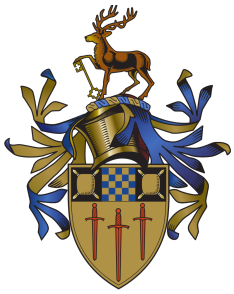


University of Surrey

Surrey Space Centre

Faculty of Engineering and Physical Sciences

Guildford, Surrey GU2 7XH, U.K.

Trajectory Design using Lyapunov Control Laws and Reinforcement Learning

Harry J. Holt

Submitted for the Degree of

Doctor of Philosophy

Supervisors

Dr Nicola Baresi

Prof Roberto Armellin

Co-supervisor

Dr Christopher Bridges

External supervisors

Dr Andrea Turconi

Dr Yoshi Hashida

March 2022

Declaration of Originality

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 Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in the text, bibliography or in footnotes.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.

Signed: Harry Holt

Date: 05/12/2022

"Earth is the cradle of humanity, but one cannot remain in the cradle forever"

Konstantin E. Tsiolkovsky, 1911 (adapted)

Abstract

Spacecraft trajectory design is critical to successful space missions, particularly with the increasing number of spacecraft and mission complexity. Satellite constellation deployment, reconfiguration, orbit raising and deep-space missions all depend on effective trajectory design. In addition, spacecraft autonomy is a major barrier to increasing the scope, ambition and affordability of both Earth-based and deep-space missions. The current state-of-the-art in spacecraft operations is still to guide space missions from the ground with extensive human intervention. Whilst automated guidance, planning and trajectory design tools do exist, they often lack the vital skill of human operators, who can act under environmental and mission uncertainty.

The goal of this thesis was to develop and investigate a lightweight and closed-loop control law that can be used for both initial trajectory design and subsequent on-board guidance. The motivation behind this research is to combine the stable yet sub-optimal nature of Lyapunov control laws with the exploration and state-dependence offered by reinforcement learning techniques. This has resulted in the development of a novel Reinforced Lyapunov Controller. The Lyapunov stability implications are examined and an analytical expression for the state-weight Jacobian is presented. Performance in Keplerian dynamics is investigated to assess the optimality and stability of the approach. New training procedures in the presence of unmodelled dynamics including perturbations, eclipse events and stochastic errors are presented. A cone-clock angle approach is devised to explore the additional degrees of freedom and further ensure Lyapunov stability. Results show the Reinforced Lyapunov Controller is optimal and stable in modelled dynamics, and robust to uncertain and stochastic environments. Including such uncertainty in the training procedure can further improve the performance. The versatility is considered by approximating finite-burn trajectories and incorporating operational constraints. Finally, the potential of the Reinforced Lyapunov Controller for Earth-Moon spiral transfers is investigated, exploring the use of a two-body control law in a three-body environment. In both cases, the Reinforced Lyapunov Controller is able to compete with conventional evolutionary algorithm methods.

Acknowledgements

This thesis concludes my PhD research over the past three and a half years. I want to take the opportunity to thank all the people who made it possible and supported me throughout.

I'd like to start by thanking my supervisors *Prof Roberto Armellin* and *Dr Nicola Baresi*. *Roberto*, thank you for taking me on, especially at short notice and giving me this opportunity to delve into the field of Astrodynamics. You have provided me with great advice throughout, not only with your vast technical knowledge but also in terms of writing, presenting and helping me to become a better researcher. Since leaving for Toulouse and then Auckland, I have been astounded by your commitment aiding in the supervision of my project. Now I have the pleasure of joining you out there and here's to many more GTOCs together.

To *Nicola*, who stepped in to supervise my project - thank you! If you had not been as enthusiastic and willing to jump in at the deep end, then I doubt I would have continued the PhD. You brought such life back to the Astrodynamics group here at Surrey and provided me with some terrific opportunities and projects to get involved in to broaden my experience, including VMMO and MMX. I learnt so much from you; thank you for being so open and willing to give advice and feedback, both technical and non-technical. One day you will have to share your time-management secrets.

On to my co-supervisor, *Dr Chris Bridges*. I really appreciate all your support, particularly during the change of primary supervision, and for your very useful advice on research and paper writing. Your different viewpoint, particularly on the machine learning side, helped me greatly. I am very grateful to both you and *Nicola* for the opportunity to go to a conference in Florida before Covid hit. I did not think that would be the only one, but it was the highlight of my PhD studies.

Now on to my funders SSTL (Surrey Satellite Technology Limited). Thank you for giving me the opportunity to pursue this research. To *Dr Andrea Turconi* and *Dr Yoshi Hashida*, I'm very grateful for the time and valuable insights you gave me during our monthly supervision meetings over the last three and a half years. I got a unique insight into the industrial perspective and really appreciate the thought and interaction you gave me. Thanks also to *Steve Eckersley* for your insights on mission analysis and facilitating these meeting.

I'd also like to thank the staff of the Surrey Space Centre for being so welcoming and to *Karen* and *Louise* for all the help along the way. A lot of my initial understanding of reinforcement learning I owe to *Andrea Scorsoglio* and *Prof Roberto Furfaro*; I really enjoyed the collaboration. I must also thank the University of Surrey's HPC Eureka facilities,

without which I might not have been able to explore as many different scenarios and test cases.

Next I'd like to thank all my friends from the Surrey Space Centre. Firstly, my fellow astrodynamicists *Laura*, *Nicolò* and *Fu*, with whom I shared so much of my PhD experience. *Laura*, thank you for welcoming me into the group and showing me the ropes in my first year, providing endless help debugging and setting the bar for organisation and presentation skills far too high. *Fu* and *Nicolò* - guys it was such a pleasure to share the office and bulk of my studies with both of you. I appreciate all the support and laughs we had - although in the end I wasn't quite able to convert you into cricket fans. After so many late nights toiling away on the HPC - *Fu* it's all yours now. Thank you to *Asma* and *Rachel*, my officemates pre-Covid, and more recently *Edo* and *Danny*, I'm sorry you both had to put up with my stress in the final months. It's good to know the group will continue to thrive.

Thanks to *Silvia*, *Gianluca*, *Thomas* and *Corinna* for really welcoming me in Guildford and inviting me along to many pizza nights I won't forget. That all continued in Linden Road with *Silvia*, *Gianluca*, *Thomas* and *Nicolò* - definitely the highlight of my time in Guildford. So many great evenings with you all - thanks for keeping me well fed and involving me in all your planned events. Then of course *Mansur*, a great friend and sounding board for many complaints and rants. You put up with so much chat and definitely helped keep me sane. Along with *Moe*, you shared my relentless enthusiasm for all things football. It's a shame our telepathic combinations on the pitch must come to an end. And of course to *Olek* and *Emilia*, for all the board games and fun evenings - I wish we had done more of those.

Then there's all my friends elsewhere who supported me throughout. Particular thanks to the *ChuSquad* and to *Dan*, *Oscar*, *Gabs* and *Illia* for providing a terrific escape and change of environment. I can't forget *Jim*, for your advice and your encouragement for me to pursue what I enjoy doing, always giving me a unique perspective on life.

Finally, to my family. *Fred* and *Ruby* - thank you for putting up with me and for pretending to show an interest when I get carried away explaining things. I love you both so much, and cherish the bond we share, it really means the world to me. To *Granny*, thank you for all your love and I wish I could share more with you. To *Grandpa*, your curiosity and enthusiasm is truly inspiring and there are few things better than a conversation with you that starts on space and the universe and ends with an in-depth analysis of the latest goalkeeping blunders.

Mum and *Dad* - you are always there for me to give me love, support and advice when I need it most. I am so grateful for all the opportunities you have given me in life, and your endless support and encouragement for me to pursue my dreams. Seeing you both, and *The Cokes*, really feels like home. Thank you.

Harry

Contents

List of Figures

List of Tables

List of Acronyms

AC	actor-critic
ACO	ant colony optimisation
ADCS	Attitude Determination and Control System
CC	cone-clock
CLFD	closed-loop feedback-driven
COCP	continuous optimal control problem
COE	classical orbital element
COV	calculus of variations
CR3BP	circular restricted 3-body problem
DAG	directional adaptive guidance
DDPG	deep deterministic policy gradient
DT	deterministic training
ECI	Earth-centred inertial
ELM	extreme learning machine
EP	electric propulsion
EX	execution
GA	genetic algorithm
GEO	geostationary orbit
GNC	guidance, navigation and control
GTO	geostationary transfer orbit
GVEs	Gauss variational equations
ICEA	improved cooperative evolutionary algorithm
KKT	Karush-Kuhn-Tucker
LEO	low-Earth orbit
LPO	Lunar polar orbit
LVLH	local-vertical-local-horizontal
MC	Monte Carlo
MCI	Moon-centred inertial

MDP	Markov decision process
MEE	modified equinoctial element
ML	machine learning
MPBVP	multi-point boundary value problem
NC	neurocontroller
NLP	nonlinear programming
NN	neural network
NRHO	near-rectilinear halo orbit
OD	orbit determination
OI	orbit insertion
PMP	Pontryagin's minimum principle
PPO	proximal policy optimisation
PSO	particle swarm optimisation
RAAN	right ascension of the ascending node
RBF	radial basis function
ReLU	rectified linear unit
RL	reinforcement learning
RTN	radial-transverse-normal
SB	shape-based
SBT	stochastic batch training
SLFN	single-layer feed-forward network
SOI	sphere of influence
SRP	solar radiation pressure
SSO	Sun-synchronous orbit
SSTL	Surrey Satellite Technology Ltd
TRPO	trust-region policy optimisation
VMMO	Volatile Mineralogy Mapping Orbiter
ZEM/ZEV	zero-effort-miss/zero-effort-velocity

List of Symbols

Coordinate Systems

$(a, e, i, \Omega, \omega, \nu)$	classical orbital elements (COEs)
(p, f, g, h, k, L)	modified equinoctial elements (MEEs)
(x, y, z, v_x, v_y, v_z)	Cartesian coordinates

Greek Symbols

α	clock angle
α_k	learning rate
β	cone angle
β_f	actor network scaling factor
β_{ELM}	ELM parameters
$\boldsymbol{\psi}$	actor network basis function
δ_e	throttle factor
ϵ	PPO clipping parameter
η	effectivity parameter
η^t	effectivity threshold
η_a	absolute effectivity
η_a^t	absolute effectivity threshold
η_r	relative effectivity
η_r^t	relative effectivity threshold
$\eta_{eclipse}$	eclipse parameter
γ	discount factor
λ	geocentric longitude
λ_X	weighting parameters for ΔV-law
μ	mass parameter
ϕ_α	in-plane control angle
ϕ_β	out-of-plane control angle
ϕ_a	azimuth
ϕ_e	elevation

ϕ_{gc}	geocentric latitude
$\pi_\theta^\star$	optimal RL policy with parameters θ
π_θ	RL policy with parameters θ
σ	actor network exploration
σ_{net}	actor network neuron spacing
$\tau(\pi(\theta))$	Trajectories computed using a stochastic policy $\pi(\theta)$
$\tau(\theta)$	Trajectories computed using a deterministic policy $\pi(\theta)$
θ	actor network parameters
ξ	Brouwer-Lyddane mean to osculating transformation

Latin symbols

$\bar{U}$	effective potential
$\bar{\boldsymbol{X}}$	spacecraft mean orbit element state
$\boldsymbol{\mathcal{B}}$	matrix representation of GVEs for slow and fast variables
$\boldsymbol{a}_d$	disturbing acceleration
$\boldsymbol{a}_p$	perturbing acceleration
$\boldsymbol{a}_T$	perturbing acceleration during training
$\boldsymbol{B}$	matrix representation of GVEs for slow variables
$\boldsymbol{b}$	ELM biases
$\boldsymbol{c}_i$	single neuron centre
$\boldsymbol{C}_X$	vector of neuron centre
$\boldsymbol{F}$	force
$\boldsymbol{H}$	vector of ELM activation functions h
$\boldsymbol{M}$	matrix representing derivative of Lyapunov function
$\boldsymbol{p}$	vector that maximises rate-of-change of Lyapunov function
$\boldsymbol{p}_L$	user-defined Lyapunov control law parameters
$\boldsymbol{r}$	spacecraft position
$\boldsymbol{u}$	spacecraft control vector
$\boldsymbol{u}_{\text{jacobian}}$	spacecraft control vector computed with state-weight Jacobian
$\boldsymbol{u}_{\text{original}}$	spacecraft control vector computed in conventional manner
$\boldsymbol{v}$	spacecraft velocity
$\boldsymbol{W}$	Lyapunov weights/gains
$\boldsymbol{w}$	ELM weights

$\boldsymbol{X}$	spacecraft orbit element state
$\boldsymbol{x}$	spacecraft Cartesian state
$\boldsymbol{X}_{\text{conv}}$	state convergence criteria
$\boldsymbol{X}_{\text{est}}$	estimated state
$\boldsymbol{Y}$	ELM training samples
Δr_{SOI}	distance between spacecraft and SOI boundary
$\Delta t_{\Delta V}$	interval between burns
ΔV	impulse per unit mass to perform a manoeuvre
ΔV_{burn}	individual burn ΔV budget
$\hat{\boldsymbol{p}}$	direction of maximum rate-of-change of Lyapunov function
$\hat{A}$	approximate advantage function
$\hat{Q}$	approximate state-action value function
$\hat{V}$	approximate state value function
A	advantage function
a	action
a	apparent radius of occulting body
b	apparent radius of occulted body
B_δ	neighbourhood of a given state
b_{petro}	Petropoulos Q-law parameter
C	Jacobi constant
C_i	discounted cost-to-go for state i
c_i	cost for state i
c_{mass}	mass-optimal cost function
c_{time}	time-optimal cost function
C_{nm}	geopotential constant coefficient of degree N and order m
D	number of actor network features
E	total energy
E_{2B}	two-body energy
E_{L1}	total energy at the $L1$ point
f	acceleration magnitude provided by engine thrust
f_{scale}	actor network scaling function
f_T	thrust-blended acceleration direction

f_X	acceleration direction for individual element X
h	specific angular momentum
h_i	ELM activation function i
I_{sp}	spacecraft engine specific impulse
J_{π_θ}	RL objective for policy π_θ
J_i	ith degree zonal gravitational harmonic
k_{petro}	Petropoulos Q-law parameter
L	number of ELM nodes
M	mean anomaly
m	spacecraft mass
m^p	spacecraft propellant mass
N	number of RL episodes
n	mean motion
n_{petro}	Petropoulos Q-law parameter
N_X	number of basis functions per orbital element
P	Lyapunov penalty function
p	semi-lactus rectum
P_{nm}	Legendre function of order m and degree N
Q	Q-law Lyapunov function
Q	state-action value function
$R(\theta)$	PPO probability ratio
$r_{\text{petro}}^{\text{p-min}}$	Petropoulos Q-law parameter: minimum periapsis altitude
r_{petro}	Petropoulos Q-law parameter
r_{SOI}	radius of sphere of influence
S_X	Lyapunov scaling function for orbit element X
S_{nm}	geopotential constant coefficient of degree N and order m
T	number of RL time-steps in an episode
T	orbital period
T	spacecraft engine thrust
t	time
t_{aim}	maximum time-of-flight for mass-optimal simulations
t_{conv}	time-to-go convergence criteria

t_{res}	estimated residual time-to-go
U	gravitational potential
V	Lyapunov function
V	state value function
v_∞	v-infinity
V_c	current circular orbit velocity
v_e	exhaust velocity
V_{cf}	final circular orbit velocity
W_P	Lyapunov penalty function weight/gain
W_X	Lyapunov weight/gain for orbit element X
x	state

Other Symbols

$\mathbb{E}[\square]$	expectation when applying the policy π_θ
$\mathcal{N}(\mu, \sigma)$	normal distribution centred on μ with variance σ^2

Physical constants

μ_E	Earth's gravitational parameter	$3.9860049 \times 10^{14}\,\text{m}^3\,\text{s}^{-2}$
μ_M	Moon's gravitational parameter	$4.9048695 \times 10^{12}\,\text{m}^3\,\text{s}^{-2}$
G	gravitational constant	$6.67430 \times 10^{-11}\,\text{m}^3\,\text{kg}^{-1}\,\text{s}^{-2}$
g_0	standard gravitational field strength	$9.80665\,\text{m}\,\text{s}^{-2}$
J_2	2^{nd} degree zonal gravitational harmonic	$1.082635854 \times 10^{-3}$ –
$R_\odot$	radius of the Sun	$6.96000 \times 10^8\,\text{m}$
R_E	Earth's mean radius	$6.3781363 \times 10^6\,\text{m}$
R_M	Moon's mean radius	$1.73710 \times 10^6\,\text{m}$

