two environments are not, since this would require a path planning component.

5.2.1 Environments

Cart Pole

The task in this environment involves balancing a pole above a cart attached to an un-actuated joint where motion occurs along a friction less track. A reward of +1 is given for each time step

Randomly initialise parameters $\tilde{\theta}$, $\mathbf{v}$ and $\tilde{\omega}$
for $o = 1, 2, \ldots$ total number of updates **do**
    Initialise memory $M \leftarrow \varnothing$
    Sample starting state: $\mathbf{s}_0^o \sim \mathcal{S}$
    Sample starting morphology: $\mathbf{v}_o$
    for $p = 1, 2, \ldots$ number of steps per update **do**
        Sample action: $\mathbf{a}_p \sim \pi_{\tilde{\theta}}(\mathbf{a}_p|\mathbf{s}_p)$
        Determine next state: $\mathbf{s}_{p+1} = \mathcal{P}(s_{p+1}|a_p, s_p, v_o)$
        Determine reward: $r_p = \mathcal{R}(\mathbf{s}_{p+1}, \mathbf{a}_p, \mathbf{v}_o))$
        Store transition: $M \leftarrow M \cup (s_p, a_p, r_p, v_o, s_{p+1})$
    end for
    Calculate $\mathcal{J}_{\mathbf{v}}$
    Calculate $\mathcal{J}_{\mathbf{v}d}$
    $\mathbf{v} \leftarrow \mathbf{v} + \mathcal{L}_v \nabla (\mathcal{J}_{\mathbf{v}} + \mathcal{J}_{\mathbf{v}d})$
    for $q = 1, 2, \ldots$ number of PPO epochs **do**
        Sample mini-batch from M
        Calculate $\mathcal{J}_a$
        Calculate $\mathcal{J}_c$
        $\tilde{\theta} \leftarrow \tilde{\theta} + \mathcal{L}_a \nabla \mathcal{J}_a$
        $\tilde{\omega} \leftarrow \tilde{\omega} + \mathcal{L}_c \nabla \mathcal{J}_c$
    end for
end for
return $\tilde{\theta}$, $\mathbf{v}$ and $\tilde{\omega}$

Algorithm 1: Implementation of ORCHID with PPO controller

that the pole remains above the cart ($\pm 15^\circ$). In this task setting the mass of the cart and length of the pole make up $\mathbf{v}$. The maximum number of timesteps that the pole can remain upright is 1000.

Pendulum

In this environment the goal is to swing the pendulum so that it remains in an upright position. A reward is given based on the angular position and velocity of the pendulum, and the force applied at each step. A less negative reward is given for a stationary upright pendulum achieved using minimal force. As such, a reward of zero is optimal, although not actually possible. In this setting the mass and length of the pole are optimised independently. This replicates a change in both the material and size of the pole.

Randomly initialise parameters in K, $\mathbf{v}$ and $\tilde{\omega}$
for $o = 1, 2, \ldots$ total number of updates **do**
    Initialise memory $M \leftarrow \varnothing$
    Sample starting state: $\mathbf{s}_0^o \sim \mathcal{S}$
    Sample starting morphology: $\mathbf{v}_o$
    Sample: k_I, k_D and k_P
    for $p = 1, 2, \ldots$ number of steps per update **do**
        Calculate action using Equation 5.6
        Determine next state: $\mathbf{s}_{p+1} = \mathcal{P}(s_{p+1}|a_p, s_p, v_o)$
        Determine reward: $r_p = \mathcal{R}(s_{j+1}, a_p, v_o))$
        Store transition: $M \leftarrow M \cup (s_p, a_p, r_p, v_o, s_{p+1})$
        if episode end **then**
            Sample k_I, k_D and k_P
        end if
    end for
    Calculate $\mathcal{J}_{\mathbf{v}}$
    Calculate $\mathcal{J}_{\mathbf{v}d}$
    Calculate $\mathcal{J}_{PID}$
    Calculate $\mathcal{J}_c$
    $\mathbf{v} \leftarrow \mathbf{v} + \mathcal{L}_v \nabla \left(\mathcal{J}_{\mathbf{v}} + \mathcal{J}_{\mathbf{v}d}\right)$
    $K \leftarrow K + \mathcal{L}_{PID} \nabla \mathcal{J}_{PID}$
    $\tilde{\omega} \leftarrow \tilde{\omega} + \mathcal{L}_c \nabla \mathcal{J}_c$
end for
return K, k_I, k_P, k_D, $\mathbf{v}$ and $\tilde{\omega}$

Algorithm 2: Implementation of ORCHID with PID controller

Mountain Car

This environment requires a cart which starts at the bottom of two ‘mountains’, to drive up the right hand side to reach the goal. The engine is not strong enough to do so immediately, so instead the cart must build up momentum by cycling between the mountains. A negative reward is given for all engine output and a sparse reward of 0 when the goal is reached. In this setting only the mass of the car is optimised.

Space Robot

This experiment uses the python implementation of the space robot simulator, defined in Section 3.2.4. The link lengths form factor and base spacecraft mass of the 11 d.o.f. system are optimised using the ORCHID pipeline, meaning $\mathbf{v}$ is defined as,

$$\mathbf{v} = [l_2, l_3, l_4, d_x, d_y, d_z, m_{SC}] \,. \tag{5.21}$$

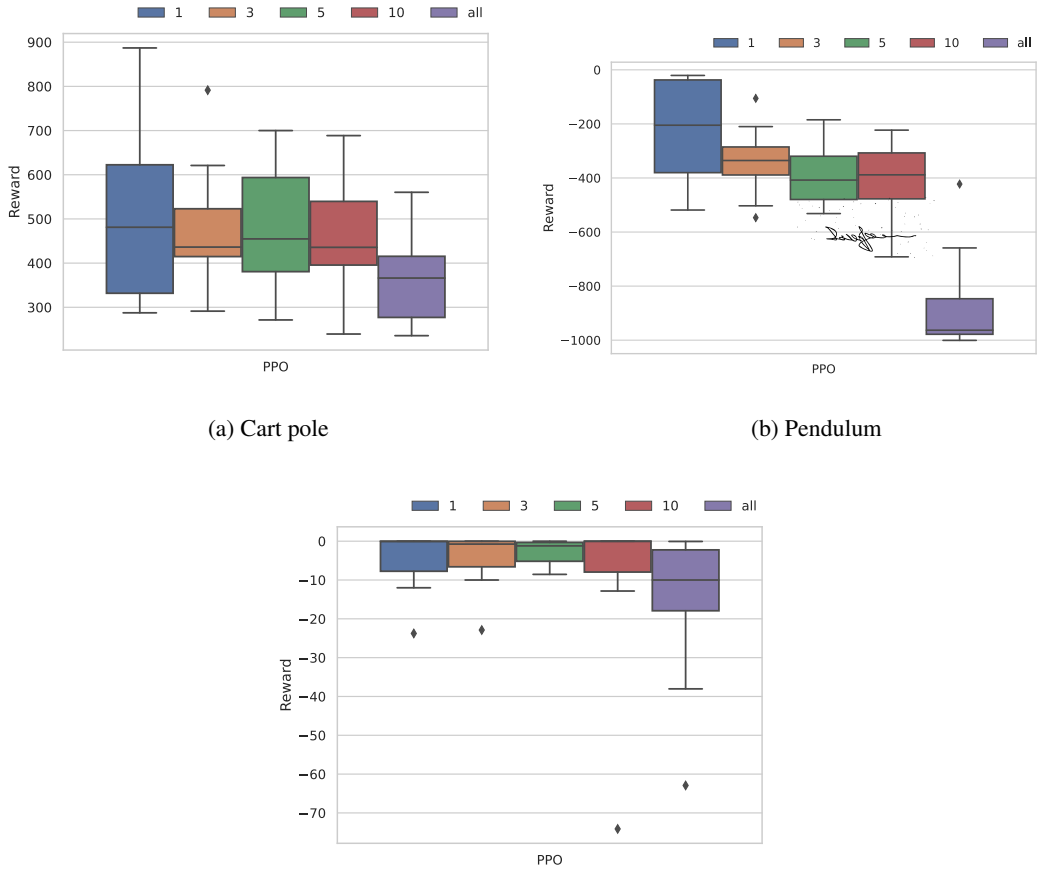
For a single instance of the space robot to complete all the sub-missions in a mission, it is necessary to implement a form of hard gating, inspired by Progressive NNs [135]. A separate actor and critic network are implemented for each of the sub-missions, where each is only trained on its corresponding task. The system is able to select the relevant network for the current sub-mission. The optimisation of the morphology, via ORCHID, is done across all the sub-missions, meaning a single robot is optimised for mission, as in Section 4.

5.2.2 Baselines

The performance of ORCHID is compared against three different baselines. The same baselines are used for the PPO and PID control options. The first is a grid search, where each different design option is trained for 1×10^6 timesteps. In the second, the morphology is optimised using Covariance Matrix Adaptation with Evolutionary Selection (CMA-ES), where the control policy is optimised using RL for each design option [87]. This inner loop also runs for 1×10^6 time steps and 20 trials are used for morphology optimisation. The third uses Random Selection (RS) to optimise the morphology where again, the control policy is optimised using RL — 20 trials of 1×10^6 steps are used. Each experiment is run 10 times and all results are given. The designs used in the grid search are used as the starting points for the 10 runs with ORCHID. Morphology initialisation for CMA-ES and RS was left to the respective algorithms.

5.2.3 Gradient Flow Over Timesteps

As discussed in Section 5.1.4 it is possible to limit the gradient flow between time steps in the ORCHID pipeline. Using more timesteps leads to far greater memory usage and computational load, but allows the system to consider how it's changes will impact more distant future states. In order to determine the optimal number of timesteps over which information should flow



(a) Cart pole

(b) Pendulum

(c) Mountain car

Figure 5.4: **Comparison of timesteps in ORCHID.** Comparison of different numbers of timesteps over which information can flow for morphology updates in ORCHID, where 'all' corresponds to a full episode in the environment.

Table 5.1: **Comparison of run times for ORCHID over different timesteps.** Times quoted are an average of the 10 different runs carried out.

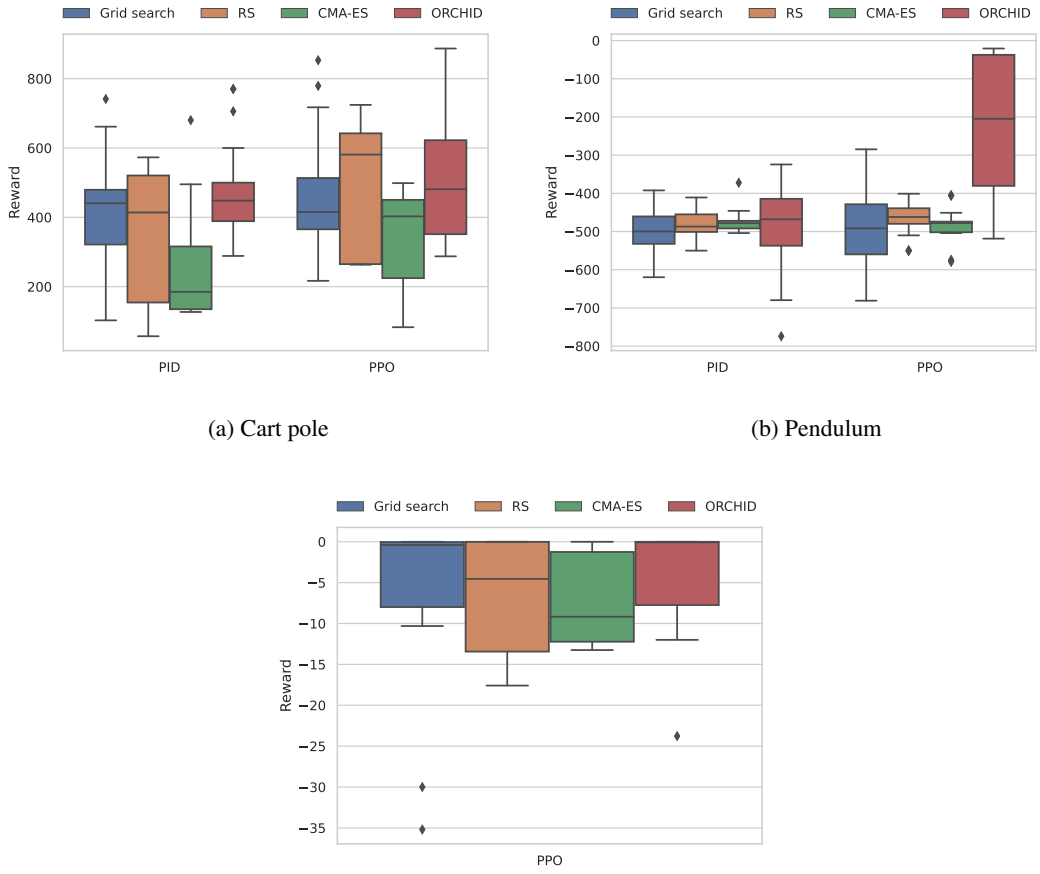
	Run time (hrs:mins)				
Time Steps	1	3	5	10	Full episode
Cart Pole	**5:06**	*5:27*	11:18	16:59	25:52
Pendulum	*6:48*	**5:07**	16:16	15:48	25:43
Mountain Car	*4:54*	**4:06**	11:22	12:25	21:51

for morphology updates, a number of different options were evaluated. The results are shown in Figure 5.4. From this it can be seen that performance is actually better when information propagates over fewer timesteps, with the worst performance occurring when gradients flow back over an entire episode. For both cart pole and mountain car, performance for all timesteps less than the full episode show fairly similar results, with a bigger variance appearing in the lower timestep runs in the pendulum environment. Following the derivative definition, it is known that there exists a single path for 1 timestep with a more complex path for 3, 5, 10 and the whole episode. Although stretching back over more steps, no additional derivative terms are introduced between 3, 5, 10 and the whole episode. It is therefore concluded that the value of the information must degrade over timesteps. This is potentially due to increased noise from more passes through the stochastic transition function, and weakened correlation with future events. Also of importance at this stage of evaluation is the time taken for each of these experiments, which is given in Table 5.1. This table shows that the run time for 1 or 3 steps is fairly consistent across all experiments, with a large jump up to those over 5 or 10 steps and a further increase for the full episode. Given the shorter run time and higher results, information flow over a single timestep is used in the rest of this chapter.

5.2.4 Performance Analysis

A comparison between the final performance of designs chosen by ORCHID and the three baselines for both the PID and PPO controller is given. Figure 5.5 shows a box and whisker plot for each of the experiments, excluding any involving the space robot which are discussed later.

The implementation of ORCHID shows the highest maximum performance across all experiments and control policies, excluding in the case of mountain car with PPO control, in which performance is comparable. This demonstrates the need for true design co-optimisation since the exceptional rewards cannot be reached when the morphology is optimised independent to the control, as with the grid search, CMA-ES and RS. The largest improvement was seen in the pendulum environment with PPO control. ORCHID optimised the pole to be very long and made of a light material, the exact parameters on which the length converged exceeded the allowable limits of all the baselines. This shows the ability of ORCHID to converge upon designs which a human designer may never have considered. The optimal pole length was over 45 units for $6/10$ of the ORCHID experiments, where the other baselines were capped at 40.



(a) Cart pole

(b) Pendulum

(c) Mountain car

Figure 5.5: **Overall performance of ORCHID compared to baselines.**

While these limits could have been increased this would have drastically increased an already long run time since the baseline methods evaluate designs across the defined space. Converse to this, ORCHID intelligently explores promising regions of a potentially infinite space. This experiment shows ORCHID's capability of discovering new morphologies that may not have been considered during initial design iterations.

In the case of mountain car with PPO control, the maximum reward of ORCHID is on a par with the other optimisation techniques. This is due to the baselines already achieving near maximum rewards, meaning that little improvement to robotic design was needed.

Smaller improvements are seen between ORCHID and the other baselines with the PID controller across all environments. In addition to this, overall performance is lower with the PID

Table 5.2: **Comparison of run times for ORCHID and all baselines.** Times quoted are an average of the 10 different runs carried out. The exception is the grid search for which a sum over the 10 runs is given. This is because each individual run holds little value independent to the others.

		Run time (hrs:mins)			
	Control	Grid Search	CMA-ES	RS	ORCHID
Cart Pole	PID	9:06	8:12	*8:03*	**4:59**
	PPO	10:48	9:26	*8:54*	**5:06**
Pendulum	PID	19:42	*15:43*	16:18	**6:31**
	PPO	20:24	*13:37*	15:43	**6:48**
Mountain Car	PPO	20:06	*12:55*	13:31	**4:54**

controller compared to with the use of a more complex policy. This is due to the limitations of using a simpler architecture to learn a complex policy. In all cases ORCHID + PID has the highest mean and maximum performance compared to all baselines, again highlighting the need to allow for hardware design optimisation.

5.2.5 Design Speed Analysis

Another advantage of ORCHID is the acceleration of the design cycle, and sample efficiency of training. This speed improvement manifests in a number of ways. Firstly only 1 run is required in ORCHID in order to optimise the full system, compared to the multiple trials needed for all the other baselines. This leads to an approximate factor of 3 increase in the runtime for these baselines, as shown in Table 5.2. On top of this, the implementation of all other baselines require maintaining the distribution of control policies and results over multiple runs, leading to high memory requirements.

ORCHID also converged in a similar time frame or slightly slower than just refining the control policy alone. This is surprising since this joint approach is strictly more complex and higher dimensional. It would therefore be natural to expect that more samples would be needed to account for the increased number of parameters. However, since the additional parameters are meaningful and directly impact performance, it did not cause a noticeable delay in convergence.

Table 5.3: **Comparison of final designs.** The standard deviation across all runs with each of the algorithms is quoted. The smallest is shown in bold with the next smallest in italics.

		PID			PPO		
	v	CMA-ES	RS	ORCHID	CMA-ES	RS	ORCHID
Cart Pole	Pole length	11.4	*8.7*	**2.0**	11.1	*9.6*	**4.3**
	Cart mass	9.7	*6.6*	**0.4**	*5.6*	8.7	**0.4**
Pendulum	Pole length	12.1	*11.6*	**2.8**	**5.9**	*12.4*	20.9
	Pole mass	11.1	*9.07*	**0.2**	*7.1*	8.9	**0.2**
Mountain Car	Car mass		n/a		*8.4*	9.5	**5.95**

5.2.6 Final Designs

The idea of ORCHID is to find optimal robot-agent pairs, meaning that the system cannot be expected to converge on a single design across all runs given variation in the optimal control parameters. However, looking at the final optimal designs, where the standard deviation across runs can be seen in Table 5.3, it can be seen that ORCHID outputs the most similar designs. Note that the starting points of the ORCHID optimisation varied, matching those of the grid search, while the starting points of CMA-ES and RS were determined by the relevant algorithms.