Operators, Subscripts and Superscripts

$\boldsymbol{f}(\square)$	function representing a dynamical system
$\ddot{\square}$	second derivative
$\Delta\square$	large variation
$\delta\square$	small variation
$\dot{\square}$	first derivative
$\frac{\partial\square}{\partial \boldsymbol{W}}$	partial derivative w.r.t. weights $\boldsymbol{W}$
$\frac{\partial\square}{\partial \boldsymbol{X}}$	partial derivative w.r.t. state $\boldsymbol{X}$
$\frac{d\square}{d\boldsymbol{X}}$	derivative w.r.t. state $\boldsymbol{X}$

$\frac{d\square}{d\boldsymbol{x}}$	derivative w.r.t. state $\boldsymbol{x}$
$\frac{d\square}{dt}$	derivative w.r.t. time t
$\hat{\boldsymbol{e}}_{\square}$	basis unit vector
$\hat{\square}$	unit vector
$\|\square\|$	vector norm
$\nabla_{\theta}\square$	gradient w.r.t. θ
$\Pi\square$	product
$\square^{\pi}$	in relation to policy π
$\square^{ECI}$	in the ECI frame
$\square^{MCI}$	in the MCI frame
$\square_{\text{class}}$	values visited by the classical Lyapunov controller
$\square_{\theta}$	transverse component in $R\theta h$ frame
$\square_{d}$	in relation to actor network node d
$\square_{h}$	normal component in $R\theta h$ frame
$\square_{k}$	k^{th} iteration
$\square_{p}$	primary body
$\square_{r}$	nominal reference motion
$\square_{r}$	radial component in $R\theta h$ frame
$\square_{s}$	secondary body
$\square_{0}$	initial value
$\square_{arr}$	arrival value at LPO
$\square_{av}$	average secular variation
$\square_{dep}$	departure value at GTO
$\square_{f}$	final value
$\square_{i}$	in-plane component
$\square_{o}$	out-of-plane component
$\square_{T}$	target value
$\sum\square$	summation

Chapter 1
Introduction

1.1 History

Since the launch of Sputnik 1 in 1957, the number of spacecraft and complexity of these missions has increased significantly. As of July 2019, there are about 2400 active spacecraft, providing a wealth of services [1]. Communications, weather forecasting, remote sensing and navigation have all become dependent on Earth-orbiting satellites. Further scientific missions have extended our understanding of the solar system and our universe immeasurably. However, with these increased services comes increased complexity. Constellations of satellites are now used to provide constant coverage of services across the globe and mega-constellations of thousands of satellites such as OneWeb and Starlink [2] are being developed, aiming to provide internet access everywhere on Earth.

Spacecraft trajectory design is a critical component of successful space missions. It not only allows the spacecraft to complete its intended mission objectives, but also increases the feasible complexity of these missions. The problem of designing optimal trajectories can be stated as determining a trajectory for a spacecraft given a set of initial and final states and an objective that is to be minimised. This is a challenging task as the problem is a continuous optimal control problem (COCP). The dynamics are non-linear and often chaotic, and may contain discontinuities. Terminal conditions and interacting forces are often time-dependent and hence are not known explicitly. The basic structure of the solution might not be known *a priori*, making attempts to guess the solution difficult. Hence, it is computationally intensive, making mission analysis challenging and trajectory design currently unsuitable for on-board implementation.

Manoeuvres can either be impulsive or continuous, depending on the duration of the engine burn. When this is short enough in comparison with the total flight time, the trajectory is modelled as impulsive. Between impulses, the spacecraft motion is deterministic and evolves according to the natural equations of motion (in Earth-orbiting satellites this is Keplerian motion to first approximation). Up until 1993, with the launch of the first electric propulsion (EP) system [3], this was sufficient to determine the trajectory as all spacecraft had chemical engines.

However, the development of high specific impulse, low-thrust engines such as EP systems has resulted in spacecraft with near continuous thrust. As of 2017, 243 satellites had launched with EP systems [3]. Initially used for station-keeping tasks, EP is now increasingly used for orbit raising applications, demonstrated in full by the SMART-1

satellite in 2003 [4], when it flew to the Moon, and confirmed by the successful flight of Eutelsat 172B in June 2017 [5]. This was the first European high power satellite to use EP for all manoeuvres, taking approximately four months to rise from geostationary transfer orbit (GTO) to geostationary orbit (GEO). Using its three low-thrust ion engines, NASA's Dawn spacecraft became the first spacecraft to orbit two extraterrestrial bodies in Ceres and Vesta in 2007 [6]. JAXA's Hayabusa 1 & 2 both used ion thrusters to offer high degrees of spacecraft manoeuvrability [3], and recently ESA's BepiColombo is using low-thrust propulsion [7]. Solar power sails (a combination of solar sails and ion engines) are also a developing technology that are near continuous low-thrust. Whilst solar sails pose other challenges and constraints to EP systems, they also share similarities in the complex trajectory design requirements. IKAROS was the first mission to deploy a deep space solar sail and use it as a propulsion mechanism. JAXA are currently developing the OKEANOS mission to visit the Jovian Trojan asteroids using solar power sails [8].

This low-thrust trajectory design is a distinctly different problem to impulsive trajectories. The manoeuvres can no-longer be approximated as impulsive and, in many cases, an optimal transfer becomes a multi-revolution problem. This is advantageous as a large ΔV can be accumulated with comparatively little propellant, allowing for an increased fraction of the spacecraft's mass to be devoted to other systems. However, this further increases the trajectory design complexity [9] as the direction, magnitude and time-history now needs to be modelled, and the effects of perturbing forces are more noticeable. This increase in computational difficultly impacts both mission design phases and, perhaps more importantly, future on-board guidance.

With an increase in the number of active satellites, along with the increasingly crowded Earth-orbit environment [10], comes a drive towards autonomy. Spacecraft autonomy is a major barrier to increasing the scope, ambition, and affordability of both Earth-based and deep-space missions. Currently, most spacecraft require in-loop ground control to follow planned trajectories, to carry out orbital manoeuvres and to perform engine burns. This requires extensive human intervention [11]. Automated guidance, planning and trajectory design tools do exist. However, they often lack the vital skill of human operators, who can account for changing mission objectives, circumnavigate temporary obstacles and act under environmental and mission uncertainty. In the case of low-thrust transfers, the time between launch and operational orbit is increased compared to those with chemical propulsion systems. These can lead to additional operational costs from ground station usage and personnel. In addition, the development of mega-constellations and the increasing tendency to launch multiple spacecraft from single launch vehicles, has led to a need for more autonomy. Spacecraft that can execute transfers without necessary ground coverage allow for faster constellation deployment as more spacecraft can be operated (or monitored) simultaneously. Increasing autonomy is therefore an important aspect of low-thrust trajectory design in particular.

1.2 Trajectory Design

The spacecraft trajectory design problem is the problem of determining the trajectory given a set of initial and final states and an objective to be minimised. The process can be divided into four main steps [12]. The first is the choice of dynamical model and states to represent the system. Next, the objectives and constraints need to be determined. The third step involves selecting an appropriate method for solving the trajectory design problem, and determining how it is best formulated. Finally, this has to be solved for the control history. The remainder of this section will consider different approaches to solving it. Shirazi *et al.* [12] and Morante *et al.* [13] provide nice surveys of the available approaches, which are summarised here. Tools developed for low-thrust trajectory optimisation can be judged by the following criteria:

1. Flexibility (to different scenarios and mission planning)
2. Robustness (sensitivity to input parameters)
3. Speed (for easy mission analysis and feasibility studies)
4. Accuracy (provides meaningful results)
5. Automation (minimise the user-interaction)
6. Optimality (provides optimal results)

Once a problem is formulated, including the equations of motion, the cost function and the constraints, calculus of variations (COV) can be used to derive the necessary conditions for optimality for these problems. However, solving the resulting systems of equations and boundary conditions is very challenging [14]. Overall, the available solution approaches can be divided into two types: analytical and numerical [13]. Analytical methods produce closed-form solutions for the optimal trajectory. Lawden's primer vector approach [15] can be used in certain simplified impulsive-thrust and continuous-thrust cases to provide the analytical necessary conditions. However, obtaining the necessary analytical solutions is very challenging and only possible in a minor number of cases: such as very-low-thrust orbit raising transfers thanks to work done by Kechichian [16] to extend that of Edelbaum [14, 17]. Thus, the vast majority of researchers and analysts today focus on numerical approaches to solving low-thrust trajectory optimisation problems. These can be divided in three types: indirect, direct and dynamic programming. These differ based on their underpinning approach and theory, but each results in a mathematical problem which can subsequently be solved using a variety of overlapping techniques.

Indirect methods rely on the Pontryagin's minimum principle (PMP), which categorises the first-order necessary conditions for an optimal solution. The goal is to solve the resulting multi-point boundary value problem (MPBVP), and is done using states and

costates which obey the Euler-Lagrange equations. However, this increases the difficulty of finding candidate solutions, particularly as the necessary number of costate variables immediately doubles the size of the dynamical system. The system is also often sensitive to the initial costate variable values and as they lack any physical significance, this can limit the ability to find the optimal solution.

Direct approaches transcribe the COCP into a nonlinear programming (NLP) problem. This requires the discretisation of the control variables into a time-grid. By parametrising the input and control vectors the problem becomes a parameter optimisation problem, where the vector of unknown decision variables ensures a set of nonlinear constraints are met. The Karush-Kuhn-Tucker (KKT) conditions then describe the first-order necessary optimal conditions. Unlike in the indirect approach, these first-order necessary conditions do not need to be derived. The downside of this approach is the optimality of a candidate solution cannot be guaranteed, as the approach may only converge to a sub-optimal local minimum [14].

Dynamic programming relies on the Hamilton-Jacobi-Bellman theory and Bellman's principle of optimality [18]: "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions". The resulting challenge is to solve the set of partial differential equations. Dynamic programming techniques can provide the global optimal solution as the whole search space is considered. In addition, once a solution is found, all possible controls have been computed. As such, a closed-loop control policy can be obtained, which is highly beneficial for uncertain and stochastic environments. The disadvantage is the well known curse of dimensionality [13].

All three of these approaches use a variety of techniques to impose the dynamical equations on the solution. These include single-/multiple-shooting techniques and collocation methods. Indirect methods and dynamic programming can also be solved by gradient methods and direct methods by differential inclusion. All three result in a vector of unknown decision variables of some form, which can then be solved using iterative methods such as gradient-based, heuristic and hybrid approaches. These differ based on their rules for evolving from one iteration to the next. Betts [19] provides a very comprehensive overview of these techniques. A quantitative comparison of the approaches for several test cases can be found in Pritchett's work [20].

The approaches discussed thus far are computationally intensive and despite the increase in computing ability in recent years, finding a solution is a non-trivial and challenging task. This also renders these techniques inappropriate for on-board autonomous use and for mission feasibility studies, as faster solutions are required due to computing power and time limitations respectively. On-board guidance involves mapping states estimates to thrust commands to steer a spacecraft to fly a trajectory that satisfies the specific objectives or targeting conditions [21, 22]. There are a few main requirements in autonomous guidance and trajectory design: low-computational cost, convergence of the

algorithm on-board and robustness of the obtained control policy. Such algorithms are currently one of two types: reference-based guidance (i.e. tracking a reference trajectory) or targeting condition guidance (i.e. drive towards a target) [23, 24]. In reference-based guidance (which includes neighbouring optimal methods), either a linear quadratic regulation or similar tracking problem is solved. Alternatively, targeting condition guidance algorithms use parametric methods that use a predictor-corrector to solve an initial value problem and reach a terminal state. Generally, the main optimisation of the reference trajectory has been carried out offline, and the onboard guidance method is tasked with either steering back to the nominal trajectory, or solving an initial value problem from the current state to achieve the targeted state.

1.2.1 Heuristic Control Laws

Heuristic control methods offer an alternative approach to solving the trajectory design problem. Due to the complexity of finding a solution for the trajectory design approaches described in Section 1.2, they are often used by mission designers for preliminary design and as initial guesses for indirect or direct methods.

Heuristic control law techniques likely developed from intuition during mission design phases, and can combine analytical expressions to help parameterise the control. Morante *et al.* [13] divide them into five types: COV, shape-based (SB) methods, Thrust-blending control, Lyapunov control laws and neurocontrollers (NCs).

For the COV-based approaches, PMP is used to obtain the optimal control history, but requires solving an analytical state-dependent matrix in the same way as indirect methods. This requires recomputing each time the objective or constraints are changed, and solving the resulting equations and boundary conditions is very difficult. As mentioned previously, Lawden's primer vector method [15] can be used to derive analytical expressions for the necessary conditions in certain simplified cases. Given the non-linearity of the problem, in many other scenarios simplifying assumptions and specific conditions are often required to achieve analytical solutions. However, sometimes these solutions correspond very well to the exact solutions. For instance, Edelbaum [17] helped derive formulae for transfers between circular orbits, even in non-planar cases. Several authors have since extended the feasibility of these solutions (see Conway *et al.* [14]), including incorporating eclipse effects and Earth's oblateness for low-thrust transfers [16]. Overall these analytical approaches are optimal, but only within the approximations they make. As such, they can often provide sub-optimal solutions and used as heuristic control laws by mission analysts.

Another analytical approach, SB methods [25–27], involve assuming the trajectory takes a predefined form and computing a control such that the spacecraft is forced to follow this form. The intention is to satisfy both the equations of motion and boundary conditions simultaneously by treating the trajectory as a whole rather than divid-

ing it into several segments. They provide a geometrical description of the trajectory, which suffices to calculate the time history of the control vector. These are very well suited to rapid trajectory design and broad searches. The shapes are usually selected with the fewest possible design parameters. Examples include exponential sinusoids, inverse polynomials, spherical and finite-fourier series [27]. The later is perhaps the most successful, capable of producing 3D trajectories, where the control history is assumed to be represented by a finite-fourier series expansion. In their simplest form, SB methods are neither time- nor fuel-optimal and the resulting trajectory can be characterised as a feasible solution to the problem. Whilst careful tailoring is able to minimise, for example, the ΔV cost, in general their sub-optimality derives from their predetermined fixed nature.

The three remaining types can be classed as closed-loop feedback-driven (CLFD) control laws. These CLFD control laws are a subset of heuristic control laws that only require knowledge of the current spacecraft state and the target state in order to determine the control to be applied. Whilst it is possible to imagine a state-dependent mapping for many different types of techniques to make them closed-loop, the remainder of this section discusses the three main classes of CLFD control laws.

NCs transform the problem of finding an optimal trajectory into determining an optimal neural network (NN) mapping from the current and target spacecraft state to the control history. Dachwald [9] introduced these NCs with evolutionary algorithms to allow low-thrust trajectories to be computed without initial guesses and without expertise in optimal control theory. The focus was on near-Earth asteroids and other interplanetary mission analysis. Ohndorf [28] used a genetic algorithm (GA) as part of an evolutionary NC to design low-thrust interplanetary transfers with gravity assists. There are many applications of NCs in astrodynamics, and more specifically when it comes to machine learning (ML) [29]. This is split between *supervised learning*, where NCs are designed to replicate optimal control behaviour, and more recently in reinforcement learning (RL). There is growing interest in RL in astrodynamics, from mission design, operations, guidance and control, to navigation and even the prediction of the dynamics [29, 30]. These will be discussed at length in Section 2.4. However, the extensive survey by Shirobokov *et al.* [29] highlights two key areas for future development with RL in astrodynamics. Firstly, they discuss the importance of stability in astrodynamics and the lack of such guarantees within many RL approaches thus far. Secondly, it is key to investigate NN behaviour in scenarios beyond model-based problems, attempting to explore real flight modes and response to unmodelled external disturbances and uncertainties. This extrapolation beyond the training environment is of considerable concern within space-based applications. In addition, whilst NN controllers can be effective for approximating nonlinear control functions, one notable limitation in directly employing a NN as a controller is that the corresponding solution may violate constraints the network is unaware of [21].

The two remaining types of heuristic control approaches are thrust-blending and

Lyapunov-based control. These two approaches consider the trajectory design problem from a targeting perspective rather than an optimisation one, meaning they are often sub-optimal. The trouble with thrust-blending approaches is the lack of guaranteed stability. Stability plays an important role in spacecraft trajectory design and as such many CLFD control laws are based on Lyapunov control theory [31, 32]. This provides them with an inbuilt stability, even if they are modified to an extent where they no longer meet the strict stability requirements. Joseph [33], Naasz [34] and Petropoulos's Q-law [35, 36] are all such examples. A critical evaluations of these can be found in Hatten's comparison of CLFD control laws [31]. This concluded that Petropoulos Q-law and Ruggiero's directional adaptive guidance (DAG) were the most versatile and optimal in terms of time-of-flight for low-thrust transfers.

The hallmark advantages CLFD control laws are:

- Computational ease, as no system of nonlinear equations needs to be solved. This make them suitable as initial guesses for indirect and direct methods [37, 38].
- Potential for on-board use as knowledge of current state is sufficient to determine the control. This can provide robustness to uncertainty in state and thruster misalignment.
- They can leverage properties of dynamics through analytical expressions, such as the maximum rate of change for specific orbital elements [36, 39].

However, the disadvantages are

- They are generally derived in low-fidelity dynamics.
- They are sub-optimal and have user-defined parameters which significantly effect performance.
- They might have stability and control chatter issues as they are not formulated as mathematically rigorously as indirect and direct methods.

1.3 Research Aims and Objectives

This PhD aims to develop a lightweight and closed-loop control law that can be used for both trajectory design and subsequent on-board guidance. In doing so, the goal is to increase the optimality and retain the stability of CLFD control laws using RL methods. The motivation behind this research is to combine the stable yet sub-optimal nature of Lyapunov control laws with the exploration and state-dependence offered by RL techniques and has resulted in the development of a novel Reinforced Lyapunov Controller. To the author's knowledge, little work has combined RL with these Lyapunov control laws for trajectory design specifically, which can result in a powerful state-dependent

and more optimal controller with guaranteed Lyapunov stability. This is tested in a variety of deterministic and stochastic environments. Novel control and learning architectures are presented to increase performance in these environments.

These aspirations resulted in the following objectives for this research project:

- To develop optimal state-dependent Lyapunov control laws for trajectory design and on-board guidance.
- To investigate the stability implications of combining RL with Lyapunov control laws.
- To improve the performance of Lyapunov control laws in the presence of perturbing accelerations and high-fidelity dynamics.
- To assess the robustness of these methods under uncertain and stochastic environments, and pave the way for future on-board use.
- To explore the limits of these techniques, both in terms of thrust magnitude and dynamical environments.

The resulting techniques should be closed-loop and lightweight, but provide near optimal yet stable solutions. In addition, the resulting techniques should be applicable to many different scenarios and suitable for both initial trajectory design and potential on-board use. In addition to exploring methods for improving Lyapunov control laws, this thesis aims to investigate two current drawbacks of RL techniques: the lack of guaranteed stability and their behaviour in the presence of unmodelled external disturbances and uncertainties. Finally, whilst the computational cost will not be directly investigated, the separation between training and closed-loop evaluation should produce a computationally efficient controller with attractive on-board characteristics, requiring neither an initial guess generation, no reference trajectory, onboard iteration nor numerical integration to operate.

1.4 Publications, Conferences and Workshops

1.4.1 Publications

As first author:

- **H. Holt**, R. Armellin, N. Baresi, A. Turconi, Y. Hashida, A. Scorsoglio and R. Furfaro (2021). *Optimal Q-laws via Reinforcement Learning with Guaranteed Stability*. Acta Astronautica vol. 187, pp. 511–528.

As contributory author:

- Shirazi, A., **H. Holt**, R. Armellin and N. Baresi (2022, foreseen). *Time-Varying Lyapunov Control Laws with Enhanced Estimation of Distribution Algorithm for Low-Thrust Trajectory Design*. Modeling and Optimization in Space Engineering – New Concepts and Approaches (resubmitted after 1st review - awaiting response).

Manuscript in preparation:

- **H. Holt**, N. Bernardini, N. Baresi and R. Armellin, (2023, foreseen) *Towards Optimal Lyapunov Controllers for low-thrust Lunar Transfers via Reinforcement Learning and Convex Optimisation*. Manuscript in preparation.

1.4.2 Conferences

As presenter and first author:

- **H. Holt**, R. Armellin, A. Scorsoglio, R. Furfaro, *Low-thrust trajectory design using closed-loop feedback-driven control laws and state-dependent parameters*. In AIAA Scitech 2020 Forum / AAS Spaceflight Mechanics Meeting Winter 2020, Orlando FL. **Presenter**.

- **H. Holt**, R. Armellin, N. Baresi, A. Scorsoglio, R. Furfaro, *Low-thrust trajectory design using closed-loop feedback-driven control laws and Reinforcement Learning*. In AAS Astrodynamics Specialist Conference Summer 2020, Lake Tahoe CA (Virtual). **Presenter**.

- **H. Holt**, N. Baresi, R. Armellin, *Towards Optimal Lyapunov Controllers for low-thrust Lunar Transfers via Reinforcement Learning*. In AAS Astrodynamics Specialist Conference Summer 2021, Big Sky MT (Virtual). **Presenter**.

As contributory author:

- N. Baresi, **H. Holt**, N. Bernardini, X. Fu, M. Tisaev, Y. Gao, C. Bridges, A. Lucca Fabris, R. Armellin, P. Murzionak, and R. Kruzelecky, *Mission Analysis and Design of VMMO: the Volatile Mineralogy Mapping Orbiter*. In 31st AAS Spaceflight Mechanics Meeting Winter 2021 (Virtual).

- N. Baresi, N. Bernardini, E. Ciccarelli, X. Fu, **H. Holt**, and R. Armellin,. *Guidance, Navigation, and Control of Retrograde Relative Orbits Around Phobos*. 33rd International Symposium on Space Technology and Science, February 2022.

1.4.3 Workshops

- COMET-ORB On-Board Autonomy in Flight Dynamics (Virtual Seminar). **H. Holt**, N. Baresi, R. Armellin, *Trajectory Design using Lyapunov Control Laws and Reinforcement Learning*. **Presenter**.

1.4.4 rlc-toolbox

- Code Toolbox: `rlc-toolbox`, manual and code examples to accompany this thesis. Delivered to Surrey Satellite Technology Ltd (SSTL) May 2022. Currently not publicly available.

1.5 Outline of Thesis

The work presented in this thesis is organised as follows. First, an introduction to orbital mechanics and spacecraft control is given in Chapter 2. Lyapunov control theory and reinforcement learning are extensively discussed here. Then the proposed methodology is presented in Chapter 3. This includes the proof of concept, the specific Lyapunov control and RL implementation used. This chapter goes on to present the results for transfers in Keplerian dynamics, demonstrating the impact of state-weight dependence on Lyapunov stability. Chapter 4 is distinguished by the dynamical model used, and a novel cone-clock (CC) approach is presented (this will be denoted CC in the tables only). Robustness to stochastic errors is presented in Chapter 5, using a new training approach, and demonstrating the importance of the closed-loop nature of the control. Chapter 6 investigates the possibility of using this approach for higher-thrust, finite-burn trajectories. Although it is not designed for this, it is interesting to explore how far the Reinforced Lyapunov Controller can be stretched to cope with such transfers. The test case considered was provided by SSTL. Chapter 7 uses the developed technique to explore low-thrust many-revolution spiral transfers from a GTO to a Lunar polar orbit (LPO), in order to test how the control law copes when the dominant dynamics vary significantly. Finally, in Chapter 8 the conclusions are given and suggestions for future avenues of research are made.

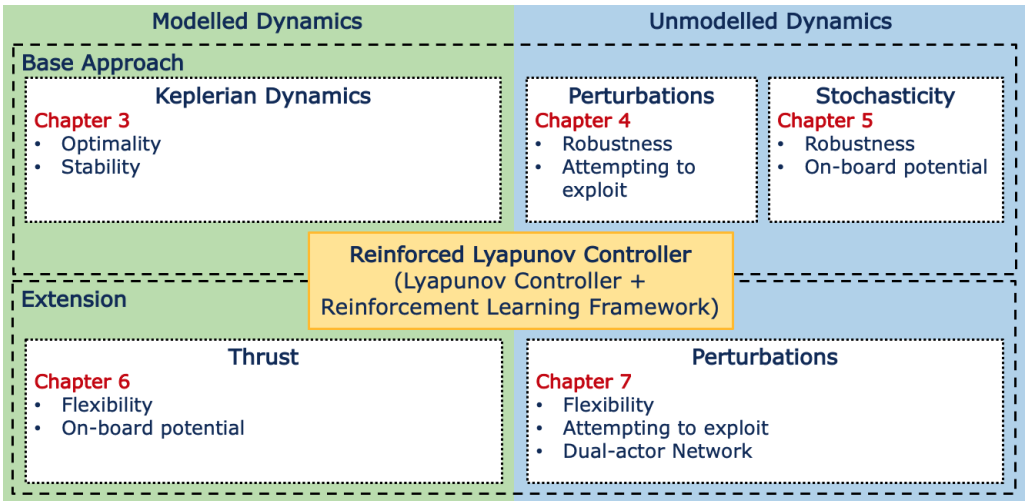


Figure 1.1: Thesis overview indicating how the chapters fit together and allow an investigation of the Reinforced Lyapunov Controller in different environments.

Table 1.1: Thesis overview and contributions

Chapter	Topic, Contribution, Research Output
3	**Topic:** Reinforced Lyapunov Controller **Contribution:** RL framework for making Lyapunov control laws state-dependent; Stability considerations for time-and state-dependent Lyapunov control laws; Analytical expression for the state-weight Jacobian to ensure stability; Investigating the impact of different actor-network architectures on stability. **Research Output:** [40–42]
4	**Topic:** Trajectory design with Perturbations **Contribution:** Demonstrating the closed-loop nature of the Reinforced Lyapunov Controller in perturbed dynamics (including J_2, 3^{rd}-body and eclipse effects); Proposing a training architecture with perturbed dynamics included; Novel cone-clock formulation for freeing control direction to both ensure Lyapunov stability for longer and the potential to exploit perturbations. **Research Output:** [42, 43]
5	**Topic:** Trajectory design with Stochastic Errors **Contribution:** Demonstrating closed-loop nature of Reinforced Lyapunov Controller to stochastic errors (including orbit insertion, orbit determination and thrust execution errors); Proposing a training architecture for including stochastic errors, Developing techniques for adding errors inside *ode113* integrators through cubic spline interpolation. **Research Output:** [42, 44]
6	**Topic:** Performance of Lyapunov Control with Finite-burn Manoeuvres **Contribution:** Investigating Lyapunov control for finite-burn manoeuvres; Incorporating operational constraints to Lyapunov controller such as maximum burn ΔV and coast arcs for momentum offloading.
7	**Topic:** Application of Reinforced Lyapunov Controller for Earth-Moon Spiral Transfers **Contribution:** Novel cone-clock formulation for freeing control direction to both ensure Lyapunov stability for longer and the potential to exploit perturbations; Forwards propagated Lyapunov control law for lunar-spiral transfers; Two-network RL architecture for two control policies; Rendezvous, patch-point and convergence criteria selection. **Research Output:** [43]

Figure 1.1 provides an overview of the thesis structure, and indicates how the different chapters fit together, whilst a breakdown of the key topics and main contributions for each chapter are given in Table 1.1. Performance in both modelled (Chapters 3 and 6) and unmodelled dynamics (Chapters 4, 5 and 7) is investigated. The base performance in Keplerian dynamics is used to investigate the optimality and stability of the approach. Then it is exposed to unmodelled dynamics in the form of perturbing accelerations, eclipse events and stochastic errors. This enables an investigation into the robustness of the approach, the suitability for on-board use, and methods for exploiting

the unmodelled dynamics to its advantage. Finally, Chapters 6 and 7 push the limits of this approach by increasing the thrust acceleration and complexity of the unmodelled dynamics. This investigates the flexibility of the approach, the on-board potential and the possibility of more complex RL architectures, stitching two actor networks together - as seen in Chapter 7.

Appendix A uses an alternative Lyapunov control law to provide the interested reader with a comparison to the results in Chapters 4 and 5. The basic Lyapunov control law is less optimal in its nominal form compared to the Q-law used, however it helps demonstrate the ability of the RL framework developed here to be used with multiple different underlying control laws. Throughout the thesis certain important results are highlighted in the tables to aid the reader.

1.5.1 Interpreting the Results

Throughout this thesis, results will be presented for time- and mass-optimal transfer scenarios. Where possible, results obtained either via optimal control theory in the literature will be provided for comparison, either using indirect or direct methods. Unfortunately access to such solutions was not possible for all scenarios considered in this thesis. However, in all cases, the performance of classical Lyapunov control laws is considered. Techniques available in the literature for improving the performance of such Lyapunov control laws, for example using particle swarm optimisation (PSO), will be presented as a benchmark. Whilst it will not be possible to match the performance of optimal control theory, it will be possible to compare the techniques developed in this thesis to classical and PSO-enhanced Lyapunov control laws. Thus when interpreting the results in this thesis, please note the following

- Optimality will refer to out-performing either the classical or PSO-enhanced Lyapunov control laws relative to a predetermined objective. It will not refer to optimal control theory solutions, which will remain the upper benchmark. Whilst it is desirable to get as close as possible to such solutions, the main objective is to improve the performance of Lyapunov control laws compared to the techniques in the literature.

- The term stability will refer to Lyapunov stability and robustness will refer to the closed-loop nature of the approach and its ability to handle uncertainties and perturbations.

- In instances where a large amount of information is provided within Tables, grey highlights will be used to indicate the most interesting solution. This is usually the method utilising the culmination of approaches discussed in the chapter, or the best performing result. In many cases, more intermediate results are provided for the interest of the reader and for context.

Chapter 2
Trajectory Design and Reinforcement Learning

In this chapter the relevant background theory and state-of-the-art in the literature is discussed. It begins by discussing orbital dynamics and introduces fundamental concepts such as Keplerian motion, classical orbital elements (COEs) and Gauss variational equations (GVEs). The problem of spacecraft control is discussed, within which the concept of Lyapunov control is presented. Then the ideas behind RL are introduced and the application of RL within astrodynamics is explored. For further details on these topics, the reader is referred to the excellent books by Schaub and Junkins [45], Betts [19] and Conway [14] for astrodynamics, and Sutton and Barto [46] on RL.

2.1 Orbital Dynamics

2.1.1 Orbital Motion

Orbital motion is primarily governed by the force of gravity. Newtonian gravity remains a very good first approximation to the observed behaviour of gravity and is still widely used as the first building block for orbital motion and solar system spaceflight [47]. Newton's law of gravitation is mathematically expressed as:

$$\boldsymbol{F} = -G\frac{m_1 m_2}{r^3}\boldsymbol{r}, \tag{2.1}$$

where $\boldsymbol{F}$ is the force exerted over a distance r between the two point masses m_1 and m_2, and G is the universal gravitational constant, which appears in both Newton and Einstein's theories of gravity. This equation holds for point masses. Complex gravitational fields can be constructed using this as a building block and integrating over various mass distributions and distances. In the case of spacecraft orbiting central massive bodies, one can assume that the spacecraft mass m_1 is negligible compared to the central body m_2 and $m_1 << m_2$. Hence, one can rewrite this equation using $\boldsymbol{F} = m_1\boldsymbol{a} = m_1\ddot{\boldsymbol{r}}$ and $\mu = Gm_2$ to give a particle's motion in a central inverse-square gravitational field subject to a disturbing acceleration $\boldsymbol{a}_d$:

$$\ddot{\boldsymbol{r}} = -\mu\frac{\boldsymbol{r}}{||\boldsymbol{r}||^3} + \boldsymbol{a}_d. \tag{2.2}$$

Keplerian motion describes the solutions to these equations when no disturbing ac-

celeration is present. Higher-fidelity dynamics refers to the presence of perturbations. In this work the disturbing acceleration $\boldsymbol{a}_d$ is split into the combination of the perturbing acceleration $\boldsymbol{a}_p$, coming from the inaccuracies in the dynamical model, and the control vector due to a spacecraft's engine $\boldsymbol{u}$.

2.1.2 Classical Orbital Elements

An orbit around a central body can be uniquely described by a set of five orbital elements. Although many different combinations of these exist, the COEs are the most intuitive. They comprise the semi-major axis a, eccentricity e, inclination i, right ascension of the ascending node (RAAN) Ω, and argument of periapsis ω. The true anomaly ν describes the instantaneous position along the orbit. See Fig. 2.1 for a visual representation. The periapsis represents the closest point to the focus and the apoapsis the point furthest away.