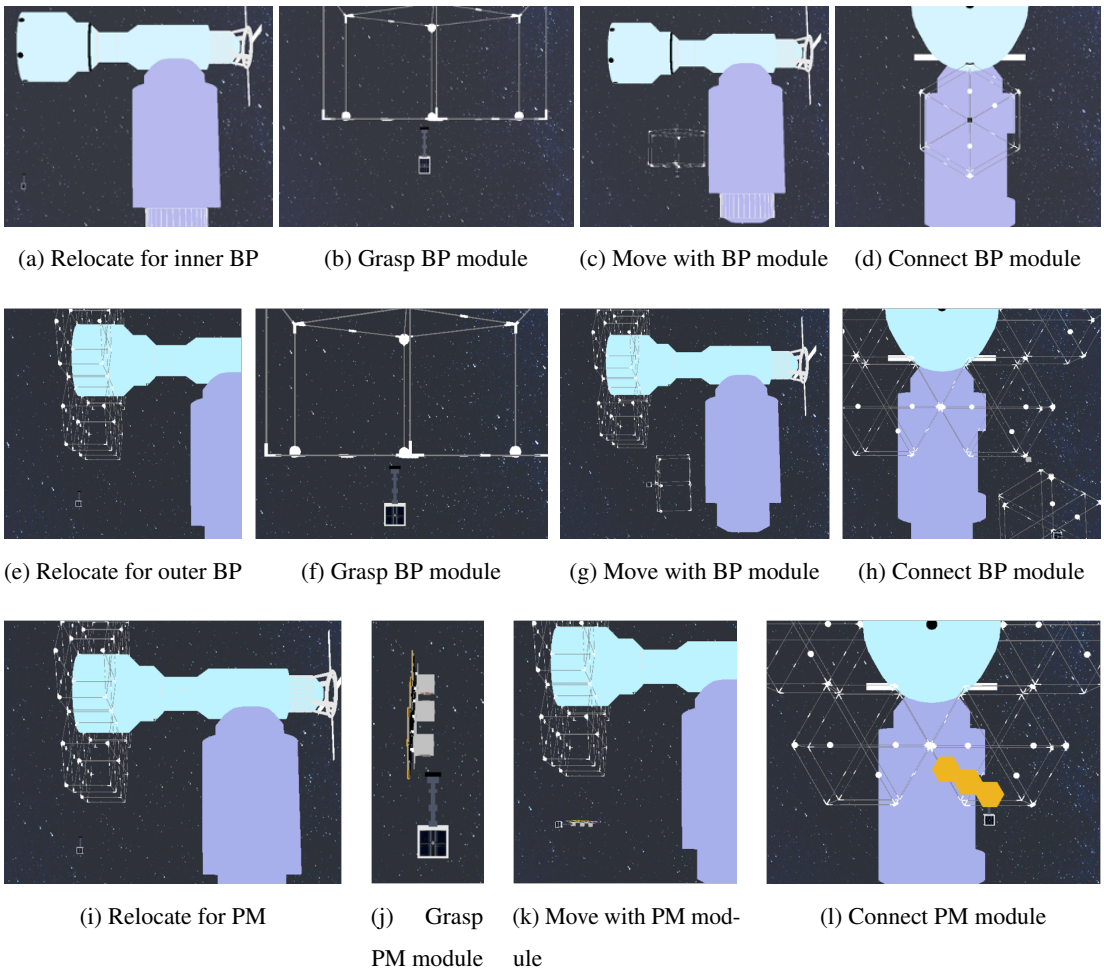
In all cases other than the pole length for the pendulum with PPO control, the ORCHID designed agents had the smallest standard deviation, where a clear difference was seen in the mass of the components. This shows how ORCHID converges on much more similar designs than the RS and CMA-ES. This is because the system leverages information from the environment, meaning that updates to the morphology are made based on system dynamics and are therefore somewhat grounded in physics. Conversely, the other baselines were only able to optimise the morphology by trial-and-error, with no knowledge of the relationship between the changes and the physics of the system. Some variation in the final designs is present due to the distribution maintained during training, in addition to noise throughout the training pipeline. The final design is taken as the mean since, in some cases the standard deviation does not converge to a low value.

Table 5.4: **Space robot performance.** Scores achieved by the Ta-DAH Design robot and the robot designed using ORCHID for each of the sub-missions.

	Sub-mission	Relocate		Grasp		Relocate		Fine Manipulation	
		Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.
Inner BP Assembly	*Payload*	*None*		*BP*		*BP*		*BP*	
	Ta-DAH Design	-425.1	-380.3	**922.2**	**1074.8**	-185.6	-146.8	-63.8	-40.1
	ORCHID	**-172.2**	**-166.7**	-30.6	-30.4	**-133.8**	**-129.5**	**-33.5**	**-31.4**
Outer BP Assembly	*Payload*	*None*		*BP*		*BP*		*BP*	
	Ta-DAH Design	**1185.0**	**1188.7**	905.7	1061.7	-95.8	-88.6	-86.2	-42.4
	ORCHID	985.2	1048.8	**1066.2**	**1079.5**	**883.8**	**1073.7**	**-37.8**	**-37.7**
PM Assembly	*Payload*	*None*		*PM*		*PM*		*PM*	
	Ta-DAH Design	-158.8	-144.1	**335.1**	**1056.1**	-1125.2	-351.5	-59.1	-57.6
	ORCHID	**985.8**	**1016.8**	-29.4	-2.2	**-90.6**	**252.7**	**-49.6**	**-37.2**

5.2.7 Space Robot Design

Following the improvements that ORCHID made compared to the baselines, the method was employed to design the space robot, discussed throughout this work. Results for the different missions are given in Table 5.4. As outlined by the reward scheme in Section 3.2.4 interest extends beyond just if the mission was completed or not, and into how efficiently this was done. As such, performance values are given — these are an average of 100 episodes. The baseline used is the optimally designed space robot from Section 4, trained with a PPO controller, and this is compared to the performance of the ORCHID designed system. While performance is not exceptional, ORCHID outperforms Ta-DAH Design in $9/12$ sub-missions. It is important to highlight that the RL controller component is run without any additional aids for improving performance. As such, this is considered as a high achievement given the complexity of the tasks and high dimensionality of the input and action space.



(a) Relocate for inner BP (b) Grasp BP module (c) Move with BP module (d) Connect BP module

(e) Relocate for outer BP (f) Grasp BP module (g) Move with BP module (h) Connect BP module

(i) Relocate for PM (j) Grasp PM module (k) Move with PM module (l) Connect PM module

Figure 5.6: **Render of simulator during training.** Capture of each sub-mission needed for the full assembly process.

Inner BP Assembly

The optimal robot design output by the ORCHID system for this task had design vector

$$\mathbf{v} = [0, 0, 0.5, 0.8, 0.8, 0.3, 97].$$

ORCHID drastically increases the mass and dimensions of the base spacecraft, pushing them beyond the standard form factors initially considered. In addition to this it reduced the apparent d.o.f. of the system to 2 by making all but one link length $0\,\mathrm{m}$.

The ORCHID optimised design performs better than the Ta-DAH Design on 3 out of 4 tasks, with the average score for the initial relocate task increasing by 59.49% and around 21.69%

for the fine manipulation task. While these increases are not due to sub-mission completion, they show that the re-sized agent is able to operate in a more controlled manner. The ORCHID agent did not learn to capture the BP module during the grasp task, explaining the low score. Figure 5.6a to Figure 5.6d show images from the simulator for the corresponding sub-missions.

Outer BP Assembly

The next set of sub-missions allows the robotic agent to assemble the outer ring of BP modules — shown in Figure 5.6e to Figure 5.6h. Unlike the previous mission, the Ta-DAH Design agent is capable of relocating to the correct location in space, scoring highly on the initial relocate sub-task. This is because the inclusion of the already assembled inner BP ring means that the agent does not need to traverse as far as with the previous task. In addition to this, the space robot is capable of reliably grasping the required payload with a very high average and maximum score, but shows lower performance on the last two sub-tasks. The higher performance of this robot configuration gives credence to the Ta-DAH Design process.

Running ORCHID on this set of sub-tasks outputs an optimal configuration of

$$\mathbf{v} = [0.4, 0.4, 0.1, 1.3, 1.1, 0.7, 80].$$

As with the optimised hardware design for the previous task, ORCHID specifies a space robot with a much larger base, both in mass and dimensions. Unlike with the optimal design for the inner BP assembly ORCHID outputs a manipulator of total length $0.9\,\mathrm{m}$, where all d.o.f. are utilised.

The ORCHID optimised design performs equally or at higher levels than the Ta-DAH Design robot in all but one task — the initial relocate task. However this decrease is negligible since both the average and maximum scores for the ORCHID designed agent are so high. The biggest improvement, of 1022.55% is seen in average score for the relocate task, where the BP module is being manipulated as the agent is now able to reach the target.

PM Assembly

The final mission that the space robot must carry out is the assembly of the PM — an illustration of the sub-missions is given in Figure 5.6i to Figure 5.6l. The space robot designed using

Ta-DAH Design showed very low levels of success across all but one of the necessary sub-tasks. The agent achieved the highest performance on the grasp task, although the average score is still comparatively low.

After running ORCHID, as with the previous missions, a bigger robotic agent was designed which facilitated higher success across most of the sub-missions. The design vector output by ORCHID was

$$\mathbf{v} = [0.6, 0, 0, 1.5, 1.8, 1.3, 81].$$

As with the sub-mission of assembling the inner BP ring the optimal manipulator mimics having 2 d.o.f. with only 1 link having a length bigger than $0\,\mathrm{m}$. As well as this, the optimal base is much larger and heavier than any of the standard form factors initially considered. The ORCHID designed agent achieved increased performance in three out of four tasks, with an increase of 276.34% in the average score. However, as with the inner BP grasp sub-task the ORCHID designed system reports lower performance than the Ta-DAH Design robot.

Discussion

Looking at the optimal robotic design output by the ORCHID system for all the tasks required to assemble the large-aperture telescope on-orbit, it is clear to see that a bigger and heavier base leads to increased performance. Although not all tasks output the same size or mass base, all are $> 80\,\mathrm{kg}$ with no single dimension $< 0.3\,\mathrm{m}$. In fact, the standard form factors initially considered are insufficient, with the largest, $27U$, weighing a maximum of $40\,\mathrm{kg}$. A standard form factor is therefore not assigned to any of the designs in this chapter. The increase in mass and dimensions of the base increases its inertia, this in turn decreases the perceived effects of the dynamic coupling which acts to destabilise the motion of the satellite. As a result the system is able to move in a more controlled and stable manner, in addition, demand on the control scheme is reduced since it no longer needs to overcome these effects.

The magnitude of the dynamic coupling forces are somewhat dependent on the ratio between the payload's inertia and the system's inertia which characterises the distribution of mass. While mass alone is not enough to quantify the dynamic coupling it can act a starting point for comparison and give a rough guideline as to what size robot might be required. The Ta-DAH Design space robot for the tasks involving the BP had a base to payload ratio of $3:1$, and $2:3$ for those involving the PM. For the ORCHID designed agents, the three optimal

designs give base to payload mass ratios of around $10:1$ and $8:1$ for the inner BP, outer BP tasks respectively and $4:3$ for the PM mission. However for the latter mission, the dimensions of the base are considerably bigger, therefore increasing its inertia independent to increasing its mass. The drawback of this change in design is that the larger systems have higher fuel and power requirements. Countering this however, is the increase in space available to store bigger batteries and more propellant.

Less clear in the results is a general trend as to what size manipulator is most appropriate for the missions in this work. This conclusion in itself is in keeping with previous work stating that the sizing of such systems will be heavily dependent on the task at hand. The total length of the manipulator for each of the three tasks varies even though, in some cases, the payloads are the same. This highlights the fact that designs must be dependent on other mission parameters such as the required trajectories, governed by object location. For example, the inner BP assembly requires a manipulator of total length $0.5\,\mathrm{m}$, while the outer BP assembly requires one of length $0.9\,\mathrm{m}$. This is not as a result of payload variation, but instead the motions required in either scenario. For the inner ring assembly, the agent must traverse longer distances, requiring the base of the spacecraft to travel further. The shorter arm aids in maintaining stability over bigger trajectories since it brings the c.o.m of the full system closer to the c.o.m of the base spacecraft. If these are misaligned it introduces an additional moment arm since the AOCS applies forces to the base's c.o.m but motion occurs around the entire system's c.o.m. This amplifies forces in addition to introducing non-linearities and higher levels of complexity in the controller. One potential solution to this problem would be to use two manipulators on a single agent and transport two payloads simultaneously, therefore bringing the system c.o.m back inline with the base c.o.m.

Also noticeable in two of the optimal designs is a reduction in d.o.f. of the manipulator. This design characteristic was optimal for the inner BP assembly task, in which the agent is required to traverse the largest distance. It also occurred with the PM assembly which requires manipulation of the heaviest payload. In both of these designs, only one link has length of $> 0\,\mathrm{m}$. It is thought that ORCHID favored fewer d.o.f. in these tasks since it reduces the effects of the dynamic coupling, which is a function of the number of places in which a system is articulated. This lowering in dynamic coupling facilitates more stable motion, contributing to higher levels of performance.

Comparing the results for all tasks there only exists two sub-tasks in which performance of the ORCHID designed agent is not higher or equal than that of the Ta-DAH Design agent. The two tasks that show lower performance are the grasp sub-task for the inner BP and the PM assembly. It is thought that this low performance is due to the reduction in d.o.f. of the manipulator, making the success of dexterous tasks very hard. In fact, for dexterous manipulation to occur, as is required with the grasp task, it is necessary for the robot manipulator to utilise all available d.o.f. ORCHID was therefore run again, this time optimising over only the grasp sub-task for the inner BP ring. This led to an optimal design vector of

$$v = [0, 0.5, 0.5, 0.5, 0.4, 1.5, 0.3, 102]$$

and gave an average score of 1033.6 and a maximum score of 1080. This exceeds the performance of the Ta-DAH Design robot and highlights that more d.o.f. are beneficial for certain tasks.

5.3 Conclusion

This chapter presented ORCHID – the first RL approach to jointly optimise robotic morphology and control policies where changes can be made to the robot's kinematics. This method addressed two main problems in prior research. The first being the need to store a number of control policies and robotic designs throughout the optimisation process and the second being the iterative way in which state-of-the-art algorithms optimise the control and hardware design separately. ORCHID overcomes both of these issues by proposing a unified optimization scheme which simultaneously updates the control parameters and design parameters. The use of a differentiable transition function that allows gradient flow throughout a rollout, led to improvements in performance for every combination of environment and control policy. In addition to finding the jointly optimal control policy and hardware pairs, ORCHID boasts the added benefit of greatly increasing design speed when compared to naive brute force search methods.

Specific results were given when using ORCHID to design a small space robot for the OOA of a large aperture telescope. The resulting designs showed improvements compared to the systems designed by Ta-DAH Design in Section 4. Although they are not considered to be final designs, inspection of the results proved to be an effective tool that uncovered a number of concepts

which may not have been apparent to a human designer without lengthy prototyping. These included reducing the d.o.f. when large distances need to be traversed by the system and the need for a considerably larger base. In addition to numerous ways to mitigate the de-stabilising effect of dynamic coupling. This chapter approximated a base to payload mass ratio of $9:1$ to be acceptable, which is in keeping with previous work in the same field.

While multiple runs of the ORCHID architecture provided less variance in designs compared to the baselines, a range was still apparent. This shows that a range of different sized agents maybe suitable for the same task. However, as discussed in Section 2, the same control policy cannot be used across multiple well-designed agents. This motivates the next chapter, where the robustness of control to changing physical parameters is investigated.

Chapter 6

Hardware Agnostic Control

The result of any co-optimised design approach is a control scheme that is suitable only for use with that robot. While this provides good performance in a single situation, the drive for general AI requires more versatile and robust systems. Unless specific training has taken place, RL agents tend to be highly sensitive to the specifics of the simulator or robotic hardware used during training. This creates a huge demand for data collected using real-world hardware, something that is costly, time inefficient and in some cases dangerous. Even worse, data collected in one lab using one robot cannot easily be utilised elsewhere, unless training is to occur on the exact same robot in the exact same configuration. Even then, differences in the calibration of the robots are likely to hurt reproducibility.

Hardware-agnostic policies would allow a single network to operate in a variety of test domains, where dynamics vary due to changes in robotic morphologies or internal parameters. In addition to this, they will increase robustness to failure and degradation of the robot hardware. This chapter presents HARL-A, an RL pipeline capable of training hardware-agnostic policies, a representation of which can be seen in Figure 6.1. It exploits the fact that learning to adapt a known and successful control policy is easier and more flexible than jointly learning numerous control policies for different morphologies. Unlike standard approaches to multi-robot transfer learning, the proposed method increases the scope of the agent's ability during the learning phase, as opposed to utilising it after data collection or training. The result is a general policy that is capable of operating on semi-identical robots with zero-shot transfer, meaning no fine-tuning is required on new embodiments.

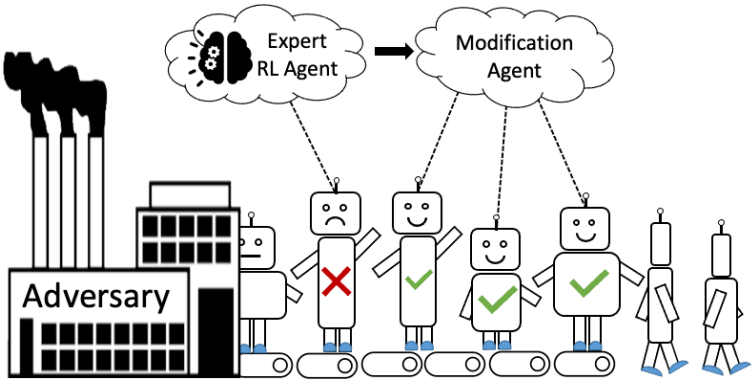


Figure 6.1: **Representation of HARL-A.** Here a single agent can successfully operate on a range of different robots, whose configuration has been determined via adversarial selection.