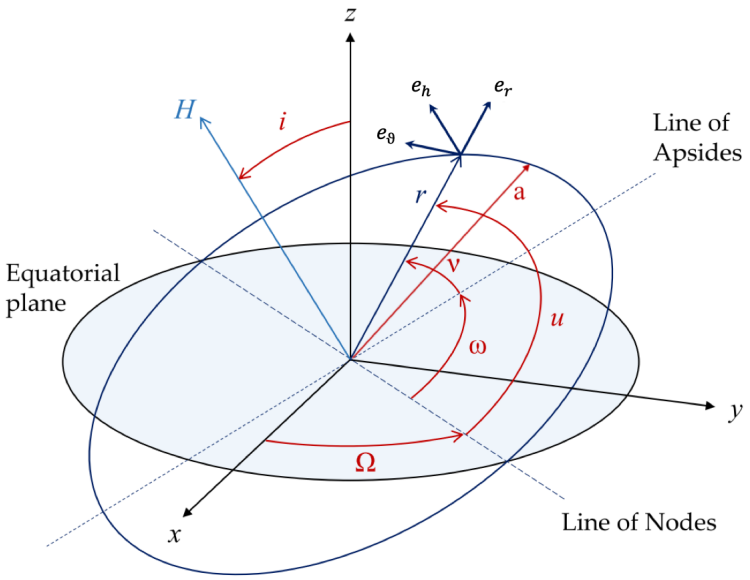


Figure 2.1: Diagram of the classical orbital elements, taken and modified from [48].

In order to describe a spacecraft's motion about a central body in terms of the COEs, one can convert Eqs. (2.2) into a set of variational equations in a, e, i, Ω, ω and the fast variable ν. This can be done in the spacecraft's $R\theta h$ coordinate system. Its origin lies at the spacecraft's centre of mass. The first axis, R, points in the direction of the instantaneous position vector relative to the central body, in this case the Earth. Meanwhile, the third axis h will point in direction of the orbit's angular momentum, which in this case lines up with the normal to the osculating orbital plane. θ is then defined to ensure a right-handed coordinate system. It is sometimes known as the radial-transverse-normal (RTN) frame or the local-vertical-local-horizontal (LVLH) frame [32]. When describing the disturbing acceleration $\boldsymbol{a}_d$ in the spacecraft's $R\theta h$ frame, the following GVEs [19, 47] are obtained:

$$\frac{da}{dt} = \frac{2a^2}{h}\left(e\sin(\nu)a_{dr} + \frac{p}{r}a_{d\theta}\right), \tag{2.3a}$$

$$\frac{de}{dt} = \frac{1}{h}\left(p\sin(\nu)a_{dr} + ((p+r)\cos(\nu) + re)a_{d\theta}\right), \tag{2.3b}$$

$$\frac{di}{dt} = \frac{r\cos(\omega+\nu)}{h}a_{dh}, \tag{2.3c}$$

$$\frac{d\Omega}{dt} = \frac{r\sin(\omega+\nu)}{h\sin(i)}a_{dh}, \tag{2.3d}$$

$$\frac{d\omega}{dt} = \frac{1}{he}\left(-p\cos(\nu)a_{dr} + (p+r)\sin(\nu)a_{d\theta}\right) - \frac{r\sin(\omega+\nu)cos(i)}{h\sin(i)}a_{dh}, \tag{2.3e}$$

$$\frac{d\nu}{dt} = \frac{h}{r^2} + \frac{1}{he}\left(-p\cos(\nu)a_{dr} - (p+r)\sin(\nu)a_{d\theta}\right), \tag{2.3f}$$

where $p = a(1-e^2)$ is the semi-latus rectum and $b = a\sqrt{1-e^2}$ is the semi-minor axis. The advantage of this formulation is two-fold. First, if the disturbing acceleration is zero, then the solution is the two-body motion and the slowly varying orbital elements a, e, i, Ω and ω are constant. Secondly, it isolates the disturbing acceleration $\boldsymbol{a}_d$ from the central gravitational acceleration [19]. In the case of low-thrust trajectories, where $\|\boldsymbol{u}\|$ and hence $\|\boldsymbol{a}_d\|$ are small, this is desirable as the orbital elements will continue to vary slowly. For later convenience, it is useful to express the dynamics as:

$$\begin{bmatrix}\dot{\boldsymbol{X}}\\ \dot{\nu}\end{bmatrix} = \begin{bmatrix}\boldsymbol{B}\\ \frac{p\cos(\nu)}{he} \quad \frac{-(p+r)\sin(\nu)}{he} \quad 0\end{bmatrix}\boldsymbol{a}_d + \begin{bmatrix}\boldsymbol{0}\\ \frac{h}{r^2}\end{bmatrix}, \tag{2.4}$$

where $\dot{\boldsymbol{X}}$ represents the dynamics of the COEs state. The matrix $\boldsymbol{B}$ represents the GVEs for the slow variables and is required later to compute the control:

$$\boldsymbol{B} = \begin{bmatrix} \frac{2a^2}{h}e\sin(\nu) & \frac{2a^2}{h}\frac{p}{r} & 0 \\ \frac{1}{h}p\sin(\nu) & \frac{1}{h}((p+r)\cos(\nu)+re) & 0 \\ 0 & 0 & \frac{r\cos(\omega+\nu)}{h} \\ 0 & 0 & \frac{r\sin(\omega+\nu)}{h\sin(i)} \\ -\frac{1}{he}p\cos(\nu) & \frac{1}{he}(p+r)\sin(\nu) & -\frac{r\sin(\omega+\nu)cos(i)}{h\sin(i)} \end{bmatrix}. \tag{2.5}$$

2.1.3 Modified Equinoctial Elements

When examining Eqs. (2.3), it is clear that singularities occur when $i = 0$ or $e = 0$. Hence, many alternative element sets exists to combat these various issues. A particularly useful set are the modified equinoctial elements (MEEs) p, f, g, h, k and fast variable L. These are related to the COEs as follows:

$$p = a(1-e^2), \tag{2.6a}$$

$$f = e\cos(\omega + \Omega), \tag{2.6b}$$

$$g = e\sin(\omega + \Omega), \tag{2.6c}$$

$$h = \tan(i/2)\cos(\Omega), \tag{2.6d}$$

$$k = \tan(i/2)\sin(\Omega), \tag{2.6e}$$

$$L = \Omega + \omega + \nu. \tag{2.6f}$$

If the disturbing acceleration $\boldsymbol{a}_d$ is described in the $R\theta h$ frame of the spacecraft, then the corresponding GVEs are [49]:

$$\frac{dp}{dt} = \frac{2p}{q}\sqrt{\frac{p}{\mu}}a_{d\theta}, \tag{2.7a}$$

$$\frac{df}{dt} = \sqrt{\frac{p}{\mu}}\sin(L)a_{dr} + \sqrt{\frac{p}{\mu}}\frac{1}{q}((q+1)\cos(L)+f)a_{d\theta} - \sqrt{\frac{p}{\mu}}\frac{g}{q}(h\sin(L)-k\cos(L))a_{dh}, \tag{2.7b}$$

$$\frac{dg}{dt} = -\sqrt{\frac{p}{\mu}}\cos(L)a_{dr} + \sqrt{\frac{p}{\mu}}\frac{1}{q}((q+1)\sin(L)+g)a_{d\theta} + \sqrt{\frac{p}{\mu}}\frac{f}{q}(h\sin(L)-k\cos(L))a_{dh}, \tag{2.7c}$$

$$\frac{dh}{dt} = \sqrt{\frac{p}{\mu}}\frac{s^2\cos(L)}{2q}a_{dh}, \tag{2.7d}$$

$$\frac{dk}{dt} = \sqrt{\frac{p}{\mu}}\frac{s^2\sin(L)}{2q}a_{dh}, \tag{2.7e}$$

$$\frac{dL}{dt} = \sqrt{\mu p}\left(\frac{q}{p}\right)^2 + \sqrt{\frac{p}{\mu}}\frac{1}{q}(h\sin(L)-k\cos(L))a_{dh}. \tag{2.7f}$$

Again, for later convenience, it is useful to express the dynamics in matrix form as:

$$\begin{bmatrix}\dot{\boldsymbol{X}}\\ \dot{L}\end{bmatrix} = \begin{bmatrix}\boldsymbol{B}\\ 0 \quad 0 \quad \sqrt{\frac{p}{\mu}}\frac{1}{q}(h\sin(L)-k\cos(L))\end{bmatrix}\boldsymbol{a}_d + \begin{bmatrix}\boldsymbol{0}\\ \sqrt{\mu p}\left(\frac{q}{p}\right)^2\end{bmatrix}, \tag{2.8}$$

where $\dot{\boldsymbol{X}}$ represents the dynamics of the MEEs state and the matrix $\boldsymbol{B}$ represents the GVEs for the slow variables:

$$\boldsymbol{B} = \begin{bmatrix} 0 & \frac{2p}{q}\sqrt{\frac{p}{\mu}} & 0 \\ \sqrt{\frac{p}{\mu}}\sin(L) & \sqrt{\frac{p}{\mu}}\frac{1}{q}((q+1)\cos(L)+f) & \sqrt{\frac{p}{\mu}}\frac{g}{q}(h\sin(L)-k\cos(L)) \\ -\sqrt{\frac{p}{\mu}}\cos(L) & \sqrt{\frac{p}{\mu}}\frac{1}{q}((q+1)\sin(L)+g) & \sqrt{\frac{p}{\mu}}\frac{f}{q}(h\sin(L)-k\cos(L)) \\ 0 & 0 & \sqrt{\frac{p}{\mu}}\frac{s^2\cos(L)}{2q} \\ 0 & 0 & \sqrt{\frac{p}{\mu}}\frac{s^2\sin(L)}{2q} \end{bmatrix}. \tag{2.9}$$

The MEEs can be used in the dynamical integrator instead of COEs to avoid the aforementioned issues.

2.1.4 Perturbations

In this work, perturbations refer to forces which have not been accounted for in two-body dynamics. A perturbing acceleration may result from an error in the central gravitational field approximation, such as the J_2 effect (due to Earth's oblateness) and third-body gravitational effects (e.g. from the Sun or Moon); or from drag and solar radiation pressure (SRP). In the case of spacecraft motion, its own control acceleration is referred to as a disturbing acceleration and differentiated from dynamical perturbations. These act to change the motion of the spacecraft from that which would be described by the unmodified Keplerian equations of motion. Figure 2.2 shows relative magnitude of some of these perturbations compared to the central gravitational field around Earth in low-Earth orbit (LEO). Drag decreases significantly with altitude as the atmospheric density decreases.

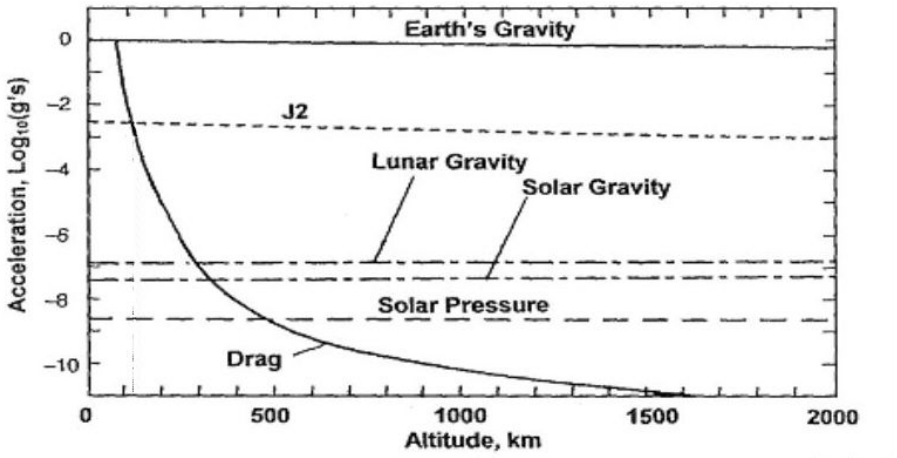


Figure 2.2: Plot showing the relative magnitudes of various perturbations with orbit altitude, taken from [50].

The choice of an appropriate dynamical model is a major issue in astrodynamics, and can significantly impact the usefulness of your results. A standard process is to start in the simplest gravitational environment appropriate (often two-body, two-body with J_2 or circular restricted 3-body problem (CR3BP)) and then add complexity later. As will become clear throughout the thesis, for planetocentric low-thrust trajectory design, this often begins with the two-body dynamics. Later sections will work with J_2 and third-body perturbations, along with eclipse effects. Atmospheric drag and solar radiation pressure are neglected at this stage.

2.1.4.1 Zonal Perturbations

The perturbing acceleration due to a non-spherical central body can be described by an aspherical potential function [51]. Described in the Earth-centred inertial (ECI) frame:

$$U(r,\phi_{gc},\lambda) = -\frac{\mu}{r} + U_{zonal}(r,\phi_{gc}) + U_{sectorial}(r,\phi_{gc},\lambda) + U_{tesseral}(r,\phi_{gc},\lambda), \tag{2.10}$$

where r is the radial distance from centre of mass of the Earth, ϕ_{gc} the geocentric latitude, λ the geographic longitude and μ the Earth's gravitational parameter. The zonal, sectorial and tesseral harmonics take the form:

$$U_{zonal}(r,\phi_{gc}) = \frac{\mu}{r}\sum_{i=2}^{\infty} J_i \left(\frac{R_e}{r}\right)^i P_i(\sin(\phi_{gc})), \tag{2.11a}$$

$$U_{sectorial}(r,\phi_{gc},\lambda) = \frac{\mu}{r}\sum_{i=2}^{\infty} (C_{ii}\cos(i\lambda) + S_{ii}\sin(i\lambda))\left(\frac{R_e}{r}\right)^i \cos(\phi_{gc})^i, \tag{2.11b}$$

$$U_{tesseral}(r,\phi_{gc},\lambda) = \frac{\mu}{r}\sum_{i=2}^{\infty}\sum_{j=1}^{i-1} \Big(C_{ij}\cos(i\lambda) + S_{ij}\sin(i\lambda)\Big)\left(\frac{R_e}{r}\right)^i P_{ij}(\sin(\phi_{gc})), \tag{2.11c}$$

where R_E the Earth's mean equatorial radius, P_{nm} the associated Legendre function of order m and degree n, and J_i, C_{nm} and S_{nm} the geopotential constant coefficients. Generally, the Earth's gravity field is assumed to be constant and the geopotential coefficients are provided by a gravity model.

As demonstrated in Fig. 2.2, the first zonal harmonic, known as J_2, is the strongest perturbing force up to GEO altitude and mainly causes secular variation of the longitude of the ascending node, argument of periapsis and mean anomaly. Using the expression for the second Legendre polynomial:

$$P_2(\sin(\phi_{gc})) = (3\sin^2(\phi_{gc}) - 1)/2, \tag{2.12}$$

the potential for first zonal harmonic J_2 becomes:

$$U_{J_2}(r,\phi_{gc}) = \frac{1}{2}\mu J_2\left(\frac{R_E^2}{r^3}\right)\Big(3\sin^2(\phi_{gc}) - 1\Big). \tag{2.13}$$

The perturbing accelerations due to J_2 are computed by taking the gradient: $\boldsymbol{a}_{J_2} = \nabla U_{J_2}$. In the $R\theta h$ frame, this perturbing acceleration is expressed as:

$$\begin{aligned}\boldsymbol{a}_{J_2} = \frac{-3\mu J_2 R_E^2}{r^4}\Bigg[&\left(\frac{1}{2} - \frac{3\sin^2 i \sin^2(\omega+\nu)}{2}\right)\hat{\boldsymbol{e}}_R \\ &+\sin^2 i \sin(\omega+\nu)\cos(\omega+\nu)\hat{\boldsymbol{e}}_\theta \\ &+\sin i \cos i \sin(\omega+\nu)\hat{\boldsymbol{e}}_h\Bigg].\end{aligned} \tag{2.14}$$

Here, $J_2 = 1.082635854\times10^{-3}$, R_E is the mean Earth radius (reference geoid) 6378.14 km and $n^2a^3 = \mu$. This full expression causes a short term variation of all orbital parameters, and a secular variation of Ω, ω and mean anomaly M_0, given by Eqs. (2.15), (2.16)

and (2.17) [32]. These are the average variation over one orbit:

$$\dot{\Omega}_{\text{av}} = -\frac{3}{2} n J_2 \left(\frac{R_E}{p}\right)^2 \cos i, \tag{2.15}$$

$$\dot{\omega}_{\text{av}} = \frac{3}{4} n J_2 \left(\frac{R_E}{p}\right)^2 \left(5\cos^2 i - 1\right), \tag{2.16}$$

$$\dot{M}_{0,\text{av}} = \frac{3}{2} n J_2 \sqrt{1-e^2} \left(\frac{R_E}{p}\right)^2 \left(\frac{3}{2}\sin^2 i - 1\right). \tag{2.17}$$

Note, if the inclination is non-zero, then Ω and ω will drift as a result of the J_2 perturbation. This phenomenon can be exploited to form a Sun-synchronous orbit (SSO), where the spacecraft passes over a particular location at the same local solar time each day [51] and see Section 4.4 for further discussion.

A first-order mapping from mean to osculating elements, and *vice versa*, is briefly explained here based on the Brouwer-Lyddane theory [52]. These concepts are used in Section 4.4 and Section 6.4. This mapping allows a translation from any osculating elements into mean elements where the short- and long-period oscillations are removed. As it is first order, only terms of the order of J_2 are retained. The dynamics can be written as:

$$\left[\dot{\boldsymbol{X}}\right] = \boldsymbol{\mathcal{B}}(\boldsymbol{X})\boldsymbol{u} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & n \end{bmatrix}^T, \tag{2.18}$$

where $\boldsymbol{X}$ represents the osculating COE state with the mean anomaly M is used instead of the true anomaly ν, the mean motion $n = \sqrt{\mu/a^3}$ and $\eta = \sqrt{1-e^2}$. The matrix $\boldsymbol{\mathcal{B}}(\boldsymbol{X})$ represents the full GVEs:

$$\boldsymbol{\mathcal{B}}(\boldsymbol{X}) = \begin{bmatrix} \frac{2a^2}{h} e\sin(\nu) & \frac{2a^2}{h}\frac{p}{r} & 0 \\ \frac{1}{h} p\sin(\nu) & \frac{1}{h}((p+r)\cos(\nu) + re) & 0 \\ 0 & 0 & \frac{r\cos(\omega+\nu)}{h} \\ 0 & 0 & \frac{r\sin(\omega+\nu)}{h\sin(i)} \\ -\frac{1}{he} p\cos(\nu) & \frac{1}{he}(p+r)\sin(\nu) & -\frac{r\sin(\omega+\nu)cos(i)}{h\sin(i)} \\ \frac{\eta(p\cos(\nu)-2re)}{he} & \frac{-\eta(p+r)\sin(\nu)}{he} & 0 \end{bmatrix}. \tag{2.19}$$

Brouwer and Lyddane showed that there exists an analytical transformation from the osculating orbit elements $\boldsymbol{X}$ to the mean elements $\bar{\boldsymbol{X}}$, denoted as $\bar{\boldsymbol{X}} = \xi(\boldsymbol{X})$. This analytical transformation is not shown in this thesis, but can be found in the book Schaub and Junkins, Appendix F [45]. Including the averaged J_2 expression the dynamics can be can be written as

$$\left[\dot{\bar{\boldsymbol{X}}}\right] = \frac{\partial \xi}{\partial \boldsymbol{X}} \boldsymbol{\mathcal{B}}(\boldsymbol{X})\boldsymbol{u} + \boldsymbol{A}(\bar{\boldsymbol{X}}), \tag{2.20}$$

where $\boldsymbol{A}(\bar{\boldsymbol{X}})$ is made up of the averaged effect of the J_2 perturbation written as:

$$\boldsymbol{A}(\bar{\boldsymbol{X}}) = \begin{bmatrix} 0 & 0 & 0 & \dot{\Omega}_{\text{av}} & \dot{\omega}_{\text{av}} & n + \dot{M}_{0,\text{av}} \end{bmatrix}^T. \tag{2.21}$$

For the purposes of developing a feedback control law, the matrix $\frac{\partial \xi}{\partial \bar{X}}$ is often approximated as an identity matrix as the off-diagonal terms are all of order J_2 or smaller and the diagonal terms are of order 1. This results in an approximated dynamics:

$$\left[\dot{\bar{X}}\right] \approx \mathcal{B}(\bar{X})\boldsymbol{u} + \boldsymbol{A}(\bar{X}), \tag{2.22}$$

which introduces an error on the order of J_2.

2.1.4.2 Third-body

Third-body gravitational accelerations from the Moon and the Sun can have a significant effect on orbital dynamics around Earth, particularly at higher altitudes. Using Cartesian coordinates, let the primary and secondary body have states $\boldsymbol{x}_p = \left[\boldsymbol{r}_p, \dot{\boldsymbol{r}}_p\right]^T$ and $\boldsymbol{x}_s = [\boldsymbol{r}_s, \dot{\boldsymbol{r}}_s]^T$, with gravitational parameters μ_p and μ_s. The standard values used for the dynamical parameters can be found in Koon *et al.* [53]. In an inertial model centred on the primary body, $\boldsymbol{x}_p = \boldsymbol{0}$. Hence, the motion of a spacecraft $\boldsymbol{x} = [\boldsymbol{r}, \dot{\boldsymbol{r}}]^T$ is given by:

$$\ddot{\boldsymbol{r}} = -\frac{\mu_p \boldsymbol{r}}{\|\boldsymbol{r}\|^3} - \frac{\mu_s(\boldsymbol{r} - \boldsymbol{r}_s)}{\|\boldsymbol{r} - \boldsymbol{r}_s\|^3} - \frac{\mu_s \boldsymbol{r}_s}{\|\boldsymbol{r}_s\|^3}. \tag{2.23}$$

In the rest of this work this system is known as the inertial CR3BP model. Although it does not require the secondary body to be in a circular orbit, in this work this is assumed. Thus it makes the same assumptions as the rotating CR3BP but is inertial and centred on either the Earth or the Moon. It will be important to note that the perturbing acceleration in the inertial reference frame is given by:

$$\boldsymbol{a}_p = -\frac{\mu_s(\boldsymbol{r} - \boldsymbol{r}_s)}{\|\boldsymbol{r} - \boldsymbol{r}_s\|^3} - \frac{\mu_s \boldsymbol{r}_s}{\|\boldsymbol{r}_s\|^3}. \tag{2.24}$$

2.1.4.3 Eclipse

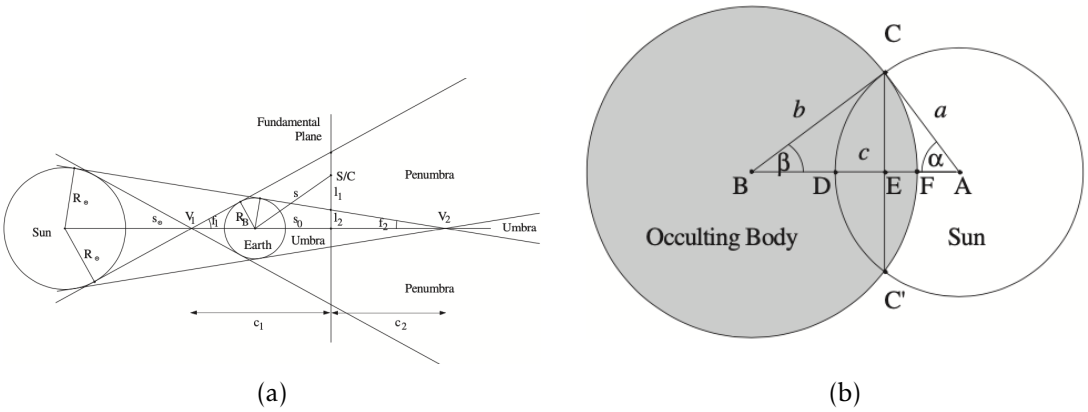
Incorporating eclipse effects requires knowledge of the positions of the spacecraft, Earth and Sun. The Sun's ephemeris is thus obtained using NAIF's *SPICE* Toolkit [54]. This can give the position of the sun as *xyz* coordinates in the ECI reference frame. A spacecraft's state in COEs can also be converted to *xyz* coordinates. Figure 2.3 shows the geometry assumed in this work: both the Earth and Sun are approximated as occulting discs. Assuming this set-up, full and partial eclipses can be modelled. The thrust of the spacecraft is modified by the eclipse parameter:

$$\eta_{\text{eclipse}} = 1 - \frac{\text{Area occulted}}{\text{Total Area}} = 1 - \frac{A}{\pi R_{\odot\,\text{app}}^2}, \tag{2.25}$$

where $R_{\odot\,\mathrm{app}}$ is the apparent radius of the Sun. The occulted area may be expressed as

$$\begin{aligned} A &= 2(A_{\mathrm{BCF}} - A_{\mathrm{BCE}}) + 2(A_{\mathrm{ACD}} - A_{\mathrm{ACE}}) \\ &= a^2 \arccos(x/a) + b^2 \arccos((c-x)/b) - cy\,, \end{aligned} \tag{2.26}$$

where Fig. 2.3 indicates the necessary geometrical arrangement.



(a) (b)

Figure 2.3: Eclipse geometry. The spacecraft is indicated as S/C. Taken from Montenbruck and Gill [55]

2.2 Spacecraft Control

In this section the concepts of controlling a spacecraft's motion, and subsequently designing manoeuvres and planning a trajectory are discussed. The sum of the external forces acting on a spacecraft is equal to the change in linear momentum of the whole system. This can be written as:

$$\sum \boldsymbol{f}_T = m\frac{dv}{dt} + v_e\frac{dm}{dt}. \tag{2.27}$$

Here the spacecraft has mass m, travelling at velocity v and ejects $\dot{m}$ fuel per unit time with an exhaust velocity v_e. A useful quantity to calculate is the velocity attained by a spacecraft after a certain amount of time, known as the ΔV. In an ideal scenario where all engine thrust goes into changing the spacecraft's velocity, the Tsiolkovsky rocket equation can be derived by integrating from the initial to final burn time [56]:

$$\int_{v_0}^{v_f} dv = \int_{t_0}^{t_f} \frac{T}{m}dt = v_e\int_{t_0}^{t_f}\frac{\dot{m}}{m}dt = -v_e\int_{m_0}^{m_f}\frac{dm}{m}, \tag{2.28}$$

$$\Delta V = -v_e \ln\left(\frac{m_f}{m_0}\right). \tag{2.29}$$

The specific impulse I_{sp} of a spacecraft engine is the total energy released by a propulsion system normalised by the weight of propellant used. For a constant thrust force T

and uniform propellant mass flow rate $\dot{m}$ in a gravitation field of strength g_0, this is:

$$I_{sp} = \frac{T}{\dot{m} g_0}. \tag{2.30}$$

The spacecraft mass m will therefore evolve according to:

$$\frac{dm}{dt} = -\delta_e \frac{T}{I_{sp} g_0}, \tag{2.31}$$

where δ_e will determine the throttle factor of the engine. If $\delta_e = 0$, the engine is off and the spacecraft will coast. $\delta_e = 1$ corresponds to the spacecraft using the maximum available thrust and intermediate values might result, for example, from partial eclipse events.

A spacecraft's thrust direction can be described by two angles, the in-plane angle ϕ_α and the out-of-plane angle ϕ_β. The resulting thrust direction in the spacecraft's $R\theta h$ frame is given by:

$$\hat{\boldsymbol{u}} = \begin{pmatrix} \sin(\phi_\alpha)\cos(\phi_\beta) \\ \cos(\phi_\alpha)\cos(\phi_\beta) \\ \sin(\phi_\beta) \end{pmatrix} = \begin{pmatrix} a_{dr} \\ a_{d\theta} \\ a_{dh} \end{pmatrix}, \tag{2.32}$$

where a_{dr}, $a_{d\theta}$ and a_{dh} are the disturbing accelerations. Hence, a low-thrust engine is usually controlled by continuously varying the direction and modulus of this acceleration vector. The resulting control vector can be expressed as:

$$\boldsymbol{u}(t) = \frac{\delta_e T}{m(t)} \hat{\boldsymbol{u}}(t). \tag{2.33}$$

2.2.1 Thrust-blending Control Laws

Thrust-blending control laws are a subset of CLFD that are bases on the principle of determining the thrust directions at each instance that produce the highest instantaneous rate of change in spacecraft state, and combined (or blending) these thrust directions together. Kluever [57] developed a thrust-blending approach for targeting the subset of a, e and i in 1998. This was formulated in COEs and involved blending thrust directions that maximised the rate of change of $\dot{a}, \dot{e}$ and $\dot{i}$ with respect to the in- and out-of-plane thrust angles respectively. They used varying weights for the transfer, chosen semi-heuristically as no analytical expressions are available for them. However, this approach was restricted to quasi-circular orbit transfers due to the semi-analytical expressions used.

Ruggiero [39] developed a similar thrust-blending technique, known as DAG. They determine the thrust angles that produce the highest instantaneous rate of change in the specific orbit elements. Targeting multiple orbital elements simultaneously is achieved by combining the multiple individual thrust directions [58]. An adaptive ratio, R_X, for each orbital element X is used to weight the thrust contribution for that orbital element.

Here X_0 and X_T represent the initial and target states respectively.

$$\boldsymbol{f}_T = \sum_X (1 - \delta_{X,X_T}) R_X \boldsymbol{f}_X \qquad \text{where} \qquad R_X = \frac{X_T - X}{X_T - X_0}. \tag{2.34}$$

Falck *et al.* [58] suggested that this control is often capable of achieving the targets but not necessarily in an optimal fashion. Hence, they go on to introduce directional weighting factors W_X which act to prioritise certain orbit elements. However, they claim to have experimented with time-varying and constant weighting factors and concluded that good results can be obtained by using simple constants, reducing the computational load required to tune them for an optimal solution.

The thrust effectivity term is defined as the ratio between the instantaneous rate of change of a specific COEs, dX/dt, and the maximum obtainable COEs variation at the specific true anomaly $dX/dt|_{\nu^{\max}}$, as:

$$\eta_X = \frac{dX/dt}{dX/dt|_{\nu^{\max}}}. \tag{2.35}$$

It acts to reduce the propellant consumption of the solution by increasing the efficiency threshold, but at the expense of time of flight. The required formulae for the individual optimal ΔV providing the maximum rate of change of each orbital element, along with the locations along each orbit and the manoeuvre efficiency terms are given in Falck *et al.* [58].

Finally, Fumenti *et al.* [59] use quasi-impulsive manoeuvres to approximate finite-burn transfers. They also define locations along the orbit where changing specific orbital elements will have little influence on the others. They develop two control strategies. The first, known at the OPM strategy, is used to correct inclination i without affecting the other COEs. This is done through a single out-of-plane control action and requires two coasting arcs, two burn arcs either side of the nominal burn location. A separate strategy for changing a and/or e using two separated in-plane control actions at apogee and perigee. This is known as OPM and consists of two separated in-plane control actions. The strategy computes the nominal point for an impulsive manoeuvre and uses this to construct a burn arc starting before and ending after this nominal point. They also adjust for the effects of zonal harmonics and atmospheric drag.

2.2.2 Lyapunov Control Laws

The trouble with some of the techniques presented so far is the lack of guaranteed stability. Stability plays an important role in spacecraft trajectory design and as such many CLFD control laws are based on Lyapunov control theory [32]. This provides them with an inbuilt stability, even if they are modified to an extent where they no longer meet the strict stability requirements.

2.2.2.1 Stability Definitions

Here some fundamental stability considerations of nonlinear dynamical systems are briefly introduced. A very useful outline of the stability definitions and study of non-linear stability can be found in Chapter 8 of [45]. This is the basis for this next section.

A nonlinear dynamical system with state $\boldsymbol{x}$ and control force $\boldsymbol{u}$ of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \tag{2.36}$$

can either be *autonomous* (if it does not explicitly depend on time) or *nonautonomous* (if it does explicitly depend on time). A *closed-loop* dynamical system can be obtained if an autonomous feedback control law $\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x})$ is available [32].

Definition 2.2.1 (Equilibrium states)**.** A state vector point $\boldsymbol{x}_e$ is said to be an equilibrium point of a dynamical system described by $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t)$ at time t_0 if $\boldsymbol{f}(\boldsymbol{x}_e, t) = 0 \; \forall t > t_0$. Thus, once a system reaches state $\boldsymbol{x}_e$, it will remain there for all time.

Definition 2.2.2 (Neighbourhood)**.** Given $\delta > 0$, a state vector $\boldsymbol{x}(t)$ is said to be in the neighbourhood $B_\delta(\boldsymbol{x}_r(t))$ of the state $\boldsymbol{x}_r(t)$ if $\|\boldsymbol{x}(t) - \boldsymbol{x}_r(t)\| < \delta$. Here if $\boldsymbol{x}_r(t)$ follows a prescribed motion, this is referred to as the nominal reference motion.

These two definitions redenable a discussion on what is meant by the term *stability*. There are three types worth mentioning here: *Lagrange* stability, *Lyapunov* stability, and *Asymptotic* stability. *Global* stability refers to these stability criteria holding for any initial state vector $\boldsymbol{x}(t_0)$.

Definition 2.2.3 (Lagrange Stability)**.** The motion $\boldsymbol{x}(t)$ is said to be Lagrange stable (or bounded) relative to a reference trajectory $\boldsymbol{x}_r(t)$ if there exists a $\delta > 0$ such that $\boldsymbol{x}(t) \in B_\delta(\boldsymbol{x}_r(t))$.