6.1 Methodology

HARL-A learns a single policy that successfully works on all morphology vectors drawn from the continuous distribution: $\mathbf{v} \in \mathcal{V}$. The architecture follows the structure shown in Figure 6.2, where three different systems are used:

1. Expert Network ($\pi_{\hat{\epsilon}}$) — corresponding parameters are shown by a $\hat{}$ accent.
2. Modification Network, made up of an actor ($\pi_{\tilde{\psi}}$) and a critic ($\pi_{\tilde{\omega}}$) — corresponding parameters are shown by a $\bar{}$ accent.
3. Adversary Network ($\pi_{\tilde{\delta}}$) — corresponding parameters are shown by a $\circ$ accent.

Note that any parameters with no overhead symbol are those related to the environment or all three of the HARL-A components, *e.g.* $\mathbf{s}$, r. The three systems interact in two different loops during training. The per-trajectory training loop is shown in red in Figure 6.2. In this loop, whole episodes are condensed into a single event and these are used to calculate l_p and update the adversary parameters. In blue is the per-step training loop, in which every step is stored and used to update the modification network.

The expert network is pre-trained to work on a single morphology, $\hat{\mathbf{v}} = \text{constant}$. It is trained using PPO, the implementation, and training loss of which is the same as that outlined in Section 5.1.1. During subsequent training of the modification network, $\mathbf{v}$ is sampled from the adversary network at the start of each episode, hence $\mathbf{v}_T$. This is combined with the environment

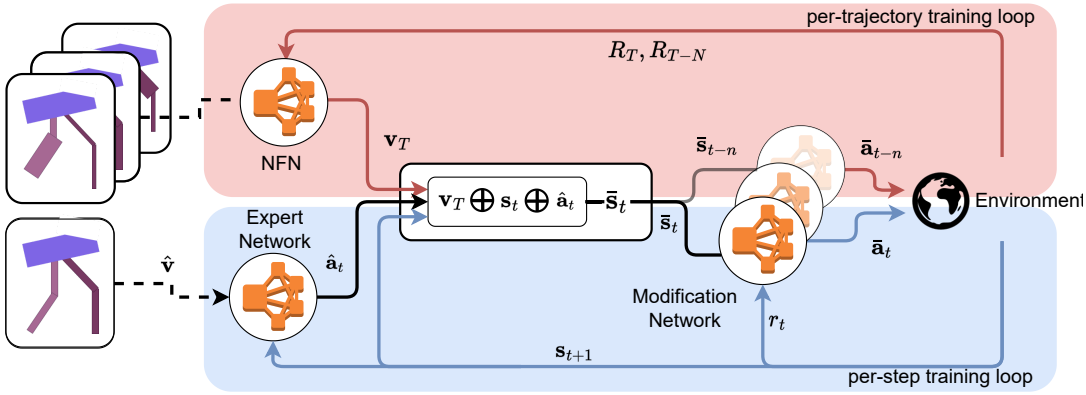


Figure 6.2: **HARL-A flow diagram.** Split of training loops used in the HARL-A pipeline. In red is the per-trajectory loop used to update the adversary parameters. In blue is the per-step loop that is used to update the modification network. The expert network is queried at every step, but its weights are fixed during the training of HARL-A

observations ($\mathbf{s}$), and the action that the canonical expert network ($\hat{\mathbf{a}}$) would have taken given that observation. This extended state ($\bar{\mathbf{s}}$) is the input to the modification network, which outputs a modified action ($\bar{\mathbf{a}}$), suitable for the robot with $\mathbf{v}_T$.

6.1.1 Modification Network

The aim of the modification network is to learn policy: $\pi_{\tilde{\psi}}(\bar{\mathbf{a}}_t|\bar{\mathbf{s}}_t)$, parameterised by $\tilde{\psi}$, that can maximise the expected reward for any $\bar{\mathbf{s}} \in \bar{\mathcal{S}}$. Unlike in a standard RL setup, the modification network leverages information from a pre-trained expert network and is conditioned on the current morphology. This is achieved by utilising a more complex network input, as previously outlined: $\bar{\mathbf{s}} = \mathbf{v}_T \oplus \mathbf{s}_t \oplus \hat{\mathbf{a}}_t$, where $\oplus$ denotes concatenation. Expanding the expression for $\pi_{\tilde{\psi}}$, gives $\pi_{\tilde{\psi}}(\bar{\mathbf{a}}_t|\mathbf{v}_T, \mathbf{s}_t, \hat{\mathbf{a}}_t)$. As with ORCHID (Section 5), the introduction of variable $\mathbf{v}$, not only introduces an additional input for the modification network, but also that means the corresponding environment's transition function becomes $\mathcal{P}\left(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{v}_T\right)$. This is still an MDP since $\mathbf{v}_T$ is constant throughout an episode. $\tilde{\psi}$ should therefore be updated according to,

$$\tilde{\psi}^* = \arg\max_{\tilde{\psi}} \mathop{\mathbb{E}}_{\mathbf{s}\sim\mathcal{P},\mathbf{a}\sim\pi_{\tilde{\psi}}} \left[R(\mathbf{a},\mathbf{s})|\pi_{\tilde{\epsilon}}(\mathbf{s})\right]. \tag{6.1}$$

Equation 6.1 is solved using PPO with loss function $\mathcal{J}_a$, where $\pi_{\tilde{\theta}t}(\mathbf{a}|\mathbf{s})$ is replaced with $\pi_{\tilde{\psi}}(\bar{\mathbf{a}}_t|\mathbf{v}_T, \mathbf{s}_t, \hat{\mathbf{a}}_t)$, as seen in Equation 5.8. This requires training of $\pi_{\tilde{\omega}}$ which is done using

Equation 5.9. However, there exists a number of different approaches to deal with the added dimension introduced by the expectation over $\mathcal{V}$. This chapter first proposes a solution of sampling $\mathbf{v}$ randomly from a pre-defined design space — referred to as Hardware Agnostic Reinforcement Learning (HARL). The limitation of this approach is that a significant amount of time is wasted training on samples that already perform well or on which learning may be impossible. The second proposed approach is the full HARL-A approach. This mitigates these issues by intelligently selecting low performing, challenging, but achievable, training samples.

The training of the modification network still relies on a critic network, since the losses used are derived from PPO. Since $\pi_{\tilde{\psi}}(\bar{\mathbf{a}}_t|\mathbf{v}_T, \mathbf{s}_t, \hat{\mathbf{a}}_t)$ is dependent on $\mathbf{v}_T$, the critic is conditioned on $\mathbf{v}_T$, in order to stabilise learning. The input to the critic network becomes $\mathbf{s}_t \oplus \mathbf{v}_T$, meaning: $\pi_{\tilde{\omega}}(V_t|\mathbf{s}_t, \mathbf{v}_T)$.

6.1.2 Adversary Network

The role of the adversary is to sample morphology vectors for the modification network to train on when the full HARL-A system is implemented. It does this by learning a mapping from an input distribution to an output distribution, where the latter characterises the performance of the modification network across the design space $\mathcal{V}$. This distribution mapping is achieved by using a NFN as the adversary network — defined as $\pi_{\tilde{\delta}}$, and parameterised by $\tilde{\delta}$.

Intuition says that the modification network should train on the lowest performing morphology vectors, in order to improve its performance in the domain of $\mathcal{V}$. In this case, the adversary network should learn to represent the distribution of exact performance over $\mathcal{V}$. However, preliminary experiments with such an adversary network demonstrate poor performance. This is because, when dealing with high dimensioned, continuous design spaces there are inevitably morphologies on which the agent can never learn to 'complete the task'. In a standard adversarial set-up, since these samples have the lowest performance, they would continually be selected for training, wasting valuable resources. Instead, the concept of learning potential (l_p) is introduced and the adversarial NFN learns to model the distribution of l_p across $\mathcal{V}$.

Learning Potential

l_p quantifies the agent's ability to learn a more effective policy for a given $\mathbf{v}$. It is calculated as the change in performance over a set number of training epochs,

$$l_p = \frac{d \sum_{t=0}^{T} \gamma^t r_t}{d\, E_c}, \tag{6.2}$$

where E_c is the episode count. Intuitively this indicates the rate of change of network performance against training episodes with a fixed morphology. It is possible to calculate l_p for a given morphology $\mathbf{v}_T$. This is done with a new copy of the modification network. A number of training epochs are run with the copied network and this fixed $\mathbf{v}_T$. The total reward for each episode during this training loop is maintained (blue loop in Figure 6.2) and a 1^{st} order polynomial is fit to the points, as seen in Figure 6.3. This data is used to solve Equation 6.2 for the chosen morphology. This approach provides a more reliable estimate of l_p by averaging over a number of episodes, therefore giving a stable estimation. Morphologies on which the modification network is most able to learn will have a large l_p. Once the NFN network has been trained to model this distribution, such morphologies will be sampled more frequently by the adversary.

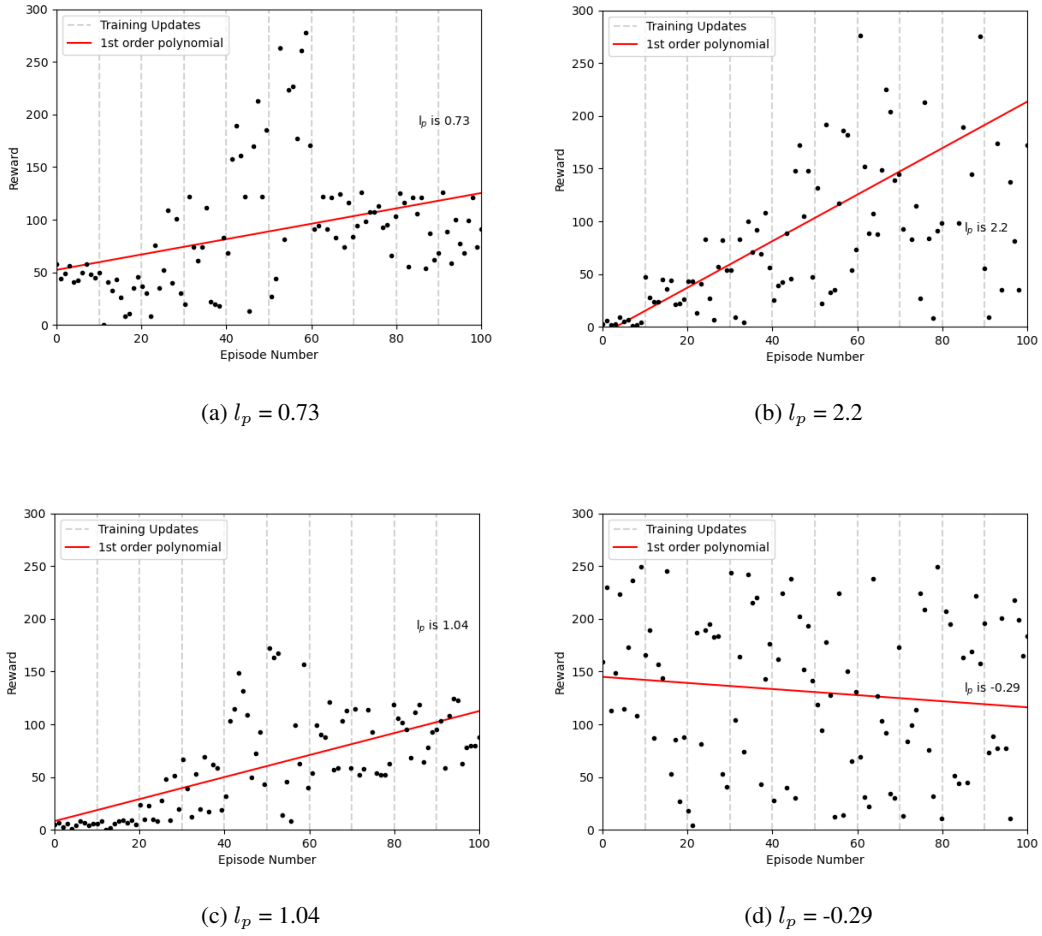
Normalising Flow Networks

An NFN embodies a transformation of a simple distribution to a more complex distribution via the composition of a number of invertible and differentiable mappings [71]. Each of these mappings are parameterised with trainable variables and the result is a complex distribution which can be sampled from or used to determine the probability of a pre-selected sample. Such a network architecture is used for the adversary network since it maps the simple random sample used by HARL the distribution of l_p over $\mathcal{V}$. Sampling from $\pi_{\tilde{\delta}}$ can then provide morphologies on which the modification network can improve its performance.

The underlying principle of NFNs is to express vector $\mathbf{p}$ as a function (f) of real vector $\mathbf{q}$ with probability distribution $\mathbb{P}_{\boldsymbol{q}}(\mathbf{q})$,

$$\mathbf{p} = f(\mathbf{q}) \quad where \quad \mathbf{q} \sim \mathbb{P}_{\boldsymbol{q}}(\mathbf{q}) \quad and \quad \mathbf{p} \sim \mathbb{P}_{\boldsymbol{p}}(\mathbf{p}). \tag{6.3}$$

f represents a warping of the real number space in order to map $\mathbb{P}_{\boldsymbol{q}}(\mathbf{q})$ to $\mathbb{P}_{\boldsymbol{p}}(\mathbf{p})$, where $\mathbb{P}_{\boldsymbol{p}}(\mathbf{p})$ is the unknown, desired target distribution. In the instance of HARL-A this is the distribution of l_p over $\mathbf{v}$. $\mathbb{P}_{\boldsymbol{q}}$ is the input distribution, usually a diagonal-covariance Gaussian [113].



(a) l_p = 0.73

(b) l_p = 2.2

(c) l_p = 1.04

(d) l_p = -0.29

Figure 6.3: **Graphical representation of l_p.**

Parameterising f and $\mathbb{P}_{\boldsymbol{q}}(\mathbf{q})$ with $\tilde{\delta}_1$ and $\tilde{\delta}_2$, respectively induces a warped distribution over $\mathbf{p}$, giving

$$\mathbf{p} = f\left(\mathbf{q} : \tilde{\delta}_1\right) \quad where \quad \mathbf{q} \sim \mathbb{P}_{\boldsymbol{q}}\left(\mathbf{q}; \tilde{\delta}_2\right). \tag{6.4}$$

For the implementation of flow models, f must be both differentiable and invertible, meaning $\mathbb{P}_{\boldsymbol{p}}(\mathbf{p})$ can be defined by a change of variables:

$$\mathbb{P}_{\boldsymbol{p}}\left(\mathbf{p}\right) = \mathbb{P}_{\boldsymbol{q}}\left(\mathbf{q}\right) \left\| \det J_f\left(\mathbf{q}\right) \right\|^{-1}, \tag{6.5}$$

where $\left\| \det J_f\left(\mathbf{q}\right) \right\|$ is the absolute of the determinant of the Jacobian of f at $\mathbf{q}$ [17]. The determinant of the Jacobian characterises the change in volume around a sample, $\mathbf{q}$, w.r.t. f.

Since f is invertible and differentiable, the successive composition (denoted by $\circ$) of many functions will result in a new, complex function, for which Equation 6.5 still applies [131]. It

follows that:

$$(f_2 \circ f_1)^{-1} = f_2^{-1} \circ f_1^{-1}, \tag{6.6}$$

and

$$\det J_{f_2 \circ f_1}(\mathbf{q}) = \det \left[J_{f_2}(f_1(\mathbf{q})) \right] \times \det \left[J_{f_1}(\mathbf{q}) \right]. \tag{6.7}$$

A Normalising Flow is made up of any number (K) of transforms: $f_K = \{f_k \circ ... \circ f_1\}$. Given the definition of f_K Equation 6.5 becomes,

$$\mathbb{P}_{\boldsymbol{p}}(\mathbf{p}) = \mathbb{P}_{\boldsymbol{q}}(\mathbf{q}) \prod_{i=1}^{K} \left| \det J_{f_{i-1}}(\mathbf{q}_{i-1}) \right|^{-1}. \tag{6.8}$$

The optimal NFN will learn $\tilde{\delta}$ such that Equation 6.8 $\approx$ the optimal target distribution $\left(\mathbb{P}^*_{\boldsymbol{q}}(\mathbf{q})\right)$. To achieve this, their divergence should be minimised. The most popular way of quantifying the divergence between the two probability distributions is using the Kullback–Leibler (KL) divergence [71]. There exists two different manipulations of the KL divergence (D_{KL}) as a loss function in order to optimise $\tilde{\delta}$. The first relies on being able to evaluate $\mathbb{P}^*(\mathbf{p})$ in a differentiable manner, and the second relies on being able to draw samples from $\mathbb{P}^*(\mathbf{p})$. The latter is used in this work. Using the definition of KL divergence, it is possible to define a loss function $(\mathcal{J}_{adv})$ to optimise parameters $\tilde{\delta}$.

$$\begin{aligned}
\mathcal{J}_{adv} &= D_{KL}\left(\mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p}) \,\middle\|\, \mathbb{P}_{\boldsymbol{p}}(\mathbf{p}; \tilde{\delta}) \right) \\
&= \int_{\mathbb{R}} \mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p}) \log \left(\frac{\mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p})}{\mathbb{P}_{\boldsymbol{p}}\left(\mathbf{p}; \tilde{\delta}\right)} \right) d\boldsymbol{p}, \\
&= \int_{\mathbb{R}} \mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p}) \log \left(\mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p}) \right) d\boldsymbol{p} - \int_{\mathbb{R}} \mathbb{P}^*_{\boldsymbol{p}}(\mathbf{p}) \log \mathbb{P}_{\boldsymbol{p}}\left(\mathbf{p}; \tilde{\delta}\right) d\boldsymbol{p} \\
&= -\mathop{\mathbb{E}}_{\mathbb{P}^*_{\boldsymbol{p}}} \left(\log \mathbb{P}_{\boldsymbol{p}}\left(\mathbf{p}; \tilde{\delta}\right) \right) + \text{const}
\end{aligned} \tag{6.9}$$

and using the inverse of Equation 6.3 and the definition of $\mathbb{P}_{\boldsymbol{p}}(\mathbf{p})$ in Equation 6.5, the loss function becomes,

$$\mathcal{J}_{adv} = \mathop{\mathbb{E}}_{\mathbb{P}^*_{\boldsymbol{p}}} \left[\log \mathbb{P}_{\boldsymbol{q}}\left(f^{-1}(\mathbf{p}) \times \tilde{\delta}_1; \tilde{\delta}_2 \right) + \log \left| \det J_{f^{-1} \times \tilde{\delta}_1}(\mathbf{p}) \right| \right] + const. \tag{6.10}$$

Sampling from $\mathbb{P}^*_{\boldsymbol{p}}$ means a Monte Carlo estimate of Equation 6.10 can be found,

$$\mathcal{J}_{adv} = -\frac{1}{\mathring{N}} \sum_{\mathring{n}=1}^{\mathring{N}} \left[\log \mathbb{P}_{\boldsymbol{q}}\left(f^{-1}\left(\mathbf{p}_{\mathring{n}}\right) \times \tilde{\delta}_1; \tilde{\delta}_2 \right) + \log \left| \det J_{f^{-1} \times \tilde{\delta}_1}\left(\mathbf{p}_{\mathring{n}}\right) \right| \right], \tag{6.11}$$

where $\mathring{N}$ is the number of samples in a batch.