Definition 2.2.4 (Lyapunov Stability)**.** The motion $\boldsymbol{x}(t)$ is said to be Lyapunov stable (or stable) relative to a reference trajectory $\boldsymbol{x}_r(t)$ if for each $\boldsymbol{\epsilon} > 0$ there exists a $\delta(\boldsymbol{\epsilon}) > 0$ such that $\boldsymbol{x}(t_0) \in B_\delta(\boldsymbol{x}_r(t_0)) \Rightarrow \boldsymbol{x}(t) \in B_\epsilon(\boldsymbol{x}_r(t)) \forall t > t_0$.

Definition 2.2.5 (Asymptotic Stability)**.** The motion $\boldsymbol{x}(t)$ is said to be asymptotically stable (or stable) relative to $\boldsymbol{x}_r(t)$ if $\boldsymbol{x}(t)$ is Lyapunov stable and there exists a $\delta > 0$ such that $\boldsymbol{x}(t_0) \in B_\delta(\boldsymbol{x}_r(t_0)) \Rightarrow \lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{x}_r(t)$.

It is worth noting that *Lyapunov* stability makes no guarantees that the motion will actually converge to the target state, but rather that the motion will remain arbitrarily close to the desired target state. Figure 2.4 gives an illustration of a Lyapunov function in two dimensions x_1, x_2. It shows how a trajectory $\boldsymbol{x}(t)$ might evolve and how this "projects" onto the Lyapunov function $V(\boldsymbol{x})$.

In order to perform stability analysis, many nonlinear dynamical systems are linearised about a nominal reference motion $\boldsymbol{x}_r(t)$. This allows for standard linear control

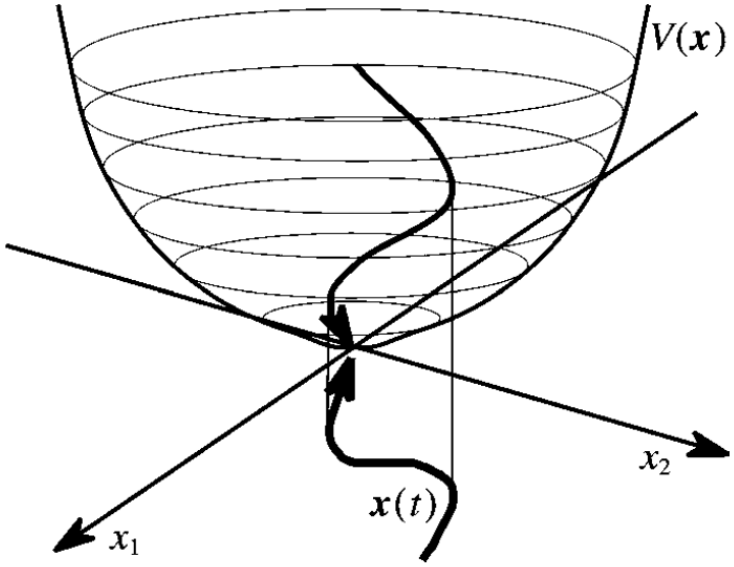


Figure 2.4: Illustration of a Lyapunov function taken from Schaub and Junkins [32]

techniques and stability theory to be applied. Usually, any stability claim resulting from this analysis will inherently be only a local stability claim. When reverting back to the nonlinear system, what one can usually claim is there exists a nonempty neighbourhood B_δ about the reference motion $\boldsymbol{x}_r(t_0)$ from which all nonlinear motions $\boldsymbol{x}(t)$ will be stable if $\boldsymbol{x}(t_0) \in B_\delta(\boldsymbol{x}_r(t_0))$. *Lyapunov's Indirect Method* can be used to make further claims on the stability of nonlinear systems.

Lyapunov's Indirect Method: Assume the linearised dynamical system is found to be

1. strictly stable (all eigenvalues of the linearisation matrix have negative real parts), then the nonlinear system is locally asymptotically stable.
2. unstable, then the nonlinear system is unstable.
3. marginally stable (all eigenvalues of the linearisation matrix have negative real parts and at least one is purely imaginary), then one cannot conclude anything about the stability of the nonlinear system without further analysis.

2.2.2.2 Lyapunov Control Theory

When considering non-linear stability, *Lyapunov's direct method* is often considered. It provides a rigorous and analytical stability claims of nonlinear system without solving the nonlinear differential equations. The approach involves studying the behaviour of a scalar, energy-like Lyapunov function, often referred to as V. *Lyapunov's direct method* leads to the basis of Lyapunov control theory, as it allows one to derive a control force which can ensure a nonlinear system is Lyapunov stable.

Before defining a Lyapunov function V, one needs to define both *positive definite* and *positive semidefinite* functions.

Definition 2.2.6 (Positive (Negative) Definite Functions)**.** A scalar continuous function $V(\boldsymbol{x})$ is said to be locally positive (negative) definite about $\boldsymbol{x}_r$ if $V(\boldsymbol{x}_r) = 0$ and there exists a $\delta > 0$ such that $\forall \boldsymbol{x} \in B_\delta(\boldsymbol{x}_r) \Rightarrow V(\boldsymbol{x}) > 0\,(V(\boldsymbol{x}) < 0)$ excluding $\boldsymbol{x} = \boldsymbol{x}_r$.

Definition 2.2.7 (Positive (Negative) Semidefinite Functions)**.** A scalar continuous function $V(\boldsymbol{x})$ is said to be locally positive (negative) semidefinite about $\boldsymbol{x}_r$ if $V(\boldsymbol{x}_r) = 0$ and there exists a $\delta > 0$ such that $\forall \boldsymbol{x} \in B_\delta(\boldsymbol{x}_r) \Rightarrow V(\boldsymbol{x}) \geq 0\,(V(\boldsymbol{x}) \leq 0)$ excluding $\boldsymbol{x} = \boldsymbol{x}_r$.

Hence, to prove the stability of a dynamical system, particular positive definite functions known as Lyapunov functions are sought.

Definition 2.2.8 (Lyapunov Function)**.** A scalar function $V(\boldsymbol{x})$ is a Lyapunov function for the dynamical system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ if it is continuous and there exists a $\delta > 0$ such that for any $\boldsymbol{x} \in B_\delta(\boldsymbol{x}_r)$

1. $V(\boldsymbol{x})$ is a positive definite function about $\boldsymbol{x}_r$.
2. $V(\boldsymbol{x})$ has continuous partial derivatives.
3. $\dot{V}(\boldsymbol{x})$ is negative semidefinite.

Definition 2.2.9 (Asymptotic Stability (for Lyapunov Functions))**.** Assume $V(\boldsymbol{x})$ is a Lyapunov function about $\boldsymbol{x}_r$ for $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$; then the system is asymptotically stable if

1. the system is stable about $\boldsymbol{x}_r$.
2. $\dot{V}(\boldsymbol{x})$ is negative definite about $\boldsymbol{x}_r$.

For convenience one usually examines the stability about the origin by making a coordinate transformation to ensure the equilibrium point or nominal reference motion is at the origin. The derivative $\dot{V}(\boldsymbol{x})$ is given by

$$\dot{V}(\boldsymbol{x}) = \frac{\partial V}{\partial \boldsymbol{x}} \dot{\boldsymbol{x}} = \frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}). \tag{2.37}$$

If the Lyapunov function $V(\boldsymbol{x})$ is radially unbounded (i.e., $V(\boldsymbol{x}) \to \infty$ as $\|\boldsymbol{x}\| \to \infty$), then the stability claims are globally valid. Also, it is worth noting that if for one Lyapunov function this theorem does not hold it does not necessarily follow that the system is unstable. It simply means that it is inconclusive regarding the stability or instability, and there may exist another Lyapunov function which can prove stability. The main issue with Lyapunov's direct method for proving the stability of nonlinear systems is the difficulty in finding an appropriate function. This is a non-trivial matter and establishing Lyapunov functions is very much an art.

For a trajectory optimisation problem, a suitable choice for the control can be found by minimising $\dot{V}$, whilst stability is guaranteed if $\dot{V}$ remains negative definite. Note Eqs. (2.5) can be used to describe the dynamics of the system as $\dot{\boldsymbol{X}} = \boldsymbol{B}\boldsymbol{u}$, where $\boldsymbol{u}$ represents the control vector and acts as a disturbing acceleration. For an autonomous system,

$$\dot{V}(\boldsymbol{X} - \boldsymbol{X}_{\mathrm{T}}) = \frac{\partial V}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} = \frac{\partial V}{\partial \boldsymbol{X}}\boldsymbol{B}\boldsymbol{u}. \tag{2.38}$$

In order to drive the system in state $\boldsymbol{X}$ towards the equilibrium state $\boldsymbol{X}_{\mathrm{T}}$, one can minimise $\dot{V}$ by setting the control such that

$$\boldsymbol{u} = -\boldsymbol{B}^T \left(\frac{\partial V}{\partial \boldsymbol{X}}\right)^T. \tag{2.39}$$

This requires freedom on the magnitude of the control. As this is normally restricted by the on-board engine, Eq. (2.39) is instead used to define the best direction to minimise (make as negative as possible) the Lyapunov function. In reference to the Primer vector theory [15], this thesis will refer to this direction as:

$$\hat{\boldsymbol{p}} = \frac{\boldsymbol{B}^T \left(\frac{\partial V}{\partial \boldsymbol{X}}\right)^T}{\left\| \boldsymbol{B}^T \left(\frac{\partial V}{\partial \boldsymbol{X}}\right)^T \right\|}. \tag{2.40}$$

Hence a unit vector in a direction opposite to $\hat{\boldsymbol{p}}$ will minimise the Lyapunov derivative, which is thus given by

$$\dot{V} = -\left\| \boldsymbol{B}^T \left(\frac{\partial V}{\partial \boldsymbol{X}}\right)^T \right\|. \tag{2.41}$$

For a spacecraft, the control vector is conventionally chosen as $\boldsymbol{u} = -\frac{T}{m}\hat{\boldsymbol{p}} = -f\,\hat{\boldsymbol{p}}$, where T is the engine thrust and m is the spacecraft mass.

2.2.2.3 Q-law

One of the most well known and versatile Lyapunov control laws is the Proximity Quotient, better known as the Q-law, developed by Petropoulos [35, 36]. It is best thought of as a weighted, squared summation of the time required to change the current state $\boldsymbol{X} = [a, e, i, \Omega, \omega]^{\mathrm{T}}$ to the target state $\boldsymbol{X}_T = [a_T, e_T, i_T, \Omega_T, \omega_T]^{\mathrm{T}}$. Through convention, the Lyapunov function is denoted Q, but represents the V function used above. It is defined as:

$$Q = (1 + W_P P) \sum_X W_X S_X \left(\frac{\delta X}{\max_\nu(\dot{X})}\right)^2, \quad X = a, e, i, \Omega, \omega, \tag{2.42}$$

where W_P is a weighting factor, P is a penalty function, W_X are weighting factors, S_X scaling functions and $\delta X = X - X_T$ for $X = a, e, i$ and $\delta X = \arccos(\cos(X - X_T))$ for $X = \Omega, \omega$. The expressions $\max_\nu(\dot{X})$ are the maximum rate of change of each classical element over the current osculating orbit.

As discussed in Petropoulos *et al.* [36], analytical expressions for the maximum rates of change over both the true anomaly and thrust directions achievable for each orbit elements, although this is only true for ω if the in-plane and out- of-plane motions are each considered individually. For $\max_\nu(\dot{\omega})$ a weighted average of in-plane and out-of-plane rates: $\max_\nu(\dot{\omega}) = (\max_\nu(\dot{\omega}_i) + b_{\text{petro}} \max_\nu(\dot{\omega}_o))/(1 + b_{\text{petro}})$ is required. The maximum rate of change of each orbital element is computed by setting $\partial \dot{X}/\partial \nu = 0$.

The penalty function P and scaling function S_X are:

$$P = \exp\left[k_{\text{petro}}\left(1 - \frac{r_p}{r_{\text{petro}}^{\text{p-min}}}\right)\right], \qquad S_X = \begin{cases} \left[1 + \left(\frac{X - X_T}{m_{\text{petro}} X_T}\right)^{n_{\text{petro}}}\right]^{1/r_{\text{petro}}} & X = a, \\ 1 & X = e, i, \Omega, \omega. \end{cases} \tag{2.43}$$

User defined parameters are nominally set to values of $k_{\text{petro}} = 100$, $m_{\text{petro}} = 3$, $n_{\text{petro}} = 4$, $r_{\text{petro}} = 2$ and $b_{\text{petro}} = 0.01$. The penalty function enforces a minimum-periapsis-radius constraint throughout the transfer, $r_{\text{petro}}^{\text{p-min}} = 6578$ km. It does so by adding a term which increases exponentially as the periapsis approaches $r_{\text{petro}}^{\text{p-min}}$, artificially increasing the Lyapunov function value. The scaling function is used to help prevent non-convergence to the target orbit. It was noted by Petropoulos *et al.* [36] that this Lyapunov function formulation can also have a minimum as $a \to \infty$. As such, this scaling function will artificially increase the Lyapunov function as $X - X_T$ increases, preventing the control from seeking $a \to \infty$.

The control minimises the rate of change of this Lyapunov function, hence resulting in a feasible trajectory for the given Lyapunov function formulation. Individual orbital element changes can be targeted, as well as a specific subset of elements. The weights W_X can be used to prioritise targeting particular elements X, changing the transfer characteristics.

Coasting can be introduced using effectivity parameters and thresholds. These attempt to quantify the effectivity of changing an orbital parameter at a given point in an orbit compared to the optimum point for changing the same orbital parameter. Definitions for both the absolute and relative effectivity parameters are:

$$\eta_a(\dot{Q}) = \frac{\min_{\phi_\alpha, \phi_\beta}(\dot{Q})}{\min_\nu(\min_{\phi_\alpha, \phi_\beta}(\dot{Q}))}, \tag{2.44}$$

$$\eta_r(\dot{Q}) = \frac{\min_{\phi_\alpha, \phi_\beta}(\dot{Q}) - \max_\nu(\min_{\phi_\alpha, \phi_\beta}(\dot{Q}))}{\min_\nu(\min_{\phi_\alpha, \phi_\beta}(\dot{Q})) - \max_\nu(\min_{\phi_\alpha, \phi_\beta}(\dot{Q}))}, \tag{2.45}$$

where ϕ_α and ϕ_β are the in-plane and out-of-plane angles of the thrust vector. ϕ_α lies in the orbit plane and is measured with respect to the transverse direction, θ, in the $R\theta h$ reference frame - see Section 2.1.2. The positive direction is defined away from the gravitational source. ϕ_β is the angle between the control vector and the orbit plane, with the positive in the direction of the angular momentum, h.

The term $\min_{\phi_\alpha,\phi_\beta}(\dot{Q})$ is computed at the current true anomaly, ν_0. This is also the value given by Eq. (2.41) The term $\min_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}))$ is computed numerically by scanning through the possible true anomaly ν values to find the maximum and minimum $\dot{Q}$ for the particular osculating orbit. Thus, assuming the current elements a, e, i, Ω and ω remain constant, the GVEs are a function of true anomaly, $B(\nu)$. Hence, $\min_{\phi_\alpha,\phi_\beta}(\dot{Q})$, $\min_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}))$ and $\max_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}))$ are calculated as follows:

$$\min_{\phi_\alpha,\phi_\beta}(\dot{Q}) = \left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)\boldsymbol{B}(\nu_0)\hat{\boldsymbol{u}} = -\left\|\boldsymbol{B}(\nu_0)^T\left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)^T\right\|. \tag{2.46}$$

$$\min_\nu\left(\min_{\phi_\alpha,\phi_\beta}(\dot{Q})\right) = \min_\nu\left(\left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)\boldsymbol{B}(\nu)\hat{\boldsymbol{u}}\right) = \min_\nu\left(-\left\|\boldsymbol{B}(\nu)^T\left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)^T\right\|\right). \tag{2.47}$$

$$\max_\nu\left(\min_{\phi_\alpha,\phi_\beta}(\dot{Q})\right) = \max_\nu\left(\left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)\boldsymbol{B}(\nu)\hat{\boldsymbol{u}}\right) = \max_\nu\left(-\left\|\boldsymbol{B}(\nu)^T\left(\frac{\partial Q}{\partial \boldsymbol{X}}\right)^T\right\|\right). \tag{2.48}$$

For the later two, one has to scan through the possible true anomaly ν values to find the maximum and minimum $\dot{Q}$ for the particular osculating orbit. It is noted that an alternative approach first found by Varga *et al.* [60] can be used which now avoids any numerical derivations thanks to recent work by Shannon *et al.* [61], however this is not implemented here. The lack of numerical derivations doesn't change the resulting behaviour, but rather ensures all expressions have analytical solutions. Studies [62] have also shown that, when varying other Q-law parameters in addition to the effectivity, there is little difference in performance between relative and absolute effectivity, and instead the specific transfer will determine which is more applicable. Hence, in this thesis the absolute effectivity is used. Finally, threshold values can be set such that engine thrusting only occurs when $\eta_a \geq \eta_a^t$ and $\eta_r \geq \eta_r^t$. By design this will occur at least instantaneously once per orbit.

All this increases the Q-law's versatility, and distinguishes it from many alternative techniques [31]. Its performance is comparable with Ruggiero *et al.*'s DAG method [39], implemented by Falck *et al.* [58], but its firmer grounding in Lyapunov stability gives it more robustness, and several comparisons have shown both are particularly good. However, there are many design parameters that would originally be left to the user's discretion, which can significantly affect the performance and optimality of the method. In particular, the weights determine the contribution of each targeted orbital element.

2.2.2.4 Changing individual Orbital Parameters

As mentioned above, analytical expressions for varying individual COEs can be obtained. Naturally, there are efficient and inefficient ways too induce desired changes. Starting

from the GVEs, Eqs. (2.5), the optimal thrust angles for a given location ν can be found by setting $\partial\dot{X}/\partial\phi_\alpha = 0$ and $\partial\dot{X}/\partial\phi_\beta = 0$. The optimal location along the orbit for each element is then found by setting $\partial\dot{X}/\partial\nu = 0$ and solving for ν. Through the coupling of the GVEs, the application of thrust in one direction can affect more than one orbital element. These maximum rates of change for the semi-major axis a, eccentricity e, inclination i and RAAN Ω are found to be as follows [31]:

$$\dot{a} = 2|a_d|\sqrt{\frac{a^3(1+e)}{\mu(1-e)}}, \tag{2.49a}$$

$$\dot{e} = \frac{2p|a_d|}{h}, \tag{2.49b}$$

$$\dot{i} = \frac{p|a_d|}{h\left(\sqrt{1-e^2\sin^2(\omega)} - e|\cos(\omega)|\right)}, \tag{2.49c}$$

$$\dot{\Omega} = \frac{p|a_d|}{h\sin(i)\left(\sqrt{1-e^2\cos^2(\omega)} - e|\sin(\omega)|\right)}. \tag{2.49d}$$

The expression is more complex for the argument of periapsis because ω can be changed by both in-plane and out-of-plane components. Petropoulos [36] handled this by treating the two components separately and combining them. Once the optimal control angles ϕ_α, ϕ_β are found, the in-plane maximum rate-of-change is given by

$$\dot{\omega}_i = \frac{|a_d|}{eh}\sqrt{p^2\cos^2(\nu_i) + (p+r_i)^2(1-\cos^2(\nu_i))}, \tag{2.50}$$

where the optimal orbit location is:

$$\cos(\nu_i) = \left[\frac{1-e^2}{e^3} + \sqrt{\frac{1}{4}\left(\frac{1-e^2}{e^3}\right)^2 + \frac{1}{27}}\right]^{\frac{1}{3}} - \left[-\frac{1-e^2}{e^3} + \sqrt{\frac{1}{4}\left(\frac{1-e^2}{e^3}\right)^2 + \frac{1}{27}}\right]^{\frac{1}{3}} - \frac{1}{e}, \tag{2.51}$$

and

$$r_i = \frac{p}{1+e\cos(\nu_i)}. \tag{2.52}$$

The out-of-plane component uses different optimal control angles ϕ_α, ϕ_β, giving

$$\dot{\omega}_o = \dot{\Omega}|\cos(i)| = \frac{p|a_d||\cos(i)|}{h\sin(i)\left(\sqrt{1-e^2\cos^2(\omega)} - e|\sin(\omega)|\right)}. \tag{2.53}$$

These do not provide the exact solution, but allows one to find an approximate analytical solution by combining the in-plane and out-of-plane components:

$$\dot{\omega} = \frac{\dot{\omega}_i + b_{petro}\dot{\omega}_o}{1+b_{petro}}, \tag{2.54}$$

where b_{petro} is a non-negative constant nominally taken as 0.01. A more complete dis-

cussion of this can be found in Petropoulos *et al.* [36] and Hatten [31].

2.2.2.5 ΔV-law

A new approach was recently presented by Locoche [63], which uses the Q-law's time-to-go formulation as inspiration for developing a ΔV-to-go control law. In the paper it is highlighted that analytical formulations for low-thrust orbit transfer ΔV are not readily available, but that it is possible to design empirical ΔV formulations by evaluating the ΔV of basic transfers: namely non-coplanar circular orbit transfers and non-coplanar eccentric orbit transfers.

Edelbaum [17] first covered these constant-acceleration circular to inclined circular orbits with with:

$$\Delta V_{a,i} = \sqrt{V_c^2 - 2V_c V_{cf} \cos\left(\frac{\pi \Delta i}{2}\right) + V_{cf}^2}, \tag{2.55}$$

where V_c is the initial circular velocity, V_{cf} is the final circular velocity and $\Delta i = i_f - i$ is the difference between the final and current inclination. Likewise, a similar expression for low-thrust transfers between circular orbits with RAAN change $\Delta\Omega$:

$$\Delta V_{a,\Omega} = \sqrt{V_c^2 - 2V_c V_{cf} \cos\left(\frac{\pi \sin(i) \Delta\Omega}{2}\right) + V_{cf}^2}, \tag{2.56}$$

The ΔV required to apply an eccentricity correction Δe at a constant semi-major axis is also available [64]. Assuming the in-plane thrust is perpendicular to the apsidal line and the initial and final orbits remain elliptical ($e < 1$) then:

$$\Delta V_{e,i} = \frac{2}{3} V_C \frac{\left|\left(\arcsin(e) - \arcsin(e_f)\right)\right|}{\cos\left(|\phi_\beta(e, i, e_f, i_f)|\right)}. \tag{2.57}$$

Here, e and e_f are the initial and target eccentricity values, i and i_f are the initial and target inclination values and ϕ_β is the out-of-plane angle that helps drive i and e to reach their target values simultaneously:

$$\tan(|\phi_\beta|) = \left| \frac{3\pi \Delta i}{4\cos(\omega)\left[\ln\left(\frac{e_f+1}{e_f-1}\right) + \ln\left(\frac{e-1}{e+1} - \Delta e\right)\right]} \right|. \tag{2.58}$$

Using these foundations, a Lyapunov function L is proposed, which is a combination

of the Eqs. (2.55)-(2.58) and some weighting parameters $\lambda_{e1}, \lambda_{e2}, \lambda_{ai}, \lambda_{ei}, \lambda_{a\Omega}$ and λ_{ω}:

$$L = \left[V_c^2 - 2V_c V_{cf} \cos\left(\frac{\pi\Delta\sigma}{2}\right) + V_{cf}^2 \right] + \frac{4}{9}\lambda_{e1} \left[\frac{[(1-\lambda_{e2})V_C + \lambda_{e2} V_{cf}] \left| \left(\arcsin(e) - \arcsin(e_f)\right) \right|}{\cos\left(|\phi_\beta|\right)} \right]^2, \tag{2.59}$$

with

$$\Delta\sigma = \sqrt{[\lambda_{ai}\Delta i]^2 + [\lambda_{a\Omega}\sin(i)\Delta\Omega]^2}, \tag{2.60}$$

$$\tan(|\phi_\beta|) = \left| \frac{3\pi\lambda_{ei}\Delta i}{4\cos(\lambda_\omega\omega)\left[\ln\left(\frac{e_f+1}{e_f-1}\right) + \ln\left(\frac{e-1}{e+1} - \Delta e\right)\right]} \right|. \tag{2.61}$$

This ΔV-law appears to have very good results once the parameters $\lambda_{e1}, \lambda_{e2}, \lambda_{ai}, \lambda_{ei}, \lambda_{a\Omega}$ and λ_ω are optimised, although these numbers can be reduced depending on the specific transfer scenario.

2.2.2.6 Lyapunov Control Law Extensions

Due to its effective time-optimal formulation, many authors have used the Lyapunov control as a basis for a variety of applications. Maddock and Vasile [65] extended a Lyapunov-based control law to handle both solar radiation pressure and 3$^{\text{rd}}$-body effects, however they noted trouble with over-shooting and control law chatter.

Baresi *et al.* [66] derived a Lyapunov controller using relative orbital elements for orbit maintenance around the Martian moon Phobos. In order to account for the gravitational perturbations, they derive the control in a similar fashion to [32]. Epenoy and Pérez-Palau [67] designed a Lyapunov-based control law combined with an invariant manifold approach for Earth-Moon transfers. This also included solar radiation pressure and 3$^{\text{rd}}$-body effects, however the control direction computed assumed the non-perturbed dynamics control terms will dominate and hence still provide a stable control. They conclude they could improve the Lyapunov-controller by varying the weights. Similarly, Jagannatha *et al.* [68] used a backward propagated Q-law to design trajectories from GTO to Earth-Moon halo orbits by utilising a patch point on a low-energy stable manifold. Peterson *et al.* [69] used a sequence of osculating target elements to design Lyapunov guidance in cis-lunar space.

One of the main uses for heuristic control laws is as initial guesses for indirect or direct optimisation approaches. Shannon *et al.* [70] combined an evolutionary algorithm with a forwards and backwards propagated Q-law to provide an initial guess for direct collocation and obtain optimal trajectories. Although the dynamics were high-fidelity, the Q-law's internal dynamics were two-body. This test case provides inspiration for Chapter 7. They extended this approach to Lunar swing-by escape trajectories, demon-

strating the huge potential for low-thrust exploration of interplanetary space [71].

Impulsive transfers, whilst not the priority in this thesis, remain important as they often provide the theoretical time-optimal and fuel-optimal extremals, and also help determine the reachability domain from a given orbit. Dating back to 1967, Edelbaum [72] presented a central problem in astrodynamics: determining the number, time, direction and magnitude of velocity impulses that minimise the total impulse. Whilst this problem has been solved for single-revolution transfers, it remains very challenging for N-impulse multi-revolution trajectories even to this day [73]. The standard approach is to start from a two-impulse solution (Lambert's solution) and then extend this to N-impulses. However, Taheri and Junkins [73] have demonstrated a direct theoretical connection between optimal finite-thrust continuous control and an optimal sequence of velocity impulses. They show the fundamental minimum-thrust solution enables them to construct a switching surface that helps indicate the desired location and magnitude of finite-burn arcs, and in the limiting case of $T \to \infty$ reveals the associated impulsive solution. The minimum-thrust case is solved using an indirect method and test cases including Earth-to-Mars, Earth-to-Asteroid and GTO-GEO transfers are considered. These transfers appear to be on the order of ~ 10 revolutions. Hence, Lyapunov-based control approaches with higher control acceleration could be used in aiding the search for optimal finite-burn manoeuvre sequences. This is subsequently considered in Chapter 6.

Gao and Li [37] identified the major problem of selecting a Lyapunov controller's weights, and used a COV-based method to create time-varying weights for Lyapunov control. This required a mapping between the parameterised control law and the Lyapunov control law, but this cannot strictly guarantee stability during the entire transfer period. They also suggest that optimising Lyapunov weights is difficult due to the chattering near to target orbits and the sensitivity of the transfer to the weight values.

Varga *et al.* [60] used a non-dominating GA to optimise the Q-law design parameters for a variety of Earth orbit transfers, however, the design parameters remained fixed throughout the transfer. Yang *et al.* [74] used a NN and improved cooperative evolutionary algorithm (ICEA) optimiser to make the design parameters of a Q-law state-dependent. However, although they demonstrate robustness to J_2 and on-orbit uncertainties, they did not take this further and explore the opportunities for actively exploiting perturbations.

2.2.3 Discussion

After this extensive discussion of Lyapunov control laws, there are a few key takeaway messages:

- Deriving a Lyapunov function is not trivial, and ensuring it is close-to-optimal is harder still.

- Multiple candidate Lyapunov functions exist with varying success, the most popular being the Q-law.
- Lyapunov functions are often derived in low-fidelity dynamics, and often handle perturbing accelerations by cancelling them out.
- They can provide useful stability criteria which makes them attractive for potential on-board use.

However, there is lots of promise for further developing Lyapunov control laws, given their potential for both fast mission analysis and potential on-board guidance. One of the main ways to improved them is using evolutionary algorithms to tune their user-defined parameters. It has also been noted there is significant potential is allowing these parameters to varying throughout the transfer, as in Gao [37] and Yang [74].

Given the good performance and extensive use observed in the literature, the Q-law was used in this investigation. However, the techniques developed here are not restricted to the Q-law, and can use a variety of Lyapunov control laws. This is demonstrated in Appendix A.1. Finally, it is important to note that the control and integration do not need be computed using the same orbital element state description. Throughout the remainder of this thesis, the control will be computed in COEs. Whilst alternative formulations of the control laws discussed exist in different orbit element descriptions, interpreting their behaviour and the proposed RL framework is arguably most intuitive in COEs for the applications considered here.

2.3 Global Optimisers

In recent years, there have been many advances in the use of global optimisation approaches to approximate solutions of spacecraft trajectory optimisation problems. These are also sometimes known as metaheuristics [12] and act as a way of improving the optimality of a given solution, either for indirect, direct or heutristic approaches, by changing the values of the parameters. Evolutionary algorithms are by far the most studied in astrodynamics. These include the GAs and PSO.

Evolutionary algorithms use the principles found in natural evolution processes to find an optimal set of discrete parameters. The main advantages of these optimisation techniques is their global nature. Unlike gradient based methods, they allow a more thorough exploration of the search space and are less susceptible to converging to local optima. In addition, no initial guess is required to start an optimisation. A random initialised parameter set is sufficient to converge to a solution, making them suitable for problems where initial guesses are difficult to provide. They require the problem to be described by a smaller number of parameters compared to NLP methods [14]. Either the problem is naturally described by a finite set of elements, such as with impulsive transfers, or by discretising the problem using polynomial parameterisations to describe

the time evolution of parameters (such as patched Chebyshev polynomials). An alternative is to use SB methods to describe low-thrust arcs, and optimise the parameters of the shapes.

The multi-objective nature of spacecraft trajectory design lends itself to evolutionary algorithms, and they still provide many of the benchmarks in the area [30]. As mentioned above, these techniques have become very common for problems ranging from rendezvous and typical transfers to interplanetary trajectories, gravity assists and Libration point orbits.

In their survey of trajectory optimisation techniques, Shirazi *et al.* [12] suggest GAs and PSOs are the first choice for most spacecraft optimisation problems. GAs can be used to solve constrained and unconstrained optimisation problems and are are inspired by natural selection. They simulate evolution by reproduction, inheritance and selection mechanisms. Starting with a population of individual solutions, the GA evolves this population over successive generations toward an optimal solution based on an objective function. The algorithm randomly selects individuals as parents to produce children for subsequent generations. Four types of children are created: elite children are the individuals with the current best objective value; crossover children are a combination of two parent vectors; mutation children are created by introducing random change to a single parent; and finally mutation and crossover are often combined to form the final set of children. GAs are very useful for solving problems where the objective function is discontinuous, stochastic or highly non-linear.

PSOs are another type of meta-heuristic approach inspired by flocks of birds, schools of fish and other intelligent swarms [30]. Like GAs, PSOs are a population-based algorithm. One can imagine the objective function as a surface in N-dimensional space, where N is the size of the optimisation vector. A population of particles are dropped onto the objective space, each with a given N-dimensional state. PSO algorithms then move these particles around the objective space, evaluating the objective function at each step. The algorithm uses knowledge of the particles' momenta and velocities to update the movement of each particle throughout the objective space, ideally towards local or global best solutions. PSOs are relatively easy to implement and generally have high convergence speeds to global optima. They are best suited to continuous variables, unlike GAs, which work well on discrete variable spaces. PSOs can, however, struggle with constrained problems.

Among the possible alternatives to GAs and PSOs are ant colony optimisation (ACO) algorithms, which are inspired by the foraging behaviour of certain ant species. Ant colonies are known for their collaboration and ability to accomplish tasks too difficult for individual ants. In particular, the foraging behaviour of ants reflects their ability to find the shortest path between food sources and their nests. In certain cases, a special hormone called pheromone is left along the trails and this acts as stimuli for other ants to follow the same trail. The more pheromone along a trail, the more attracted ants

are, and subsequently the more pheromone is deposited on the trail. ACOs exploit this idea to solve problems whose solutions can be formulated as a least cost path between an origin and a destination. This cost-to-go idea can be seen in many optimisation problems, including optimal control theory, dynamic programming and RL [75].