Following the definition of $\mathcal{J}_{adv}$, the implementation of a NFN relies on the definition of an appropriate set of functions, or flow blocks — f_K. Constraints include those mentioned previously in addition to the Jacobian of each being easy to calculate. Two different types of flow are utilised in this chapter. The first is a simple $1D$ transform, used when $\mathbf{v} \in \mathbb{R}^1$. This situation is trivial, since the idea behind NFNs is to model the probability of higher dimensional data, therefore the majority of the literature proposes flow blocks that do not work in the 1D case. However, it is possible to use a weighted combination of Cumulative Distribution Functions (CDF), separated by a non-linear activation function (Φ), where the μ and σ and weights (w) make up $\tilde{\delta}$:

$$f_{\text{CDF}} = \Phi(\text{w}\mathcal{N}(\mu, \sigma)) \tag{6.12}$$

By nature, CDF and non-linear activation functions are differentiable, and their Jacobian's are well-defined, while the fact they are continuously increasing makes them invertible.

For higher dimensional spaces, Real-Valued Non-Volume Preserving (RNVP) flows are used [42]. This block works by splitting input $\mathbf{q}$; the first j dimensions are unchanged, and an affine transform, defined by the first dimensions, is applied to the remaining components of $\mathbf{q}$,

$$f_{\text{RNVP}} = \begin{cases} \mathbf{p}_{1:j} & = \mathbf{q}_{1:j} \\ \mathbf{p}_{j+1:d} & = \mathbf{q}_{j+1:d} \odot exp\left(S\left(\mathbf{q}_{1:j}\right)\right) + T\left(\mathbf{q}_{1:j}\right), \end{cases} \tag{6.13}$$

where $S()$ and $T()$ are the scale and translation functions and $\odot$ is the element-wise Hadamard product. Since $\mathbf{p}_{1:j} = \mathbf{q}_{1:j}$ neither S or T need to be invertible in order to solve Equation 6.8. This is because f^{-1}_{RNVP} can be calculated without needing to invert either function,

$$f^{-1}_{\text{RNVP}} = \begin{cases} \mathbf{q}_{1:j} & = \mathbf{p}_{1:j} \\ \mathbf{q}_{j+1:d} & = \mathbf{p}_{j+1:d} - T\left(\mathbf{p}_{1:j}\right) \oslash exp\left(S\left(\mathbf{p}_{1:j}\right)\right), \end{cases} \tag{6.14}$$

where $\oslash$ is Hadamard division. In addition to this, $\det(J_{f_{\text{RNVP}}})$ is easy to compute since $J_{f_{\text{RNVP}}}$ is lower triangular and does not require $\det(J_T)$ or $\det(J_S)$. This means that S and T can be complex functions that are parameterised via $\tilde{\delta}$. The implication of this splitting however, is that if many of these flows are appended, the first j dimensions will always remain unchanged. Therefore, these blocks are used in pairs where neighboring blocks have their channels reversed.

In summary, the adversary network is a NFN which is made up of a stack of f_K, either f_{RNVP}, or f_{CDF}, depending on the dimensionality of $\mathbf{v}$. The network learns a complex characterisation

of the ability of the modification network to improve its performance over $\mathcal{V}$. The aim of this is that when queried, $\pi_{\tilde{\delta}}$ will provide appropriate samples for the modification network to train on.

Optimisation of the NFN requires a batch of morphologies drawn from the target distribution, where the target distribution is directly related to the l_p. As such, it is necessary to develop a method for turning sets of l_p and corresponding **v** (calculated in the blue loop using the method outlined in Section 6.1.2) into a set of points drawn from the desired distribution. The l_p values are positively scaled and used as the weights for multinomial sampling [16]. This distribution can then be queried to provide as many samples as required for adversary training, this is done with replacement.

6.1.3 Implementation

Algorithm 1 outlines the system training. Training occurs in two separate loops as defined by the colors in Figure 6.2, the first collects samples for and updates $\pi_{\tilde{\psi}}$ and the second does the same but for $\pi_{\tilde{\delta}}$. $\pi_{\tilde{\psi}}$ has the same architecture as in Section 5, with both an actor and critic component. $\pi_{\tilde{\delta}}$ is made up of the flows discussed in Section 6.1.2. In the 1D case, $K = 3$ and in the higher dimensional problems $K = 8$ with dimension swapping in between each.

A number of different loops are required during training:

- Total number of updates to the entire system: $o = 10e3$
- Training epochs for the modification network per 1 system update: $m = 128$
- Steps in the environment per modification training batch *i.e.* modification batch size: $p = 2500$
- Iterations over one batch of collected experience for the modification network: $q = 4$
- Training epochs for the adversary network per 1 system update: $n = 64$
- Number of l_p and morphology pairs collected per adversary updated *i.e.* adversary batch size: $F = 15$

The same values are used for all experiments in this chapter. The same learning rate is used for updating $\tilde{\delta}$, $\tilde{\omega}$ and $\tilde{\psi}$, where $\mathcal{L}_{mod}$=$\mathcal{L}_{NFN}$=$\mathcal{L}_c$= $7e-4$.

1: Randomly initialise $\tilde{\delta}$ and $\tilde{\psi}$
2: **for** $o = 1, 2, \ldots$ total number of updates **do**
Modification Network Update
3: **for** $m = 1, 2, 3, \ldots$ number of modification network updates **do**
4: Initialise memory for modification network $\bar{M} \leftarrow \varnothing$
5: Sample morphology and start state: $\mathbf{v}_T^0 \sim \pi_{\tilde{\delta}}, \mathbf{s}_0^m \sim \mathcal{S}$
When training the adversary sample $\mathbf{v}_T^0$ from a random distribution
6: **for** $p = 1, 2, \ldots$ number of steps per modification network update **do**
7: Sample action: $\hat{\mathbf{a}}_p^m$ from expert network
8: Determine mod state: $\bar{\mathbf{s}}_p^m = \mathbf{v}_T^m \oplus \mathbf{s}_p^m \oplus \hat{\mathbf{a}}_p^m$
9: Sample modified action: $\bar{\mathbf{a}}_p^m \sim \pi_{\tilde{\psi}}\left(\bar{\mathbf{a}}_p | \bar{\mathbf{s}}_p^m\right)$
10: Determine reward: $r_p^m = \mathcal{R}(\mathbf{s}_{p+1}^m, \bar{\mathbf{a}}_p^m, \mathbf{v}_T)$
11: Store transition: $\bar{M} \leftarrow \bar{M} \cup (\bar{\mathbf{s}}_p^m, \bar{\mathbf{a}}_p^m, r_p^m, \bar{\mathbf{s}}_{p+1}^m)$
12: **if** Episode ends **then**
13: Sample morphology and state: $\mathbf{v}_T^t \sim \pi_{\tilde{\delta}}, \mathbf{s}_p^m \sim \mathcal{S}$
When training adversary only sample state: $\mathbf{s}_p^m \sim \mathcal{S}$
14: **end if**
15: **end for**
16: **for** $q = 1, 2, \ldots$ number of PPO epochs **do**
17: Sample mini batch from $\bar{M}$
18: Calculate $\mathcal{J}_a$ and $\mathcal{J}_c$ and update: $\tilde{\psi} \leftarrow \tilde{\psi} + \mathcal{L}_{mod}\nabla(\mathcal{J}_a)$ and $\tilde{\omega} \leftarrow \tilde{\omega} + \mathcal{L}_c\nabla + \mathcal{J}_c)$
19: **end for**
20: **end for**
Adversary Network Update
21: **for** $v = 1, 2, 3 \ldots$ number of adversary network updates **do**
22: Initialise memory $\mathring{M} \leftarrow \varnothing$
23: **for** $F = 1, 2, \ldots$ number of updates for l_p **do**
24: Make copy of $\pi_{\tilde{\psi}} \sim \mathring{\pi}_{\tilde{\psi}}$
25: Repeat lines 4 to 15, using $\mathring{\pi}_{\tilde{\psi}}$
26: Calculate l_p and store data: $\mathring{M} \leftarrow \mathring{M} \cup (\mathbf{v}_T^v, {l_p}^v)$
27: **end for**
28: Draw a batch of samples of $\mathbf{v}$ from $\mathring{M}$ using multinomial sampling
29: Calculate $\mathcal{J}_{adv}$ and update: $\tilde{\delta} \leftarrow \tilde{\delta} + \mathcal{L}_{NFN}\nabla\mathcal{J}_{adv}$
30: **end for**
31: **end for**
32: **return** $\tilde{\delta}$ and $\tilde{\psi}$

Algorithm 3: Implementation of HARL-A

Depending on the desired target environment for use of the trained HARL-A system, it is necessary to restrict the output of $\pi_{\tilde{\delta}}$ between certain values. This is achieved by applying the sigmoid function to all the sampled $\mathbf{v}_T$ within a batch (of which there will be q samples) and rescaling the output to the desired range.

6.2 Experiments and Results

The performance of HARL and HARL-A is demonstrated in three environments, Cart Pole, Bipedal Walker and the Space Robot.

6.2.1 Environments

Cart Pole

Experiments are carried out varying the length of the pole: $\mathbf{v} \in \mathbb{R}^1$ or the length of the pole and the mass of the cart: $\mathbf{v} \in \mathbb{R}^2$. The same material choices are used throughout so varying the pole's length also changes its mass. The expert network is trained for a pole length of 1 and a cart mass of 1. The same expert network is used for all experiments.

Bipedal Walker

The aim is for the walker to travel as far as it can in 250 time steps using as little energy as possible. The reward is total distance traveled minus energy expended due to actuator motion. Two different setups are used with the bipedal walker. In both cases the size of the walker's legs are varied. The first mode is referred to as 'whole' and legs are varied by the same amount: $\mathbf{v} \in \mathbb{R}^1$. When the length and width of each leg segment is varied independently: $\mathbf{v} \in \mathbb{R}^8$ and experiments are referred to as 'individual'. For both situations the expert network is trained for a walker of scale or leg size 1.

Space Robot

The ORCHID trained systems from the previous chapter are used as the expert network and baseline morphology for all experiments using the space robot. This means that each mission has a different baseline design. The length of each link, and the size and mass of the base is varied during the following experiments. As such, $\mathbf{v} \in \mathbb{R}^7$.

Table 6.1: **Testing ranges.**

Environment	Parameters		A	B	C
Cart Pole	Pole	$\mathbf{v} \in \mathbb{R}^1$			[0.5,10]
	Cart, Pole	$\mathbf{v} \in \mathbb{R}^2$			[0.5,10]
Bipedal Walker	Whole	$\mathbf{v} \in \mathbb{R}^1$	[1,1.25]	[0.75,1]	[0.75,1.25]
	Individual	$\mathbf{v} \in \mathbb{R}^8$	[1,1.25]	[0.75,1]	[0.75,1.25]

6.2.2 Testing Parameters

A number of experiments were run to test the generalisation of HARL-A over different morphology distributions. For experiments other than those involving the space robot, which are included later, values are quoted in Table 6.1. The values in the table refer to the range of values used during training. For the HARL system the range randomly sampled from, and for HARL-A this is the range which the adversary is trained over. Each experiment was run 3 times with 3 different seeds, for 1×10^6 steps in the corresponding environment and all results quoted are an average of the 3 runs. To fairly evaluate performance, a consistent test distribution was used for each experiment. This consists of 500 random morphology vectors from the defined range. HARL and HARL-A are both compared against current state-of-the-art in the field of generalisation to embodiment — Hardware Conditioned Policies (HCP) [29], and a standard PPO baseline.

6.2.3 Modification Network *vs* Direct Learning

The use of both the modification network and expert network in the HARL architecture is validated by comparing the final performance of HARL, HCP and PPO. In this set of experiments, each algorithm was trained with a random sample from $\mathcal{V}$ for each T without the use of a l_p based adversary. The results for all experiments, excluding the space robot are shown in Table 6.2. In all but one of the experiments both HCP and HARL perform better than PPO, showing that giving the network knowledge of its embodiment can improve its ability to learn a robust policy. Further to this, HARL reports the best performance in $8/9$ experiments, with HCP showing the best performance in the remaining experiment. The average improvement of HARL over HCP (the next best performance) in experiments in the cart pole environment is 69.5% and

Table 6.2: **Comparison between HARL and baselines.**

Experiment			PPO	HCP	HARL
Cart Pole	Pole	C	366	*580*	**650**
	Cart, Pole	C	*427*	338	**645**
Bipedal Walker	Whole	A	147	*173*	**203**
		B	123	*185*	**190**
		C	126	**181**	*171*
	Individual	A	132	*191*	**222**
		B	133	*139*	**164**
		C	126	*174*	**202**

26.75% for those in the bipedal walker environment. Conversely, for the remaining experiment in which HARL does not report the highest performance, the corresponding decreased is 5.5%. This decrease is considerably lower than the increase seen across the other experiments. This improvement between HARL and HCP shows how learning to modify the output of a pre-trained network in contrast to learning a single robust policy from scratch leads to higher performance, validating the use of the modification network and expert network in series.

Figure 6.4 and Figure 6.5 show how policies learned to perform as a function of the varied design parameter. While the performance of HARL is not the highest at all values, particularly in the case of Cart Pole, it does show the highest levels of generalisation across embodiment, which was the aim of this chapter. This shows there may be a trade-off between generalisation and localised performance. It is interesting to note that the high generalisation performance of HARL extends outside of the testing range, to embodiments which could not have been seen during training. In the case of larger scales for cart pole and lower scales for bipedal walker, HARL is able to extrapolate its control policy at a considerably higher level than the baselines — something that is not captured in Table 6.2.

6.2.4 Adversarial Learning

The next experiments explore the benefits of the adversarial training using NFNs and l_p with both HARL and the baseline HCP. These are named HARL-A and HCP-A respectively. Results

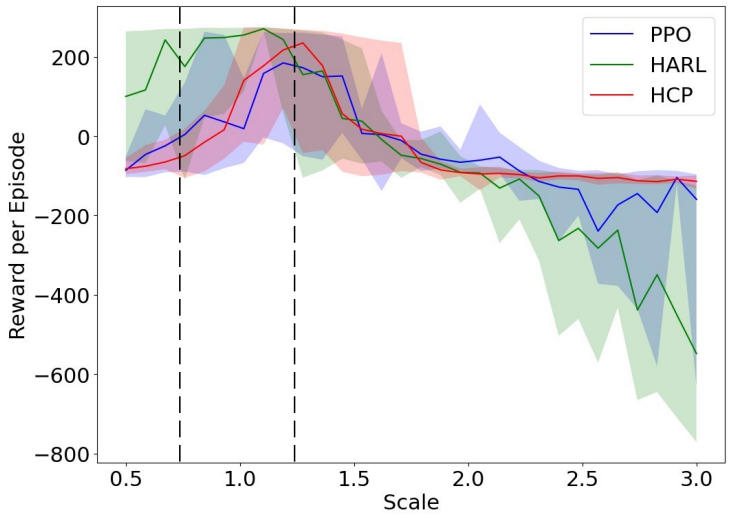


Figure 6.4: **Performance in Bipedal Walker.** Comparison of the performance of different agents in the Bipedal Walker environment as a function of leg scale. This training distribution for this experiment was $[0.75, 1.25]$, as shown by the dotted lines.

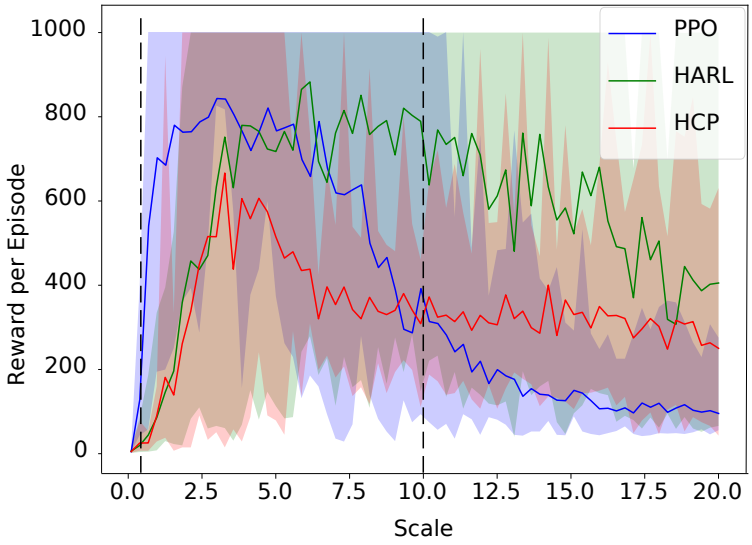


Figure 6.5: **Performance in Cart Pole.** Comparison of the performance of different agents in the Cart Pole environment as a function of pole length. In this experiment only the pole length was varied from $[0.5, 10]$, as shown by the dotted lines.

Table 6.3: **Performance with intelligent morphology sampling.**

Experiment			HCP	HCP-A	HARL	HARL-A
Cart Pole	Pole	C	580	537	**650**	*647*
	Cart, Pole	C	338	442	*645*	**749**
Bipedal Walker	Whole	A	173	192	*203*	**241**
		B	185	188	*190*	**213**
		C	181	*183*	171	**195**
	Individual	A	191	173	*222*	**229**
		B	139	163	*164*	**207**
		C	174	177	**202**	*189*

are shown in Table 6.3. For all but two experiments HARL-A, shows the highest performance. The first experiment in which HARL-A does not have the highest performance is when the pole length of Cart Pole is varied. In this instance, HARL has the highest performance. It is likely that this is due to the limited effect that changing the physical parameters has on the dynamics of the simple problem, as well as the fact that the system is near saturation. By definition, the adversarial network is only of use if there exist morphologies on which the system cannot learn. It is very possible that there are no pole lengths on which learning is impossible in this setting — as implied by Figure 6.5. In this case, the use of intelligent sampling would be redundant. This hypothesis is backed up by a similarity in results between HCP & HCP-A and HARL & HARL-A. Instead, improvements in performance are seen in this experiment due to the overall architecture alone.

The other experiment in which HARL-A does not report the highest performance is with the hardest experiment involving the Bipedal Walker. Again, the best performance is seen with HARL. It is possible that there was insufficient training time for the adversary to learn a reasonable distribution across this wider space.

It is also interesting to note that in all but two experiments the use of the adversarial NFN also improved the performance of the baseline HCP architecture. In fact, in experiment C with the whole Bipedal Walker, HCP-A reports higher performance than HARL. This proves the

importance of the intelligent sampling technique, and the novel loss function — l_p, used in this work.

6.2.5 Normalising Flow Network Analysis

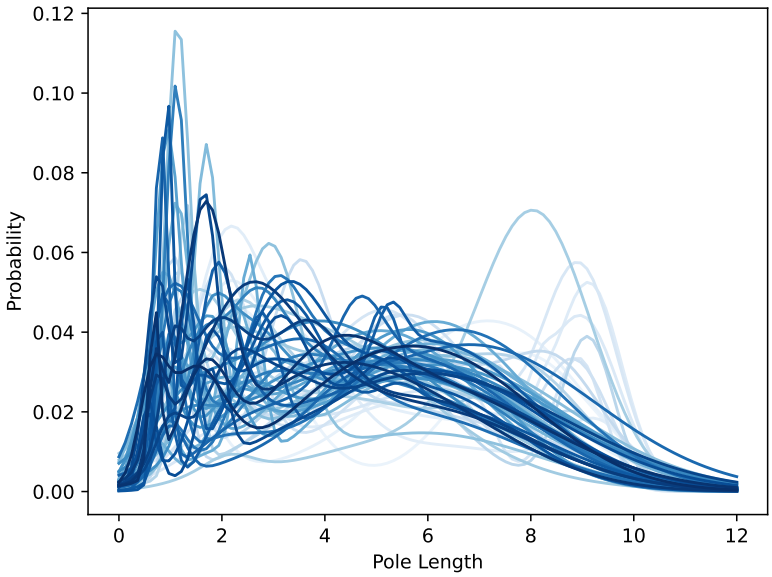
It is possible to draw a number of assumptions from the variation in the NFN throughout training. Figure 6.6 shows the variation in the estimation of the learning potential over time for the experiment when the pole length of Cart Pole is varied. Looking at the first 10 iterations (Figure 6.6b), it becomes apparent that the system believes it can learn on scales around 2, 6 and 10. Comparing this to the distribution for the last 10 iterations (Figure 6.6c), sampling from the network will likely give scales clustered around 6 and 2. While direct conclusions cannot be drawn about performance at larger scales, since this characterises the distribution of l_p not reward, some assumptions can be made via comparison with Figure 6.5. It is most likely that the RL agent learned to perform the task to a high standard fairly quickly at longer pole lengths. This is because it shows high final performance in this region following a few samples at the start of training. Conversely, reaching the same levels of performance at the lower lengths required more training since this is the region of the space that was most sampled later on.

The results in Table 6.3 showed that HARL gave marginally better performance than HARL-A for the experiment shown in Figure 6.6. It is possible that this is due to high performance across all morphologies. This is heavily backed up by the changing distribution in Figure 6.6a, which shows that there exists no part of the design space in which the probability of selection is zero.

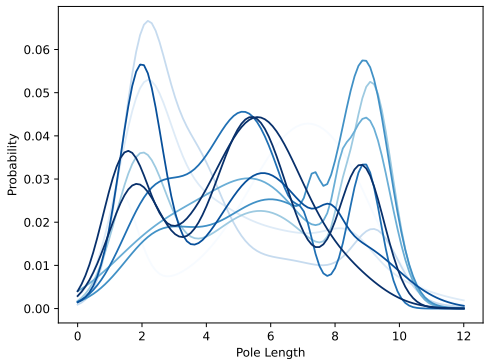
The distribution of l_p throughout training can also be utilised to give information about certain designs. If there is a region of the morphology space that has not been sampled from, it implies that it has a very low or negative l_p. This can mean one of two things — that the agent has already achieved the highest performance on this morphology (as with the previous experiment), or more likely that it is not able to learn at all.

6.2.6 Failure Modes

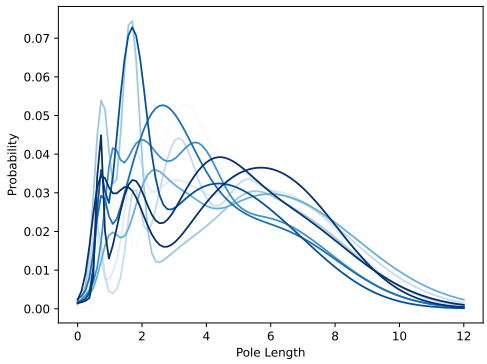
The HARL-A framework is designed to produce hardware agnostic policies which can be deployed effectively on a range of similar hardware. However, another intriguing possibility is to adapt HARL-A to provide robustness against outright failure cases. To achieve this we



(a) All updates



(b) First 10 updates



(c) Last 10 updates

Figure 6.6: **Variation of NFN over time for HARL-A with 1 dimension.** Changing distribution on which HARL-A was trained for just the pole length in the Cart Pole environment. The lighter colour indicates an older version of the distribution.

modify the framework to disable a different set of sensors in every episode. The expanded state which the modification network operates on includes a binary vector defining which sensors are disabled for that training episode.

Table 6.4: **Performance of failure modes.**

Experiment	PPO	HCP	HCP-A	HARL	HARL-A
Cart Pole	280	323	273	*742*	**920**
Bipedal Walker	-88.1	-91.5	-79.8	*89.8*	**150.4**

In addition to a varied input to the modification network, the NFN set-up varies since this is now a discrete distribution as opposed to continuous. To account for this, a threshold is applied to the output of the adversary network. If the output is > 0.5 then the sensor is on, and if it is lower then it is off. Random samples from a continuous distribution between 0 and 1 are used to train the NFN in this instance. The expert network remains the same and is trained on a fully working consistent robotic agent.

Results

The ability of HARL-A to increase redundancies in robotic agents at no extra cost is demonstrated in both the Cart Pole and Bipedal Walker environment. This is done by disabling any number of sensors. A comparison between the performance of HARL, HARL-A and the previously mentioned baselines is given in Table 6.4. This shows that the proposed architectures in this work were the only one to achieve any reasonable level of performance. This implies that the use of the expert network makes it considerably easier for an agent to learn a policy robust to missing inputs. In both cases HARL-A also outperformed HARL. This is an indication that there are sensors in both systems which are pivotal to success. Further to this, it demonstrates that the NFN was able to learn which these were — showing success over a discrete design space. Comparison with results in Section 6.2.4 show that for any algorithm, higher performance can be reached with a loss of input information compared to with a variation in the system dynamics for Cart Pole. Conversely, the opposite can be seen for the Bipedal Walker. This could be due to the relationship between sensors in the different environments. Those of Cart Pole are much more correlated, and inputs can be estimated from each other. Whereas with the Bipedal Walker observations are uncorrelated so a loss of any input causes a unrecoverable loss of information.

Table 6.5: **Space robot performance with HARL-A.**

		Sub-mission			
		Relocate	Grasp	Relocate	Fine Manipulation
Algorithm		Average reward			
Inner BP Assembly	PPO	-253.3	-61.1	-511.6	-104.6
	HCP	-181.2	-40.6	-263.2	-127.5
	HCP-A	*-175.5*	*-40.2*	-201.7	-100.1
	HARL	-196.9	-42.1	*-199.2*	*-92.3*
	HARL-A	**-175.2**	**-40.1**	**-180.3**	**-39.4**
Outer BP Assembly	PPO	999.5	-54.1	-205.6	-221.9
	HCP	1104.1	*-30.6*	-160.4	-167.5
	HCP-A	1126.7	-30.7	*-159.6*	-142.8
	HARL	**1138.9**	-35.1	-171.1	*-137.9*
	HARL-A	*1133.6*	**-30.3**	**-101.3**	**-47.3**
Outer BP Assembly	PPO	738.2	-64.4	-1023.0	-131.3
	HCP	886.6	-30.2	-470.5	-173.3
	HCP-A	**1018.3**	**-30.3**	*-109.5*	*-159.6*
	HARL	1014.1	-35.5	-178.3	-143.6
	HARL-A	*1011.6*	**-30.3**	**98.4**	**-129.0**

6.2.7 Robust Space Robot Control

Finally, the performance of HARL-A was evaluated in the space robot environment. This was done by varying each of the 7 parameters in the design vector by $\pm 2\%$ compared to the ORCHID designs of the previous chapter — the results are shown in Table 6.5. For $10/12$ sub-missions HARL-A shows the maximum or comparable reward, with the PPO baseline showing the worst performance across the board. Of interest in this set of experiments compared to those on the Bipedal Walker or in Cart Pole is that in the majority of cases the performance of HCP-A is second to HARL-A. It is likely there are a number of morphologies for which the missions are unsolvable. As a result, the added value of the intelligent sampling technique is considerably higher in this setting since it is capable of learning the morphologies on which

mission completion is unlikely, preventing wasted training epochs. The exception is the grasp task for the outer BP assembly in which HARL-A shows the highest results, with HCP showing the next best. While unexpected, performance for all algorithms other than the PPO baseline is comparable with only a 16% difference between the best and worst performing algorithm.

Overall, performance of the space robot when controlled with a HARL-A trained policy is lower for all missions than when controlled with an ORCHID trained policy. This demonstrates the trade-off between localised and generalised performance as discussed previously. However, a small change to the robot calibration or embodiment is likely to invalidate the ORCHID policy, while the HARL-A policy will remain largely unaffected.

6.3 Conclusion

This chapter presented HARL-A — a system to train a new type of hardware-agnostic policy using a form of adversarial selection. This system aimed to tackle two generalisation problems. The first was in domains with varying dynamics, *i.e*. across similar robots with different morphologies or calibrations. The second was as a method of increasing robustness to sensor failures. HARL-A was shown to overcome both of these issues, showing improved performance over state-of-the-art and standard RL algorithms.

First the idea of a modification network architecture to enable multi-robot learning was presented. This led to performance improvements in current state-of-the-art, showing that learning to modify pre-learnt behavior is more successful than learning from scratch. Then the concept of using adversarial inspired selection based on the learning potential was introduced. This is a novel metric that quantifies the ability of a policy to train on a particular robot morphology. The implementation of this metric with the adversarial network used in conjunction with the modification network (the full HARL-A system) enabled a single policy to act on a range of semi-identical robots despite a range of sensor failures.

Chapter 7

Conclusions and Future Work

The use of robotic manipulators to achieve tasks is a long explored topic spanning a number of different fields. Finding the optimal geometry of such a manipulator remains one of the most complicated problems in robotic design [74]. Traditional approaches to implementing systems involved designing robots whose quality was evaluated using simplistic control schemes [70, 73, 82]. Upon finalising designs, it was typical for the controller to be refined and optimised for the specific task. If further changes to the design were made at this point, controller refinement needed to be carried out again due to dependence on the robot's dynamics.

This intrinsic coupling between a robot's physical design and control provided the motivation for the thesis. The overarching aim was to investigate the relationship between robotic hardware and control and how either could be modified to improve overall performance. Throughout this work the motivating use case of a small free-flying robotic agent for OOA was used for evaluation. There were three main objectives of this work:

1. To explore the automation of hardware optimisation techniques as a way of improving the performance of a small space robot carrying out OOA.

2. To develop an automated design methodology that will reason over hardware and software simultaneously, rather than using a disconnected two stage approach.

3. To develop RL task-based control solutions for generalisation to different robotic hardware.

This chapter will give a breakdown of each contribution demonstrating the successful implementation of each across a variety of simulations. This work has advanced state-of-the-art in multiple fields, with general contributions in the area of OOA and RL. In addition to this, two simulators that capture the full kino-dynamic system of a space robot were developed. Both are valuable tools and are available for use in future research.

7.1 Conclusions

Chapter 4 introduced Ta-DAH Design, an automated design technique for small free-flying space robots. This addressed objective 1 in addition to providing a benchmark design for the rest of the work in this thesis. The design of such a system is a challenging problem due to the phenomenon of dynamic coupling. Only the presentation of final designs exist in the literature, and even these lack detail on the actual spacecraft parameters [39, 44, 163]. Ta-DAH Design used a cost function in a MOO pipeline to output an optimal design for a space robot based on input conditions. These input conditions required the definition of tasks at a low level in the form of joint angles and base position. A space robot was designed for each of the missions necessary for the OOA of a large aperture space telescope, the architecture of which was also presented in this work. An 11 d.o.f. system was deemed optimal for all missions, where the base sizing and link lengths vary with mission specifics. This chapter provided clear details on how each design parameter was reached. An in-depth analysis of how the different parameters interact with each other and impact overall performance was also carried out and presented. This information can be used by entities in the future when looking at sizing space robots for different missions, cutting down on the time needed for R&D.

Chapter 5 introduced ORCHID, a method for the co-optimisation of robotic embodiment and control. This addressed objective 2. Previous approaches to the co-design problem suffered from two main limitations. First, the requirement to store a number of control policies and robotic designs throughout training for later comparison [9, 170]. Second, having to iterate between policy updates and morphology updates [88]. Conversely, ORCHID allowed for simultaneous updates to a parameterised design and control policy via the use of a differentiable simulator. This allowed information on how variations in robotic morphology would impact overall performance to be quantified and exploited using back-propagation. Not only were improvements in overall performance seen when implementing ORCHID, but design times were

also drastically lowered. Experiments showed that the iterative nature in which the optimisation previously occurred led to sub-par results in comparison to the simultaneous optimisation used in ORCHID. This method of co-optimisation can aid in streamlining the design process of robotic agents, with particular gains in the early prototyping stages.

ORCHID was also evaluated in the OOA mission. In general, the resulting designs showed improvements over those designed by the Ta-DAH Design system, this highlighted the need for control and morphology co-optimisation. However, there did exist a lack of consistency in the final results as well as in the agent design. Therefore, it is not suggested that these are a finalised design or control scheme. The industry is not in a place to implement neural network control in-situ due to memory overheads, and a lack of consistency in the stochastic policies that does not yet conform with the high safety requirements needed for space missions. Instead, the application of ORCHID validated the possibility of using free-flying space robots to assemble the proposed telescope. More importantly, analysis of the final designs provided a number of usable conclusions for future mission design. This included ways to reduce the effect of dynamic coupling and an empirical approximation of the necessary payload mass to base ratio for a stable system.

Chapter 6 presented HARL-A and in doing so addressed objective 3. This architecture improved the robustness of an RL controller to changes in robotic morphology and failure modes. This problem had received less attention than controller robustness to changing tasks [102, 125]. Solutions that did address the problem directly usually required some experience in the new environment [6, 49], while others showed limited improvements over baselines [112, 122]. The architecture used in HARL-A involved an expert network and a modification network to enhance learning. This showed improvements over state-of-the-art, demonstrating how modifying a pre-learnt behavior is more successful than learning and distilling multiple policies. The introduction of an intelligent sampling technique allowed the system to self-regulate the morphologies on which the system trained, leading to further improvements in performance. This sampling technique alone made improvements to the performance of baseline algorithms as well as to HARL-A.

7.2 Short Term Future Work

The techniques presented in this thesis show some limitations and thus provide promising areas for short-term future work.

In Chapter 4, Ta-DAH Design relied on the definition of tasks at a low-level, requiring an in-depth knowledge of the mission scenario in order to define start and finish configurations. These were then interpolated using a high order polynomial to determine intermediate states. This is not an easy task, and it has been shown that a slight variation in input can lead to a large variation in both the optimal design and output behavior. Instead, Ta-DAH Design should be combined with a path planning algorithm that can determine the optimal joint angles and base position throughout the desired trajectory. However, path planning algorithms can be complex to implement and are dependent on the robotic morphology. As such, it would be interesting to develop a more simplistic, quick to implement path planning approach that is parameterised by the robot's embodiment. Thus, providing more realistic data for analysis in the Ta-DAH Design pipeline, improving the output designs.

The methods presented in both chapter 5 and 6 are constrained by the input and output dimensions of the actor network. This limits each to always operate on an agent with the same d.o.f. As such, the development of compact techniques in which the control policy can perform over various input and output dimensions would be an interesting field of research. This would allow HARL-A to operate across robots with varying numbers of joints, further increasing robustness to changing dynamics. With this ability, the architecture could then be adapted for use in failure cases, meaning that the same control policy could be used when certain joints or motors on a robot fail.

In addition, there exist opportunities to improve the efficiency of HARL-A at runtime, further increasing its appeal for implementation. The major constraint is the need to sample from both the modification network and expert network. This increases computational demand. Instead, the modification network should learn and encode the expert network such that they are not both needed at run time.

The nature of ORCHID requires differentiable training of the control policy. This limits additional RL techniques that can be utilised in order to improve the performance, such as HER or curriculum learning [5]. An interesting avenue for future work would therefore be

to investigate methods that can be used in the ORCHID pipeline. This would mean that the performance of agents would improve, ideally facilitating the completion of all tasks in the space robot simulator.

7.3 Directions for the Field

The contributions of this thesis provide a number of interesting areas for future research in the field of space robotics and RL. In particular, topics are discussed that will unify the two fields in order to strive for the implementation of RL techniques in-situ.

The methods presented in chapter 4 and 5 look into the design of robotic agents. Both contributions showed how modifying the design of an agent can lead to performance improvements, and together they provide an exciting avenue for future research. As with the techniques presented in Section 2, both approaches look at modifying certain parameters on a mostly fixed robot. To this end, the overall configuration and design of the robots remained the same. Future work should explore the possibility for true design in addition to parameter optimisation. For instance, the arrangement, orientation and geometry of components, including the base or links on the space manipulator. The method could even be advanced to select and locate sensors and motors in a manner that improves performance and efficiency. The implementation of such a technique is likely to lead to the discovery of completely new and innovative designs, potentially leading to improvements in performance of orders of magnitude.

As for the use of RL in the space robotics community there still exists a large technology gap. While some of these are hardware dependent *e.g.* the radiation hardening of chips used on satellites, issues also arise at the validation and testing stage. A promising area for future research is to look at ways to implement RL control techniques external to NNs. Not only would this reduce computational demand, but the use of more easily understandable systems would increase the confidence that control schemes would perform the same on-orbit as on Earth. The introduction of such techniques along with more powerful space-qualified computers may even allow for on-board learning.

The culmination of the work in this thesis concludes that a number of different space robots are required to achieve the OOA of a large aperture space telescope. However, work at this stage has treated each as an individual entity operating in series with the other agents. A pivotal

area of research for pushing this mission concept forwards is investigating how heterogeneous, potentially multi-arm, space robots could operate collaboratively to achieve missions. Furthermore, such a system could be combined with ORCHID in a manner that allows agent design to occur simultaneously across multiple agents. The result would be a fully operational and optimal mission concept for OOA. This could be extended to a robust framework that could be used to optimise mission concepts for other OOO tasks.

In summary, the work carried out in this thesis has opened up a number of new research directions for furthering the field of space robotics and RL. It is hoped that these contributions and conclusions will aid in the ultimate goal of a space robot fully autonomously assembling large structures in space.

Appendix A

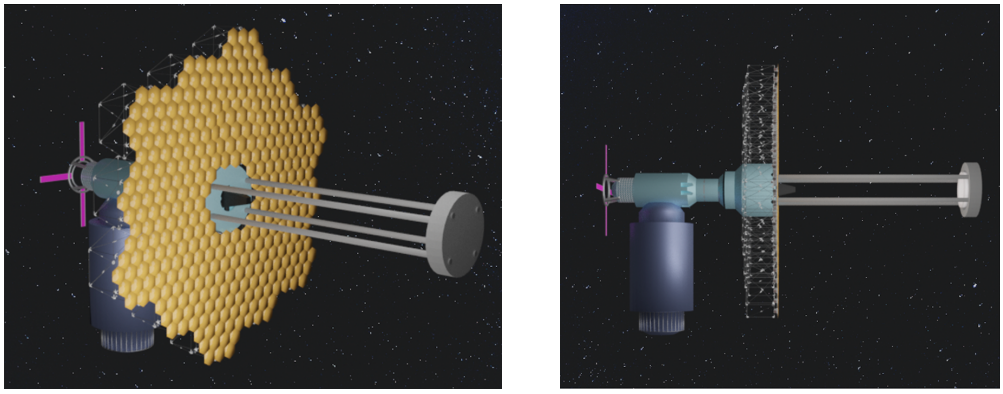
Telescope Architecture

A.1 Introduction

This section outlines the design of a large aperture space telescope for OOA with a free-flying robotic spacecraft. A detailed science case behind design choices is not presented and instead the mechanical design is the focus of this section. The telescope is a Ritchey-Chretien telescope, with a PM of $25\,\mathrm{m}$ in diameter and a spatial resolution of $1\,\mathrm{m}$ from GEO. The scientific validation of the PM and SM sizes have been discussed and validated by Nanjangud *et al.* [106]. The design aims to utilise as much technology already in-orbit or planned for use due to the high TRL, making testing and verification a less expensive and arduous task. In addition to this, design choices are made to minimise costs and mission complexity.

A.2 Optical Space Telescopes

There are three main types of telescope that can image in the optical range. The first are refractor telescopes, which rely on lenses to focus light. Reflector telescopes rely on mirrors, and catadioptrics use a combination of mirrors and lenses to focus light. While all types have contributed to great scientific discoveries, reflector telescopes are favored in space applications [106]. This is because they do not suffer from chromatic aberrations at apertures over $1\,\mathrm{m}$ [124].



(a) Isometric view of full telescope assembly (b) Side view of full telescope assembly

Figure A.1: **Artist impression of a large aperture telescope.** This shows an illustration of what a large aperture telescope that has been assembly on-orbit may look like. Visible is the storage satellite, PM, SM and modular BP. At the rear a number of deployable solar arrays can also be seen.

Figure A.1 shows an artist's impression of the space telescope. There are three main parts that will require assembly or deployment:

1. **Segmented PM assembly**: The PM or aperture is the main light gathering surface on a reflective telescope. Since resolution is dependent on how much light the telescope can collect, the size of this mirror plays a large part in the overall performance of the telescope.

2. **Modular BP**: This is the structure that supports the PM modules and acts as the contact between the PM and the main telescope housing.

3. **SM**: A smaller mirror that is used to redirect and refocus the light reflected from the PM.

While a functional space telescope will require more components than just those named above, focus is on these components since they will require robotic assembly/deployment. Other parts such as the AOCS, communication unit and power subsystems will be developed at a later date when a systems design approach is taken. Since the size of the satellite bus does not affect the telescope architecture, it is assumed that suitable subsystems will be available and easily integrated into the final design.

Table A.1: **Comparison of available standard interfaces.** Comparison of the standard interfaces suitable for use with robotic OOA [80, 168, 72].

		iSSI	SIROM	HOTDOCK (Active)
Size	Diameter	199 mm	120 mm	128 mm
	Height	48 mm	30 mm	70 mm
	Mass	0.9 kg	1.5 kg	1.4 kg
Mechanical Loads	Axial	6000 N	200 N	3000N
	Lateral	400 N	200 N	n/a
	Bending	400 N m	40 N m	300 N m
	Torque	400 N m	40 N m	n/a
Transfer Rates	Power	5 kW/100 V	120 W/100 V	2.5 kW/120 V
	Thermal	5 K W	2 to 2.5 K W	2.5 K W
	Data	1 Gbit s^{-1}	100 Mbit s^{-1}	n/a
Temperature Range		−50 to 70 °C	−128 to 50 °C	−40 to 70 °C
TRL		6	4	4
Capture		Symmetrical about 90°	Not symmetrical for capture	Symmetrical about 90°
Alignment Tolerance	Axial	3 mm	10 mm	15 mm
	All Axis	15°	1.5°	10°

A.3 Standard Interface

The implementation of OOA missions will be reliant on cooperative design and plug-and-play principles. This requires modular designs and standardised connection interfaces, two ideas that have gained a lot of traction in recent years. There are currently three standard interfaces in production aimed for use in space, information on each can be found in Table A.1.

The iSSI interface is selected for use in this design. This choice was made since the interface can withstand the highest mechanical loads and has the highest transfer rates. On top of this, it is the most developed of all the systems. The drawback is that it has more precise alignment

tolerances for capture compared to SIROM and HOTDOCK. However, the symmetry of the capture mechanism about 90° means docking will be significantly easier. The choice of this interface also limits design complexity since it is androgynous and hard capture is automatic once the systems are aligned.

The space robot will host an end-effector of the same interface, meaning it can easily capture/grasp any other module. These will be mounted in strategic locations to facilitate the capture, placement and connection of all the modules required for telescope operation. This same interface will be used to connect the different telescope parts to each other. This means that each component will require at least two interfaces, one for manipulation and one for assembly.

A.4 Primary Mirror

The fully assembled PM will be $25\,\mathrm{m}$ in diameter, constructed of 342 of identical hexagonal segments that measure $1\,\mathrm{m}$, flat edge to flat edge [106]. A segmented design is necessary since monolithic mirrors can only be manufactured to a certain size. In addition to this, large mirrors must survive the harsh launch environment, which is dependent on their stiffness, which decreases with diameter [109]. The segments are arranged in a honeycomb design, the same as that utilised by the JWST and all other modular telescope designs proposed in the literature. The advantage is a symmetrical design where the segments interface with as little wasted space as possible, resulting in a fully filled PM.

In order to minimise the number of pick-and-place operations needed for the full mirror assembly, sub-assemblies of the $1\,\mathrm{m}$ segments are to be used. This approach makes the phasing and aligning of the mirrors easier since each $1\,\mathrm{m}$ segment in the sub-assembly will have its own actuators, therefore requiring less force to manipulate than if each sub-assembly had only one set of actuators. When deciding on the geometry of these sub-assemblies a number of things were taken into account:

- The overall number of pick and place operations that will be required for the full PM assembly. Fewer operations will lower mission complexity.
- The number of mirrors in each sub-assembly. In the case of failure the whole module will be replaced so collateral wastage should be limited.

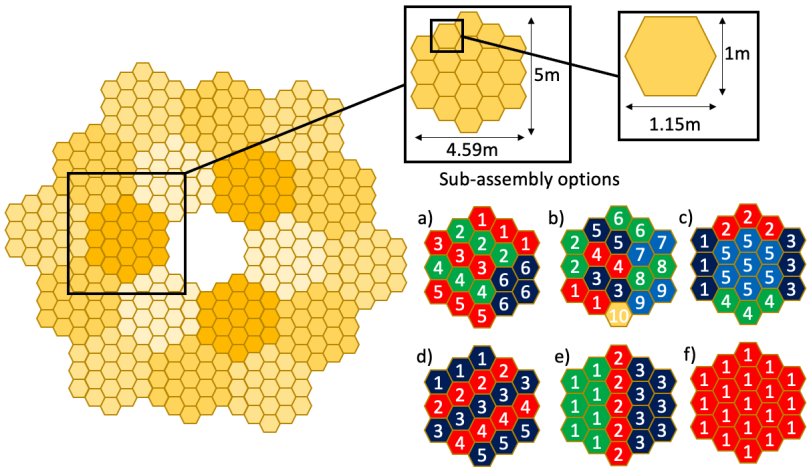


Figure A.2: **Primary mirror sub-assemblies.** The left image shows the full PM set up. On the right, labelled with letters, are the different combinations of segment assemblies that could be used to make the full structure.

- The mass of each sub-assembly. This will govern the overall mass of the PM.
- The size of the sub-assembly in relation to launch vehicle fairings.
- The shape of each sub-assembly. This should be as consistent as possible to limit complexity in both the assembly and servicing missions.

A number of configurations were considered, all are shown in Figure A.2 with a comparison in Table A.2. The PM sub-assemblies relate to a hexagonal combination of 19 segments. The full mirror is then made of 18 of these larger sections. This split was chosen since it maintains the hexagonal shape and design symmetry.

Designs **d**, **e** and **f** were discounted due to their large maximum dimension. This requires a larger manipulator during assembly, and it limits the launch vehicles that can be utilised. At a maximum dimension of over $4.6\,\mathrm{m}$ (internal fairing diameter of the Falcon-9), these pieces would depend on specific launch vehicles with lower launch frequencies adding huge cost and lowering mission flexibility. The novel control and planning method taken in this work means that once learned, the assembly process becomes a well-defined repeatable operation. As a result, the number of pick and place operations required is not considered to be the most important of the trade-off factors. Therefore, the selected sub-assembly geometry is option **a**. This geometry

Table A.2: **Mirror subassembly geometry comparions.** The different PM sub-assembly combinations are presented here. A comparison of their desired attributes is given.

	Additional mass from connectors (kg)	Number of pick and place operations	Number of different shaped sub-assemblies	Maximum dimension (m)	Maximum number of mirrors in one sub-assembly
a	10.8	108	2	3	3
b	18	180	2	2	2
c	9	90	3	3	7
d	9	90	3	5	5
e	5.4	54	3	5	7
f	1.8	18	1	5	19

will result in minimal wastage if a single mirror is damaged compared to **c** and hosts a lower overall mass than option **b**. Complementary to this is the lower design complexity with only two different geometries being required. This aids in minimising the complexity of servicing missions.

In order for these segments to act as one, they must be co-aligned, co-focused and co-phased. This is achieved using a number of actuators and rods attached to the rear of each segment. This allows for flexing of the mirror to facilitate alignment once assembled. In addition, each mirror will be equipped with edge sensors, so its exact location is known — the same technology is used on the JWST [85]. In order to minimise complexity each mirror segment is mounted on a thin rear cover. One of the main reasons for this is that the alignment system can be tested and installed on the ground and then the robotic manipulator can interface with a standard connector on the rear cover. Figure A.3 shows the PM sub-assembly. The perpendicular facing iSSI module is for robotic grasping and the top facing module is for connection with the BP. This set-up means that the alignment system is protected from the robotic manipulator during assembly, limiting the chance of damage and generation of space debris.

Employing the same material choices made by Nanjangud *et al.* for the PM, each $1\,\mathrm{m}$ segment with wave sensing and control hardware is estimated to weigh $17.3\,\mathrm{kg}$ [106]. Included in the design is the thin rear cover, iSSI support structure and the iSSI modules themselves. It is therefore estimated the 3 segment PM mirror sub-assembly (sub-assembly 1) weighs $57.49\,\mathrm{kg}$,

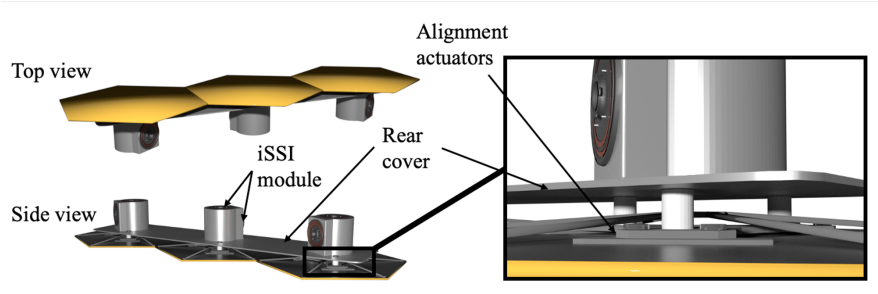


Figure A.3: **Different views of the primary mirror sub-assembly.** Three different views of the PM are visible. A top view, a side view and a zoomed in section. The purpose of the last image is to show the placement of the alignment actuators.

Table A.3: **Mass of primary mirror sub-assemblies.** Break-down of mass of the two different sub-assemblies used in with the PM

	Sub-Assembly 1			Sub-Assembly 2		
Component Description	Mass (kg)	Quantity	Total Mass (kg)	Mass (kg)	Quantity	Total Mass (kg)
PM segment and wave front control sensing	17.3	3	51.9	17.3	4	69.2
iSSI module	0.9	6	5.4	0.9	8	7.2
Rear cover	0.02	1	0.02	0.01	1	0.01
iSSI support column	0.06	3	0.17	0.06	4	0.24
TOTAL			**57.69**			**76.65**

and the 4 mirror sub-assembly (sub-assembly 2) weighs $76.75\,\mathrm{kg}$. A breakdown of this can be seen in Table A.3.

A.5 Modular Back-Plane

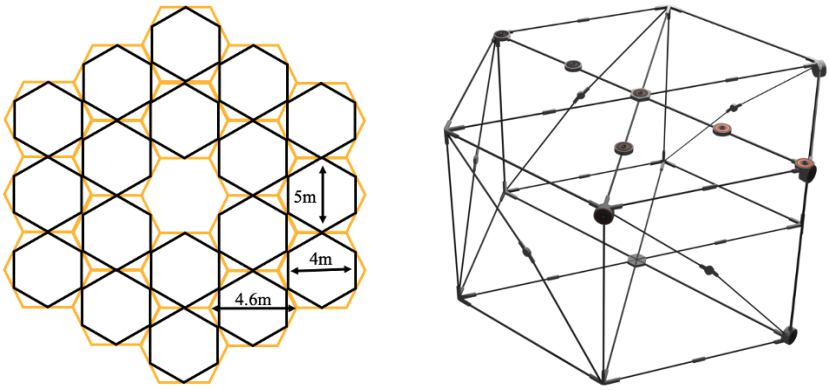
In order for the PM sub-assemblies to form a single filled mirror and attach to the main telescope hub, a rigid BP is needed. The same hexagonal modular design is desired, meaning one PM sub-assembly can interface with one BP module. Unlike with the PM, the BP does not have a given volume/surface area that must be solid. This provides huge scope for stowed volumetric minimisation. Two main solutions exist to such a problem. The first is to assemble the BP from

struts and nodes, and the second is to use quasi-rigid members connected by mechanical joints that auto-deploy. The first idea is disregarded due to the additional complexity in mission design and increased burden on the robotic agent. A design exercise carried out by Doggett, showed that to make an $8m$ truss structure from nodes and struts would take upwards of 20 hours using many pick and place operations [43]. Instead a deployable structure will be used.

A sparse tessellation configuration of the BP modules is favorable since it leads to lower mass and lower design complexity compared to a filled tessellation. It also allows room for the robot to access the rear of the telescope's PM when assembled. With this configuration the dimensions of the BP modules differ from that of the PM sub-assemblies, instead they measure $4m$ flat-to-flat. A visualisation of how the PM and BP interface is shown in Figure A.4a. Similar to all the telescope designs in the literature this work utilises the PacTruss modules developed by NASA [56]. Following the convention of this truss structure, the depth of each BP module is half the desired flat-to-flat length, in this case, $2m$. An issue not addressed in prior work is how these modules will connect to each other to form a rigid BP. The design in this thesis uses iSSI modules in various locations on each module: two will be used for inter-module connection, one for connection to the main telescope/inner ring and six for connections to each of the PM components. A single BP module with iSSI modules can be seen in Figure A.4b. At the connection points the iSSI module is housed in a special unit that interfaces with the Pactruss module and aids in locking its joints once deployed. It should be noted that the BP modules sit in two rings, with the inner ring connecting directly to the main telescope hub and the second, larger ring, connected only to other BP modules. According to Nanjangud *et al.* , a single truss module without connectors will weigh around $4.2kg$, it can therefore be estimated that each sub-assembly including all other components to weigh $13.2kg$ [106]. This is given that each iSSI and housing unit weighs $1kg$.

A.6 Secondary Mirror

The SM is a $2.4m$ hyperbolic mirror that sits $4.55m$ away from the primary mirror. It will be pre-assembled on Earth so the task here is to get it to the correct distance from the PM, and then fix it rigidly. The accuracy of placement of this mirror is paramount to the successful operation of the telescope in-orbit. In addition to this, along with all other parts of the telescope it should be serviceable by the space robot. As a result the sub-assembly will host an iSSI interface,



(a) Geometry of PM and BP modules

(b) A single fully deployed BP module with iSSI interface

Figure A.4: **Dimensions and structure of back-plane.** Image shows a comparison between the dimensions of PM module and a BP module as well as how they line up in the full assembly. Note that each PM module is made up of the selected sub-assembly geometry.

mounted on the rear of the module. The JWST uses a hinged mechanism to deploy the SM but this requires sufficient space in a launch vehicle for the full length of the mast, in this case $4.55m$. This can be avoided by instead using a deployable mast.

In order to achieve the desired accuracy the SM must be fixed by six constraints, 1 for each d.o.f.. Therefore a triangular truss will be used – this will be fixed at the center of the PM surrounding the optics and will be the final component of the telescope to be deployed. Sensors mounted in the SM module will measure its exact position relative to the PM. Deployment will happen automatically without the aid of the space robot.

Bibliography

[1] Robotic manipulation and capture in space: A survey. *Frontiers in Robotics and AI*, 8, 2021.

[2] Ajith Abraham and Lakhmi C. Jain. Evolutionary multiobjective optimisation. In *Evolutionary Multiobjective Optimisation Theoretical Advances and Applications*, pages 1–6. 2006.

[3] Hatem Al-Dois, A. K. Jha, and R. B. Mishra. Task-based design optimization of serial robot manipulators. *Engineering Optimization*, 45(6):647–658, August 2012.

[4] S. F. Alyaqout, P. Y. Papalambros, and A. G. Ulsoy. Combined robust design and robust control of an electric dc motor. *IEEE/ASME Transactions on Mechatronics*, 16(3):574–582, 2011.

[5] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[6] Timothée Anne, Jack Wilkinson, and Zhibin Li. Meta-learning for fast adaptive locomotion with uncertainties in environments and robot dynamics. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4568–4575, 2021.

[7] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[8] Randy Attwood. MOST: Canada's first space telescope part 2. *Journal of the Royal Astronomical Society of Canada*, 97:7, 2003.

[9] J E Auerbach and J C Bongard. Dynamic resolution in the co-evolution of morphology and control. In *In Proc. of the Twelfth International Conference on Artificial Life (ALIFE XII)*, 2010.

[10] Dave Baiocchi and Philip H. Stahl. Enabling future space telescopes: mirror technology review and development roadmap. *The Astronomy and Astrophysics Decadal Survey*, (23), 2009.

[11] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutckever, and Igor Mordatch. Emergent complexity via multi-agent competition. In *Proc. 6^{th} International Conference on Learning Representations (ICLR)*, Vancouver, BC, 2018.

[12] Johathan T. Barron. Continuously differentiable exponential linear units. *CoRR*, abs/1704.07483v1, 2017.

[13] Santanu Basu. Conceptual design of an autonomously assembled space telescope (AAST). In Howard A. MacEwen, editor, *UV/Optical/IR Space Telescopes: Innovative Technologies and Concepts*, volume 5166, pages 98 – 112. International Society for Optics and Photonics, SPIE, 2004.

[14] Santanu Basu, Terry S. Mast, and Gary T. Miyata. A proposed autonomously assembled space telescope (AAST). *Space*, 2003.

[15] Simone Battistini, Chantal Cappelletti, and Filippo Graziani. An attitude determination and control system for a nano-satellite alternative launch platform. In *Proceedings of the 67^{th} International Astronautical Congress*, 26^{th} - 30^{th} Sep, 2016. Guadalajara, Mexico.

[16] Saul Blumenthal. Multinomial sampling with partially categorized data. *Journal of the American Statistical Association*, 63(322):542–551, 1968.

[17] Vladimir Igorevich Bogachev and Maria Aparecida Soares Ruas. *Measure theory*, volume 1. Springer, 2007.

[18] Adrien Bolland, Ioannis Boukas, Mathias Berger, and Damien Ernst. Jointly learning environments and control policies with projected stochastic gradient ascent. *Journal of Artificial Intelligence Research*, 73, 2022.

[19] William J. Borucki, David Koch, Gibor Basri, Natalie Batalha, Timothy Brown, Douglas Caldwell, John Caldwell, Jørgen Christensen-Dalsgaard, William D. Cochran, Edna DeVore, Edward W. Dunham, Andrea K. Dupree, Thomas N. Gautier, John C. Geary, Ronald Gilliland, Alan Gould, Steve B. Howell, Jon M. Jenkins, Yoji Kondo, David W. Latham, Geoffrey W. Marcy, Søren Meibom, Hans Kjeldsen, Jack J. Lissauer, David G. Monet, David Morrison, Dimitar Sasselov, Jill Tarter, Alan Boss, Don Brownlee, Toby Owen, Derek Buzasi, David Charbonneau, Laurance Doyle, Jonathan Fortney, Eric B. Ford, Matthew J. Holman, Sara Seager, Jason H. Steffen, William F. Welsh, Jason Rowe, Howard Anderson, Lars Buchhave, David Ciardi, Lucianne Walkowicz, William Sherry, Elliott Horch, Howard Isaacson, Mark E. Everett, Debra Fischer, Guillermo Torres, John Asher Johnson, Michael Endl, Phillip MacQueen, Stephen T. Bryson, Jessie Dotson, Michael Haas, Jeffrey Kolodziejczak, Jeffrey Van Cleve, Hema Chandrasekaran, Joseph D. Twicken, Elisa V. Quintana, Bruce D. Clarke, Christopher Allen, Jie Li, Haley Wu, Peter Tenenbaum, Ekaterina Verner, Frederick Bruhweiler, Jason Barnes, and Andrej Prsa. Kepler planet-detection mission: Introduction and first results. *Science*, 327(5968):977–980, 2010.

[20] R. Boumans and C. Heemskerk. The european robotic arm for the international space station. *Robotics and Autonomous Systems*, 23(1-2):17–27, March 1998.

[21] Lynn M. Bowman, W. Keith Belvin, Erik E. Komendera, John T. Dorsey, and Bill R. Doggett. In-space assembly application and technology for NASA's future science observatory and platform missions. In *Proceedings of SPIE Astronomical Telescopes and Instrumentation*, Austin, Texas. 12-14, June 2018.

[22] Bradford. Reaction wheel unit datasheet. URL, http://bradford-space.com/products-aocs-reaction-wheel-unit.php. Date Accessed 22/08/2019.

[23] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, abs/1606.01540, 2016.

[24] Christopher J. Burrows, Jon A. Holtzman, S. M. Faber, Pierre Y. Bely, Hashima Hasan, C. R. Lynds, and Daniel Schroeder. The Imaging Performance of the Hubble Space Telescope. *Astrophysical Journal Letters*, 369:L21, March 1991.

[25] Luca Carlone and Carlo Pinciroli. Robot co-design: Beyond the monotone case. In *International Conference on Robotics and Automation (ICRA)*, Montreal, Canada. 20-24, May 2019.

[26] Kieran A. carroll, Robert E. Zee, and Jaymie Matthews. The MOST microsatellite mission: Canada's first space telescope. In *Proceedings of the 12th AIAA/USU Conference on Small Satellites*, 1998.

[27] Kuang-Hua Chang. *e-Design Computer-Aided Engineering Design*, chapter 19, pages 1105–1173. Academic Press, 2015.

[28] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–897, Montreal, QC, 2019.

[29] Tao Chen, Adithyavairavan Murali, and Abhinav Gupta. Hardware conditioned policies for multi-robot transfer learning. *CoRR*, abs/1811.09864, 2018.

[30] Tianjian Chen, Zhanpeng He, and Matei Ciocarlie. Hardware as policy: Mechanical and computationalco-optimization using deep reinforcement learning. In *Conference on Robotic Learning (CoRL)*, 2020.

[31] Xuanzhen Chen and Shiyin Qin. Kinematic modeling for a class of free-floating space robot. *IEEE Access*, 5:12389–12403, 2017.

[32] G. Chiandussi, M. Codegone, S. Ferrero, and F.E. Varesio. Comparison of multi-objective optimization methodologies for engineering applications. *Computers & Mathematics with Applications*, 63(5):912–942, 2012.

[33] Carlos A. Coello Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, 1(3):269–308, 1999.

[34] Carlos A. Coello Coello. Twenty years of evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, 1(1):28–36, 2006.

[35] S Dahbi, A Aziz, S Zouggar, N Benazzi, H Zahboune, and M Elhfyani. Design and sizing of electrical power source for a nanosatellite using photovoltaic cells. In *Proceedings of the 3rd IEEE International Renewable Sustainable Energy Conference*, 10th-13th Dec, December 2015. MarrakechOuarzazate, Morocco.

[36] Tianhong Dai, Kai Arulkumaran, Tamara Gerbert, Samyakh Tukra, Feryal Behbahani, and Anil Anthony Bharath. Analysing deep reinforcement learning agents trained with domain randomisation. *Neurocomputing*, 493:143–165, 2022.

[37] Indraneel Das and J. E. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998.

[38] Filipe de A. Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J. Zico Kolter. End-to-end differentiable physics for learning and control. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, 2018.

[39] Thomas J. Debus and Sean P. Dougherty. Overview and performance of the front-end robotics enabling near-term demonstration (FREND) robotic arm. In *Proceedings of AIAA Infotech @Aerospace Conference*, Seattle, Washington. 6-9, April 2009.

[40] Jonas Degrave, Michiel Hermans, Joni Dambre, and Francis wyffels. A differentiable physics engine for deep learning in robotics. *Frontiers in Neurorobotics*, 13, 2019.

[41] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2169–2176, Marina Bay Sands, Singapore, 2017.

[42] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2016.

[43] William Doggett. Robotic assembly of truss structures for space systems and future research plans. *IEEE Aerospace Conference Proceedings*, 7:3589–3598, 2002.

[44] William R. Doggett, John T. Dorsey, Thomas C. Jones, and Bruce King. Development of a tendon-actuated lightweight in-space MANipulator (TALISMAN). In *Proceedings*

of 42nd Aerospace Mechanisms Symposium, volume 405, NASA Goddard Space Flight Center. 14-16, May 2014.

[45] Tim W. Dorn, Anthony G. Schache, and Marcus G. Pandy. Muscular strategy shift in human running: Dependence of running speed on hip and ankle muscle performance. *Journal of Experimental Biology*, 1(215):1944–1956, July 2012.

[46] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbell. rl^2 : fast reinforcment learning via slow reinforcement learning. In *Proc. 5th International Conference on Learning Representations*, Toulon, France. 24-26, April 2017.

[47] Michael T. M. Emmerich and André H. Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing*, 17(3):585–609, 2018.

[48] Lee D. Feinberg, Jason G. Budinoff, Howard A. MacEwen, Gary W. Matthews, and Marc Postman. Modular assembled space telescope. *Optical Engineering*, (9):1 – 9, 2013.

[49] C. Finn, P. Abbel, , and S. Levine. Model-agnostic meta-learning for fast adapation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1126–1135, Sydney, NSW, Australia, 2017.

[50] Angel Flores-Abad, Ou Ma, Khanh Pham, and Steve Ulrich. A review of space robotics technologies for on-orbit servicing. *Progress in Aerospace Sciences*, 68:1–26, July 2014.

[51] Robert B. Friend. Orbital express program summary and mission overview. In *Proceedings of SPIE Defense and Security Symposium*, volume 6958, Orlando, Florida, USA. 15, April 2008.

[52] Jonathan P. Gardner, John C. Mather, Mark Clampin, Rene Doyon, Matthew A. Greenhouse, Heidi B. Hammel, John B. Hutchings, Peter Jakobsen, Simon J. Lilly, Knox S. Long, Jonathan I. Lunine, Mark J. Mccaughrean, Matt Mountain, John Nella, George H. Rieke, Marcia J. Rieke, Hans-Walter Rix, Eric P. Smith, George Sonneborn, Massimo Stiavelli, H. S. Stockman, Rogier A. Windhorst, and Gillian S. Wright. The james webb space telescope. *Space Science Reviews*, (123):485–606, 2006.

[53] David Ha. Reinforcement learning for improving agent design. *Neural Information Processing Systems Reinforcement Learning Workshop*, 2018.

[54] Zhou Hao, Nikos Mavrakis, Pedro Proenca, Richard Gillham Darnley, Saber Fallah, Martin Sweeting, and Yang Gao. Ground-based high-DOF AI and robotics demonstrator for in-orbit space optical telescope assembly. In *70^{th} International Astronautical Congress (IAC)*, Washington D.C., United States, 21/10/2019-25-10-2019, 2019.

[55] Zhou Hao, R. B. Ashith Shyam, Arunkumar Rathinam, and Yang Gao. Intelligent spacecraft visual GNC architecture with the start-of-the-art components for on-orbit manipulation. *Frontiers in Robotics and AI*, 8(639327):65, 2021.

[56] John M. Hedgepeth and Richard K. Miller. Structural concepts for large solar concentrators. *NASA Contractor Report 4075*, 1962.

[57] Honeybee. Microsat CMG attitude control array datasheet. URL, https://honeybeerobotics.com/wp-content/uploads/2014/03/Honeybee-Robotics-Microsat-CMGs.pdf. Date Accessed 22/08/2019.

[58] G.S. Hornby, H. Lipson, and J.B. Pollack. Generative representations for the automated design of modular physical robots. *IEEE Transactions on Robotics and Automation*, 19(4):703–719, 2003.

[59] Kirk Hovell and Steve Ulrich. On deep reinforcement learning for spacecraft guidance. In *AIAA Scitech 2020 Forum*, Orlando, Florida, 06/01/2020-10/01/2020, 2020.

[60] Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B. Tenenbaum, William T. Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6265–6271, 2019.

[61] Donald J. Henja III, Pieter Abbeel, and Lerrel Pinto. Task-agnostic morphology evolution. In *International Conference on Learning Representations*, 2021.

[62] Lucy Jackson, Chakravarthini M. Saaj, Asma Seddaoui, Calem Whiting, Steve Eckersley, and Mark Ferris. Design of a small space robot for on-orbit assembly missions. In *Proceedings of the 5th International Conference on Mechatronics and Robotics Engineering (ICMRE)*, pages 107–112, 16^{th}-19^{th} February, 2019. Rome, Italy.

[63] Lucy Jackson, Chakravarthini M. Saaj, Asma Seddaoui, Calem Whiting, Steve Eckersley, and Simon Hadfield. Downsizing an orbital space robot: A dynamic system based evaluation. *Advances in Space Research*, 65(10):2247–2262, May 2020.

[64] Lucy Jackson, Celyn Walters, Steve Eckersley, and Simon Hadfield. HARL-A: Hardware agnostic reinforcement learning through adversarial selection. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3499–3505, 27^{th} September - 1^{st} October, 2021. Virtual.

[65] Lucy Jackson, Celyn Walters, Steve Eckersley, Pete Senior, and Simon Hadfield. ORCHID: Optimisation of robotic control and hardware in design using reinforcement learning. In *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4911–4917, 27^{th} September - 1^{st} October, 2021. Virtual.

[66] Lucy Jackson, Celyn Walters, Chakravarthini M. Rai, Steve Eckersley, and Simon Hadfield. Ta-DAH: Task driven automated hardware design of free-flying space robots. In *Proceedings of the 16^{th} International Conference on Space Robotics and Automation (ICSRA)*, 19^{th} - 20^{th} July, 2022. Virtual.

[67] Deb Kalyanmoy. *Search Methodologies*, chapter 10, pages 403–449. Springer US, 2014.

[68] Oussama Khatib, Xiyang Yeh, Gerald Brantner, Brian Soe, Boyeon Kim, Shameek Ganguly, Hannah Stuart, Shiquan Wang, Mark Cutkosky, Aaron Edsinger, Phillip Mullins, Mitchell Barham, Christian R. Voolstra, Khaled Nabil Salama, Michel L'Hour, and Vincent Creuze. Ocean one: A robotic avatar for oceanic discovery. *IEEE Robotics Automation Magazine*, 23(4):20–29, 2016.

[69] Jin-Oh Kim and Pradeep K. Khosla. A formulation for task based design of robot manipulators. In *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2310–2317, 1993.

[70] Tuomo Kivela, Jouni Mattila, and Jussi Puura. A generic method to optimize a redundant serial robotic manipulator's structure. *Automation in Construction*, 81:172–179, 2017.

[71] Ivan Kobyzev, Simon Prince, and Marcus Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 05 2020.

[72] Joerg Kreisel, Thomas A. Schervan, and Kai-Uwe Schroeder. A game-changing space system interface enabling multiple modular and building block-based architectures for orbital and exploration missions. In *Proceedings of the 70^{th} International Astronautical Congress (IAC)*, 21^{st} - 25^{th} October, 2019. Washington D.C., United States,.

[73] S. Kucuk and Z. Bingul. Robot workspace optimization based on a novel local and global performance indices. In *Proceedings of of the IEEE Symposium on Industrial Electronics*, Dubrovnik, Croatia. 20-23, June 2005.

[74] Serdar Kucuk and Zafer Bingul. Comparative study of performance indices for fundamental robot manipulators. *Robotics and Autonomous Systems*, (54):567–573, 2006.

[75] A. Kumar and K. J. Waldron. The workspaces of a mechanical manipulator. *Journal of Mechanical Design*, 103(3):665–672, 07 1981.

[76] Jet Propulsion Laboratory. Mars 2020 perseverance landing press kit. Technical report, NASA, January 2020.

[77] Jeffrey C Lagarias, James A Reeds, Margaret H Wright, and Paul E Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, 9(1):112–147, 1998.

[78] Nicolas Lee, Paul Backes, Joel Burdick, Sergio Pellegrino, Christine Fuller, Kristina Hogstrom, Brett Kenedy, Junggon Kim, Rudranarayan Mukherjee, Carl Seubert, and Yen-Hung Wu. Architecture for in-space robotic assembly of a modular space telescope. *Journal of Astronomical Telescopes, Instruments and Systems*, 2(24), 2016.

[79] Floor Van Leeuwen. The hipparcos mission. *Space Science Reviews*, 81(201-409), 1997.

[80] Pierre Letier, Torsten Siedel, Mathieu Deremetz, Edgars Pavlovskis, Benoit Lietaer, Korbinian Nottensteiner, Maximo A. Roa, Juan Sánchez García Casarrubios, Javier Luis Corella Romero, and Jeremi Gancet. Hotdock: Design and validation of a new generation of standard robotic interface for on-orbit servicing. In *Proceedints of the 71^{st}*

International Astronautical Congress , The CyberSpace Edition, 12^{th}-14^{th} October, 2020. Virtual.

[81] Q. Li, W.J. Zhang, and L. Chen. Design for control-a concurrent engineering approach for mechatronic systems design. *IEEE/ASME Transactions on Mechatronics*, 6(2):161–169, 2001.

[82] RuiQin Li and Jian S Dai. Orientation angle workspaces of planar serial three-link manipulators. *Science in China Series E: Technological Sciences*, 52:975–985, 2009.

[83] Yuxi Li. Deep reinforcement learning: An overiview. *arXiv*, abs/1701.07274, 2017.

[84] Junband Liang and Ming Lin. Differentiable physics simulation. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differenttial Equations*, 2020.

[85] Paul A. Lightsey, Charles Atkinson, Mark Clampin, and Lee D. Feinberg. James webb space telescope: large deployable cryogenic telescope in space. *Optical Engineering*, 51(1), 2012.

[86] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, 7^{th}-9^{th} May, 2015. San Juan, Puerto Rico.

[87] Ilya Loshchilov and Frank Hutter. CMA-ES for hyperparameter optimization of deep neural networks. *CoRR*, abs/1604.07269, 2016.

[88] Kevin Sebastian Luck, Heni Ben Amor, and Roberto Calandra. Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. In *Proceedings of the Conference on Robot Learning, PMLR*, volume 100, pages 854–869, October 2020.

[89] Jens Lundell, Murtaza Hazara, and Ville Kyrki. Generalizing movement primitives to new situations. In *Proc. 18th Towards Autonomous Robotic Systems*, pages 16–31, Guildford, England. 19-21, July 2017. Springer International Publishing.

[90] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Adversarially robust policy learning: Active construction of physically-plausible perturbations. In *2017*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3932–3939, 2017.

[91] Marc Manz, Sebastian Bartsch, Romain Caujolle, Torsten Vogel, Mark Shielton, Elie Allouis, Stefan Gornig, Francisco Javier Colmenero, Sebastian Torralbo, Marko Jankovic, Wiebke Brinkmann, Isabel Soto, Gonzalo Guerra, Daniel Silveira, Serra Carolina, Björn Ordoubadian, Eric Bertels, Jeremi Gancet, Pierre Letier, Manfred Doermer, and Stéphane Estable. Robotic architecture and operational concept for in-space assembly and servicing missions. In *16^{th} Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2022)*, ESTEC Noordwijk, The Netherlands. 1–2, June 2022.

[92] R. Timothy Marler and Jasbir S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, 2010.

[93] R.T Marler and J.S Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, April 2004.

[94] A. Messac, A. Ismail-Yahaya, and C. A. Mattson. The normalized normal constraint method for generating the pareto frontier. *Structural and Multidisciplinary Optimization*, (25):89–98, 2003.

[95] Kaisa Miettinen and Marko M. Mäkelä. On scalarizing functions in multiobjective optimization. *OR Spectrum*, 24(2):193–213, May 2002.

[96] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR.

[97] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis and. Human-level control through deep reinforcement learning. *Nature*, (518):529–533, 2015.

[98] Amr Mohamed, Chakravarthini Saaj, AsmaSeddaoui, and Manu Nair. Linear controllers for free-flying and controlled-floating space robots: a new perspective. *Aeronautics and Aerospace Open Access Journal*, 3(4), 2020.

[99] Lucian A. Montagnino. Test And Evaluation Of The Hubble Space Telescope 2.4-meter Primary Mirror. In Gregory M. Sanger, editor, *Large Optics Technology*, volume 0571, pages 182 – 190. International Society for Optics and Photonics, SPIE, 1986.

[100] Tom A Mulder. Orbital express autonomous rendezvous and capture flight operations. part 2 of 2. In *Proceedings of AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Honolulu, Hawaii. 18-21, August 2008.

[101] S. Nader Nabavi, Morteza Shariatee, Javad Enferadi, and Alireza Akbarzadeh. Parametric design and multi-objective optimization of a general 6-pus parallel manipulator. *Mechanism and Machine Theory*, 152:103913, 2020.

[102] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *CoRR*, arXiv:1803.11347, 2018.

[103] Muhammad Nagy, Yasser Mansour, and Sherif Abdelmohsen. Multi-objective optimization methods as a decision making strategy. *International Journal of Engineering Research and Technology (IJERT)*, 9(3), 2020.

[104] Manu H. Nair, Chakravarthini M. Saaj, Sam Adlen, Amir G. Esfahani, and Steve Eckersley. Advances in robotic in-orbit assembly of large-aperture space telescopes. In *i-SAIRAS*, Virtual Conference, 19/10/20-23/10/20, 2020.

[105] A. Nanjangud, P. Blacker, S. Bandyopadhyay, and Y. Gao. Robotics and AI-enabled on-orbit operations with future generation of small satellites. *Proceedings of the IEEE*, 106(3):429–439, February 2018.

[106] Angadh Nanjangud, Craig I. Underwood, Christopher P. Bridges, Chakravarthini M. Saaj, Steve Eckersley, Sir Martin Sweeting, and Paolo Bianco. Towards robotic on-orbit assembly of large space telescopes: Mission architectures, concepts and analyses. In *Proceedings of the 70^{th} International Astronautical Congress (IAC)*, pages 1–25, $216st$ - 25^{th} October, 2019. Washington D.C., United States.

[107] Nanoavionics. Reaction wheels system 'SatBus 4RWO' datasheet. URL, https://n-avionics.com/subsystems/cubesat-reaction-wheels-control-system-satbus-4rw/. Date Accessed 22/08/2019.

[108] John Nella, Paul D. Atcheson, Charles B. Atkinson, Doug Au, Allen J. Bronowicki, Ed Bujanda, Andy Cohen, Don Davies, Paul A. Lightsey, Richard Lynch, Ray Lundquist, Michael T. Menzel, Martin Mohan, John Pohner, Paul Reynolds, Henry Rivera, Scott C. Texter, David V. Shuckstes, Debra D. Fitzgerald Simmons, Robert C. Smith, Pamela C. Sullivan, Dean D. Waldie, and Rob Woods. James webb space telescope (jwst) observatory architecture and performance. *Optical, Infrared, and Millimeter Space Telescopes*, pages 576–587, 2004.

[109] Jerry Nelson. Segmented mirror telescopes. In Renaud Foy and Françoise Claude Fot, editors, *Optics in Astrophysics*, pages 61–72. Springer Netherlands, 2005.

[110] Tønnes F. Nygaard, Charles P. Martin, Eivind Samuelsen, Jim Torresen, and Kyrre Glette. Real-world evolution adapts robot morphology and control to hardware limitations. *In Proceedings of the Genetic and Evolutionary Computation Conference*, July 2018.

[111] M Oda. Space robot experiments on NASDA's ETS-VII satellite - preliminary overview of the experiment results. In *IEEE International Conference on Robotics and Automation*, volume 1-4, pages 1390–1395, Detroit, MI, USA. 10-15, May 1999.

[112] Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krahenbuhl, Vladlen Koltun, and Dawn Song. Assessing generalization in deep reinforcement learning, 2018. arXiv preprint arXiv:1810.12282,.

[113] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. 2019.

[114] J Park and H Asada. Concurrent design optimization of mechanical structure and control for high speed robots. In *1993 American Control Conference*, June 1993.

[115] Joon-Young Park, Pyung-Hun Chang, and Jeong-Yean Yang. Task-oriented design of robot kinematics using the grid method. *Advanced Robotics*, 17(9):879–907, 2003.

[116] A. P. Pashkevich and P.J. Flemming. A multiobjective optimisation approach to robotic manipulator design. In *IFAC Procedings Volumes*, volume 24, pages 387–392. July 1991.

[117] Sarosh Patel and Tarek Sobh. Manipulator performance measures - a comprehensive literature survey. *Journal of Intelligent and Robotic Systems*, 77(3-4):547–570, 2015.

[118] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *CoRR*, abs/1710.06537, 2017.

[119] Luis F. Penin, Kohtaro Matsumoto, and Sachiko Wakabayashi. Force reflection for ground control of space robots. *IEEE Robotics Automation Magazine*, 7:50–63, December 2000.

[120] M.A.C Perryman, E. Høg, J. Kovalevsky, L. Lindegren, C. Turon, P.L Bernacca, M. Crézé, R. Donati, M. Grenon, M. Grewing, R. van Leeuwen, H. van der Marel, C.. Murray, R.S. Le Poole, and H. Schrijver. In-orbit performance of the hipparcos astronomy satellite. *Astronomy and Astrophysics*, (258):1–6, 1992.

[121] Maciej Petko and Grzegorz Karpiel. Hardware/software co-design of control algorithms. In *2006 International Conference on Mechatronics and Automation*, pages 2156–2161, 2006.

[122] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2817–2826. PMLR, 06–11 Aug 2017.

[123] Li Qiao, Chris Rizos, and Andrew G. Dempster. Analysis and comparison of a cubesat lifetime. In *Proceedings of 13th Australian Space Conference*, pages 246 – 249, 30th Sep - 2nd Oct, 2013. Sydney, Australia.

[124] René Racine. The historical growth of telescope aperture. *The Publications of the Astronomical Society of the Pacific*, 116(815):77–83, 2004.

[125] A. Rajeswaran, Sarvjeet Ghotra, Sergey Levine, and Balaraman Ravindran. Epopt: Learning robust neural network policies using model ensembles. In *Proc. 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[126] Thambirajah Ravichandran, David Wang, and Glenn Heppler. Simultaneous plant-controller design optimization of a two-link planar manipulator. *Mechatronics*, 16(3):233–242, 2006.

[127] Benjamin B. Reed, Robert C. Smith, Bo Naasz, Joseph Pellegrino, and Charles Bacon. The restore-L servicing mission. In *Proceedings of AIAA Space Forum*, Long Beach, California. 13-16, September 2016.

[128] D. Reintsema, B. Sommer, T. Wolf, J. Theater, A. Radthke, J. Sommer, W. Naumann, and P. Rank. DEOS - the in-flight techonolgy demonstration of germans robotics approach to dispose of malfunctioned satellites. In *Proceedings of ESA Workshop on Advanced Space Technologies for Robotics and Automation.*, ESTEC Noordwijk, The Netherlands. 12 - 14, April 2011.

[129] Richard Rembala and Cameron Ower. Robotic assembly and maintenance of future space stations based on the iss mission operations experience. *Acta Astronautica*, 65:912–920, November 2009.

[130] Mark J. Rentmeesters, Wei K. Tsai, and Kwei-Jay Lin. A theory of lexicographic multi-criteria optimization. In *Proceedings of ICECCS '96: 2nd IEEE International Conference on Engineering of Complex Computer Systems (held jointly with 6th CSESAW and 4th IEEE RTAW)*, pages 76–79, 1996.

[131] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2015.

[132] Maximo Roa, Korbinian Nottensteiner, Armin Wedler, and Gerhard Grunwald. Robotic technologies for in-space assembly operations. In *Proceedings ESA 14th Symposium on Advanced Space Technologies In Robotics and Automation (ASTRA)*, pages 1 – 8, 20th - 22nd Jun, June 2017. Leiden, The Netherlands.

[133] Máximo A. Roa, Korbinian Nottensteiner, Gerhard Grunwald, Pablo Lopez Negro, Aurelian Cuffolo, Sabrina Andiappane, Mathieu Rognant, Antoine Verhaeghe, and Vincent Bissonnette. In-space robotic assembly of large telescopes. In *Proceedings of 15th Symposium on Advanced Space Technologies in Robotics and Automation*, Noordwijk, Netherlands, 27/05/2019-28/05/2019, 2019.

[134] RockwellCollins. RSI 12 momentum and reaction wheels datasheet. URL, https://www.rockwellcollins.com/Products-and-Services/Defense/Platforms/Space/RSI-12-Momentum-and-Reaction-Wheels.aspx. Date Accessed 22/08/2019.

[135] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

[136] Fereshteh Sadeghi and Sergey Levine. (cad)$ˆ2$rl: Real single-image flight without a single real image. *CoRR*, abs/1611.04201, 2016.

[137] Christian Sallaberger. Canadian space robotic activities. *Acta Astronautica*, 41(4):239 – 246, 1997.

[138] Mine Sarac, Massimiliano Solazzi, Edoargo Sotgiu, Massimo Bergamasco, and Antonio Frisoli. Design and kinematic optimization of a novel underactuated robotic hand exoskeleton. *Meccanica*, 52(3):749–761, 2017.

[139] C. Schaff, D. Yunis, A. Chakrabarti, and M. R. Walter. Jointly learning to construct and control agents using deep reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[140] Charles Schaff and Matthew R. Walter. N-limb: Neural limb optimization for efficient morphological design, 2022.

[141] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2015.

[142] Jürgen Schmidhuber, Jieyu Zhao, and Marco Wiering. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28(1):105–130, Jul 1997.

[143] Jürgen Schmidhuber and Rudolf Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(1), 1991.

[144] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

[145] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[146] A. Seddaoui and C M. Saaj. H_∞ controller for a controlled floating robotic spacecraft. In *Proceedings of 14th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 4^{th} - 6^{th} June, 2018. Madrid, Spain.

[147] Asma Seddaoui and Chakravarthini M. Saaj. The controlled floating motion of space robots using a nonlinear H_∞ controller. *AIAA Journal of Guidance, Control and Dynamics*, 42(8), 2019.

[148] Hiroaki Shioya, Yusuke Iwasawa, and Yitaka Matsuo. Extending robust adversarial reinforcement learning considering adaptation and diversity. *Proc. 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, 2018.

[149] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[150] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 387–395, Bejing, China, 22–24 Jun 2014. PMLR.

[151] Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, page 15–22, New York, NY, USA, 1994. Association for Computing Machinery.

[152] Ye. Somov, S. Butyrin, T. Somova, and S. Somov. Control of a free-flying robot at preparation for capturing a passive space vehicle. *IFAC-PapersOnLine*, 51(30):72–76, 2018. 18th IFAC Conference on Technology, Culture and International Stability TECIS 2018.

[153] Ivan P. Stanimirovic. Compendious lexicographic method for multi-objective optimization. *FACTA UNIVERSITATIS, Ser. Math. Inf.*, 27(1):55 – 66, 2012. Cited by: 11.

[154] L. Stocco, S.E. Salcudean, and F. Sassani. Matrix normalization for optimal robot design. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 2, pages 1346–1351 vol.2, 1998.

[155] Freek Stulp, Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. Learning to grasp under uncertainty. In *2011 IEEE International Conference on Robotics and Automation*, pages 5703–5708, 2011.

[156] Richard S. Sutton. *Reinforcement learning an introduction*. Adaptive computation and machine learning series. MIT Press, Cambridge, Mass, 1998.

[157] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *CoRR*, abs/1804.10332, 2018.

[158] Geraud Nangue Tasse, Steven James, and Benjamin Rosman. Generalisation in lifelong reinforcement learning through logical composition. In *International Conference on Learning Representations*, 2022.

[159] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, 2017.

[160] Olivier Toupet, Jeffrey Biesiadecki, Arturo Rankin, Amanda Steffy, Gareth Meirion-Griffith, Dan Levine, Maximilian Schadegg, and Mark Maimone. Terrain-adaptive wheel speed control on the curiosity mars rover: Algorithm and flight results. *Journal of Field Robotics*, 37(5):699–728, 2020.

[161] Steve Ulrich and Jurek Z Sadiadek. Modified simple adaptive control for a two-link space robot. In *Proceedings of the 2010 American Control Conference*, pages 3654–3659, 2010.

[162] Y. Umetani and K. Yoshida. Workspace and manipulability analysis of space manipulators. *Transactions of the Society of Instrument and Control Engineers*, 1(1):116–123, 2001.

[163] Tethers Unlimited. Kraken^{TM} robotic arm data sheet. Technical report, Tethers Unlimited, 2018.

[164] Vacco. CuSP propulsion system. URL, https://www.cubesat-propulsion.com/cusp-propulsion-system/. Date Accessed 27/08/2019.

[165] Vacco. AFRL propulsion unit for cubesats. URL, https://www.cubesat-propulsion.com/propulsion-unit/. Date Accessed 27/08/2019.

[166] Vacco. NASA C-POD micro cubesat propulsion system data sheet. URL, https://www.cubesat-propulsion.com/reaction-control-propulsion-module/. Date Accessed 23/08/2019.

[167] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016.

[168] Javier Vinals, Eduardo Urgoiti, Gonzalo Guerra, Ignacio Valiente, Judit Esnoz-Larraya, Michel Ilzkovitz, Mario Franceski, Pierre Letier, Xiu-Tian Yan, Gwenole Henry, Albino Quaranta, Wiebke Brinkmann, Marko Jankovic, Sevastian Bartsch, Alessandro Fumagalli, and Manfred Doermer. Multi-functional interface for flexibility and reconfigurability of future european space robotics. *Advances in Astronautics Science and Technology*, 1(1):119–133, June 2018.

[169] Tingwu Wang, Yuhao Zhou, Sanja Fidler, and Jimmy Ba. Neural graph evolution: Automatic robot design. In *International Conference on Learning Representations*, 2019.

[170] Tingwu Wang, Yuhao Zhou, Sanja Fidler, and Jimmy Ba. Neural graph evolution: Towards efficient automatic robot design. *CoRR*, abs/1906.05370, 2019.

[171] Ronald J. Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

[172] James D. Wray, Harlan J. Smith, Karl G. Henize, and George R. Carruthers. Space Schmidt Telescope. In Lawrence D. Barr and Geoffrey Burbidge, editors, *Advanced Technology Optical Telescopes I*, volume 0332, pages 141 – 150. International Society for Optics and Photonics, SPIE, 1982.

[173] Yun-Hua Wu, Zhi-Cheng Yu, Chao-Yong Li, Meng-Jie He, Bing Hua, and Zhi-Ming Chen. Reinforcement learning in dual-arm trajectory planning for a free-floating space robot. *Aerospace Science and Technology*, 98:105657, 2020.

[174] Lei Xiujuan and Shi Zhongke. Overview of multi-objective optimization methods. *Journal of Systems Engineering and Electronics*, 15(2):142–146, 2004.

[175] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2048–2057, Lille, France, 07–09 Jul 2015. PMLR.

[176] Wenfu Xu, Jianquing Peng, Bin Liang, and Zonggao Mu. Hybrid modeling and analysis method for dynamic coupling of space robots. *IEEE Transactions on Aerospace and Electronic Systems*, 52(1):85–98, February 2016.

[177] Tsuneo Yoshikawa. Manipulability and redundancy control of robotic mechanisms. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 1004–1009, 1985.

[178] Tsuneo Yoshikawa. Manipulability of robotic mechanisms. *The International Journal of Robotics Research*, 4(2):3–9, 1985.

[179] Wenhao Yu, C Karen Liu, and Greg Turk. Policy transfer with strategy optimization. *CoRR*, abs/1810.05751, 2018.

[180] L.A. Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control*, 8:59–60, 1963.

[181] Dan Zhang and Bin Wei. Modelling and optimisation of a 4 DoF hybrid robotic manipulator. *International Journal of Computer Integrated Manufacturing*, 30(11):1179–1189, 2017.