Tree-search algorithms are another promising but less studied alternative to evolutionary algorithms [30]. If the problem can be divided into a sequential set of sub-problems, then it can be described by a tree-search algorithm. This is often the case for the combinatorial challenges such as complex rendezvous and fly-by problems. In these instances, the problem can be thought of as a bi-level optimisation problem, with an inner continuous layer (e.g. trajectory optimisation) encapsulated by a discrete decision points (e.g. the order of individual trajectories). The performance of the optimised trajectories at the inner-level guide the selection at the outer level, whilst the out level changes the fitness landscape of the inner level. Decision points are modelled as nodes and tree-search algorithms use a strategy to only follow the most promising branches in order to prevent the need for an exhaustive enumeration of all possible node expansions. One such strategy can be found in beam-search algorithms, which balances the exploration and exploitation of the search tree through a ranking criteria which limits tree expansion to a subset of nodes.

This is particularly powerful for combinatorial problems such as the travelling salesman problem, where PSO begins to struggle. Although direct application of evolutionary algorithms to combinatorial problems is possible, it may lead to sub-optimal results if the search space becomes too large to be sampled effectively [30]. Tree search algorithms can be made stochastic, for example using Monte Carlo (MC) tree search algorithms, which are frequently used in complex game environments [30, 76].

ACO combined with beam-search algorithms has been used for multi-rendezvous spacecraft trajectory optimisation problems [77]. According to [30], a similar approach was used to win the 9^{th} Global Trajectory Optimisation Competition. Both ACO and MC tree search algorithms are closely related to RL [75, 76, 78]- see Section 2.4. The advantages are how they handle the exploration-exploitation dilemma. RL methods require the states to follow a Markov decision process (MDP), whereas ACO and MC tree search do not. However, this enables RL methods to apply updates to their solutions before observing the full solution (a process known as boot-strapping), whilst ACO and MC algorithms must wait until they construct a full solution before updating. This can make RL more universally-applicable across wider range of problems [75].

2.4 Reinforcement Learning

RL is one of the three subsets of ML. First, there are algorithms which are given a set of training examples from which to learn the relations, before applying them to unseen examples - *supervised learning*. *Unsupervised learning* is where the algorithm is free to find

structure in an unlabelled training data set. The third is *RL*, where an agent explores an environment and receives feedback in the form of rewards and improves its actions to maximise the associated reward.

2.4.1 General Concepts

Critical to the understanding of RL are the agent, policy and reward. Given the state of the environment, the policy will determine what action the agent takes. The agent explores the environment and rewards provide feedback for a given action during the training process [46]. Hence, the formulation and structure of the reward (or cost) function is critical. During training, an agent will take actions based on an untrained policy, and over time this policy will improve until the optimal policy is learnt. This maps states to actions to maximise the total reward (or minimise the total cost) for the environment.

The RL training methods have two branches: model-free algorithms and model-based algorithms. Model-based algorithms use in-built knowledge of the dynamical model during the learning process. They are able to predict the behaviour of the agent when selecting an action and hence can plan their actions in advance. In the model-free algorithms, the agent is assumed to have no knowledge of the dynamical model. Hence, it can only be trained via iterations and experience in the environment. Model-free algorithms are advantageous as they can be applied to situations where the dynamics model is unknown. A few well known model-free RL algorithms include REINFORCE [79], proximal policy optimisation (PPO) [80], deep deterministic policy gradient (DDPG) [81], advantage [82] and asynchronous actor-critic (AC) approaches [83].

There are three primary ways of training the model-free RL agent algorithms: value-based, policy-based and actor-critic based methods. Value-based learning that takes a value function to estimate how valuable the given state observation is in terms of the future rewards. A value function is estimated and updated to minimise the error where the target value can be calculated from the optimal Bellman equation. Next, the policy could select an action in a way to maximise the estimated value function of the next state. Policy-based learning can directly update the policy towards the optimal policy by sampling the gradient of the objective function. Finally, AC-based methods are a combination of these two approaches.

The training methods can also be categorised depending on how the data is sampled: on-policy and off-policy algorithms. On-policy algorithms use the data that is sampled only by the most recent policy, while off-policy algorithms exploit the data whenever it is sampled by any policy.

NNs are commonly used in conjunction with RL techniques [46]. They are composed of activation functions (nodes) connected in layers which allow them to model complex functions and operations through matrix multiplications and nonlinear functions applied element-wise. This makes them powerful function approximators. They can have

single or multiple hidden layers and will be discussed in Section 2.4.2.4.

2.4.2 Value and Advantage Functions

More formally, RL problems are usually posed as a MDP, with a sequence of states x_i, actions a_i and a transition dynamics distribution with conditional density $p(x_{i+1}|x_i, a_i)$. This represents the dynamical relationship between states x_i and x_{i+1}, given an action a_i. This MDP repeats sequentially and requires no knowledge or history of the states previously visited to determine the next state. It is pictorially represented as [84]:

$$x_0 \xrightarrow{a_0} x_1 \xrightarrow{a_1} x_2 \xrightarrow{a_2} x_3 \xrightarrow{a_3} .... \tag{2.62}$$

The agent (in this case the spacecraft) interacts with the environment using a parameterised policy $\pi_\theta(a|x)$ that defines the action taken $a \sim \pi_\theta(a|x)$ (where the symbol $\sim$ stands for 'belonging to'). Notice how the chosen action is a function of the state. As it interacts with the MDP it collects observations x_i, receives costs $c(x_i, a_i)$ based on the actions taken, and will transition to a new state $x_{i+1} \sim p(x_{i+1}|x_i, a_i)$. For the sequence of states, the discounted cost-to-go C_i is often used, which is a discounted sum of the cost $c_i = c(x_i, a_i)$ at each remaining state along the sequence with discount factor $\gamma \in (0, 1]$, and is written as:

$$C_i = c_i + \gamma c_{i+1} + \gamma^2 c_{i+2} + \gamma^3 c_{i+3} + ... = \sum_{j=i}^{\text{end}} \gamma^{j-i} c_j. \tag{2.63}$$

The agent's goal is to obtain a policy that minimises this discounted cumulative cost (or if you prefer, maximises the discounted cumulative reward) from the start state to the end state. This is denoted by the performance objective $J(\pi_\theta)$ and can be written as an expectation

$$J(\pi_\theta) = \mathbb{E}_{a\sim\pi_\theta}[C_0(x, a)] \tag{2.64}$$

where $\mathbb{E}[\cdot]$ is the expectation operator calculated when applying the policy π_θ. More information can be found in Sutton and Barto [46], Schulman *et al.* [80], Scorsoglio *et al.* [85] and Miller *et al.* [86]. This performance objective can be thought of as the expected cost-to-go at the initial state, C_0, under the policy π_θ. As a result of Bellman's equations [46], this objective can be minimised by finding the optimal policy

$$\pi_\theta^\star = \arg\min_\pi J(\pi_\theta). \tag{2.65}$$

During training, the agent needs to estimate J for a given policy π_θ, a procedure known as policy evaluation. The estimate of J is known as the value function. There are different

ways of defining the value function. The state value function $V^{\pi}(x)$ is defined as:

$$V^{\pi}(x) = \mathbb{E}\left[\sum_{j=0}^{\text{end}} \gamma^{j} c_{j+1} | x_0 = x, \pi\right], \tag{2.66}$$

which depends only on the state and assumes the policy π is followed starting from this state. The notation indicates this is the expected cost-to-go from state x to the end given the policy π is used. Recall $c_i = c(x_i, a_i)$ is function of both the state and action taken. The state-action value function $Q^{\pi}(x, a)$ instead is defined as:

$$Q^{\pi}(x, a) = \mathbb{E}\left[\sum_{j=0}^{\text{end}} \gamma^{j} c_{j+1} | x_0 = x, a_0 = a, \pi\right], \tag{2.67}$$

and depends both on state and action. However, instead of assuming the action a is generated by the policy π, the action chosen is a free variable [82]. Once the transition to the next state has been made, the policy π determines the subsequent actions a. The two value functions can be used to define a quantity called the advantage function:

$$A^{\pi}(x, a) = Q^{\pi}(x, a) - V^{\pi}(x), \tag{2.68}$$

which uses the state value function as a state-dependent baseline to estimate how much better an action is with respect to the expected optimal action under the current policy and is used in general to increase the learning performance.

2.4.2.1 Policy Gradient

For a continuous state and action space, the most popular RL algorithms are policy gradient algorithms [81, 87]. These use the value function to adjust parameters θ of a policy π_{θ} in the opposite direction of the gradient of the objective function:

$$\nabla_{\theta} J(\pi_{\theta}) = \mathbb{E}[\nabla_{\theta} \log \pi_{\theta}(a|x)\, A^{\pi}(x, a)]. \tag{2.69}$$

Hence, the update is given by:

$$\theta_{k+1} = \theta_k - \alpha_k \nabla_{\theta} J_k(\pi_{\theta}). \tag{2.70}$$

AC algorithms are based on these policy gradient algorithms where there are two main components. A critic evaluates the quality of a given action by approximating the value function, often using a NN. The actor learns the policy and maps the state to the action the agent should take, often also using a NN. For more information, Grondman *et al.* [82] provide a good survey of AC approaches.

As Fig. 2.5 shows, at each stage in the process, actions are sampled from the actor using the current policy and result in a trajectory. A cost is assigned to each state and ac-

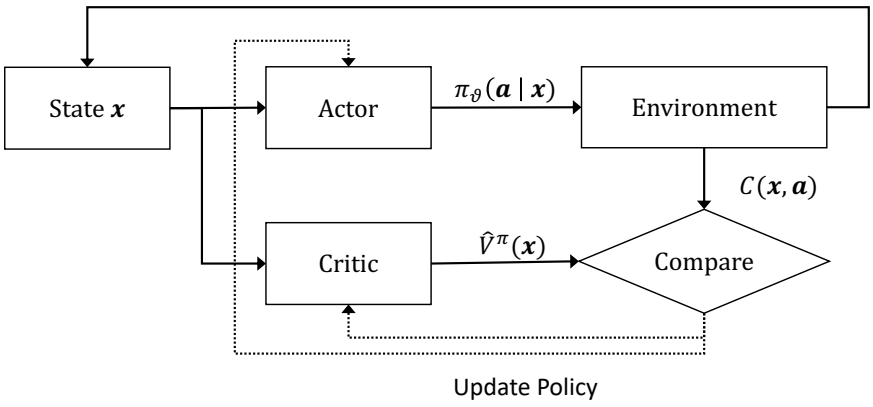


Figure 2.5: Basic actor-critic architecture

tion pair. These costs provide an insight into the value of that state-action pair and allow the algorithm to learn and improve from its previous decisions. The critic is responsible for learning this value function $V^\pi(x)$.

The continuous nature of both the state and action spaces mean these methods are only really successful if NNs are employed for both the actor and critic parts of the algorithms. If RL is to be used successfully in this work, AC approaches are the most appropriate.

2.4.2.2 Update

For a batch of N episodes each with T time-steps, the update in Eq. (2.69) can be approximated as:

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{T} \nabla_\theta \log \pi_\theta(a_{n,i}|x_{n,i}) \hat{A}^\pi(x_{n,i}, a_{n,i}), \tag{2.71}$$

which uses the approximate advantage function

$$\begin{aligned} \hat{A}^\pi(x_{n,i}, a_{n,i}) &= \hat{Q}^\pi(x_{n,i}, a_{n,i}) - \hat{V}^\pi(x_{n,i}) \\ &= c(x_{n,i}, a_{n,i}) + \gamma \hat{V}^\pi(x_{n,i+1}) - \hat{V}^\pi(x_{n,i}). \end{aligned} \tag{2.72}$$

Here $\hat{V}^\pi(x_i)$ indicates the expected cost-to-go of the current state - its value - and $\hat{V}^\pi(x_{i+1})$ the expected cost-to-go of the subsequent state. Both are approximate given they are learnt by the critic network. The difference between the two is the expected cost of the current state x_i. Hence, the advantage function reflects how much better the action a_i taken in state x_i is with respect to the expected action, which can be determined by having multiple trajectories within the same batch during the learning process.

2.4.2.3 Proximal Policy Optimisation

An RL update strategy based on PPO [80] can be used to increase the algorithm's overall performance both in terms of optimality and robustness. This is an AC on-policy algorithm which clips the objective function to remove incentives for the new policy to get too far away from the old policy. In other words it ensures the update size is within a trusted region, attempting to prevent accidentally bad updates. PPO is a simpler and more flexible alternative to trust-region policy optimisation (TRPO) and can achieve the same high performance [88]. The objective $J(\pi_\theta)$ is written in terms of a probability ratio $R(\theta)$ between the current and new policies:

$$R(\theta) = \frac{\pi_\theta(a|x)}{\pi_{\theta_k}(a|x)}. \tag{2.73}$$

π_θ represents the latest policy which is assessed using the current batch. π_{θ_k} is the active policy at learning iteration k and is used to generate the batch of trajectories. Using the clipped-PPO approach, the gradient of the clipped objective is given by two slightly different equations, depending on whether the advantage function is positive or negative. For $\hat{A}^\pi \geq 0$,

$$\begin{aligned}\nabla_\theta J(\theta)^{\text{CLIP}} = \mathbb{E}[\nabla_\theta \log \pi_\theta(a|x) \min\Big(&R(\theta)\hat{A}^\pi(x,a),\\ &\text{clip}(R(\theta), 1+\epsilon)\hat{A}^\pi(x,a)\Big)],\end{aligned} \tag{2.74}$$

and for $A^\pi < 0$:

$$\begin{aligned}\nabla_\theta J(\theta)^{\text{CLIP}} = \mathbb{E}[\nabla_\theta \log \pi_\theta(a|x) \max\Big(&R(\theta)\hat{A}^\pi(x,a),\\ &\text{clip}(R(\theta), 1-\epsilon)\hat{A}^\pi(x,a)\Big)],\end{aligned} \tag{2.75}$$

where ϵ is a hyperparameter which determines how far the new policy can deviate from the old.

2.4.2.4 Neural Networks

NNs are composed of layers of nodes inspired by the structure inside the brain. Figure 2.6 shows an example NN with two hidden layers. The input layer, the first layer in the diagram, is usually a set of normalised states associated with the environment. For instance, in spacecraft trajectory design these could be the position and velocity vectors in a given frame, or the spacecraft mass. Next there are a sequence of hidden layers, each with a set of nodes and nonlinear activation functions such as rectified linear unit (ReLU), hyperbolic tangent, or sigmoid. In the simplest form, these are connected to the input layer using a set of weights (denoted as ϑ_1) and biases. Depending on the specific architecture there can be multiple hidden layers, each with a different set of weights and biases, and even different activation functions and different number of nodes. The final

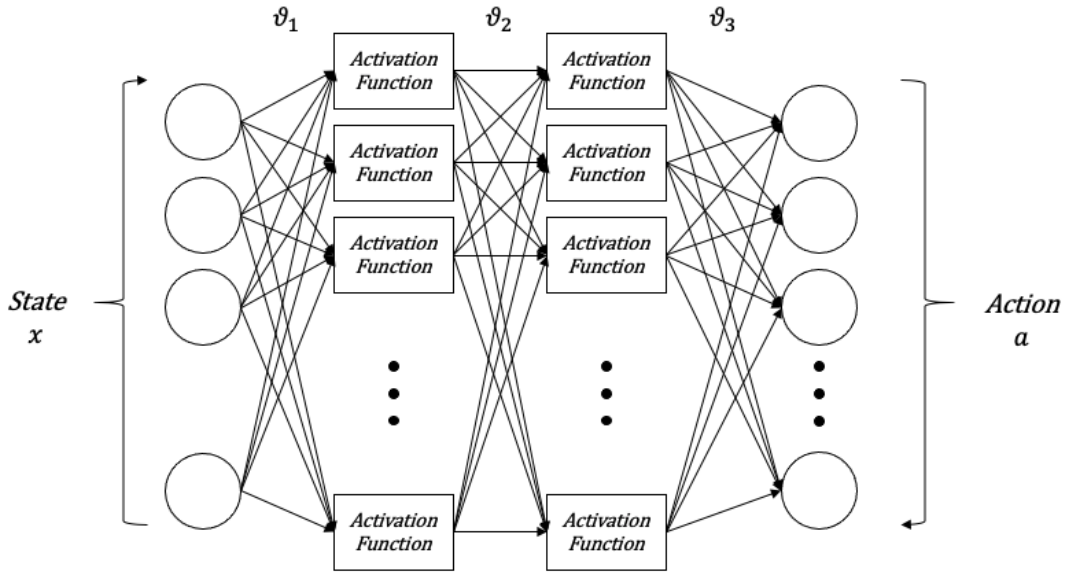


Figure 2.6: Example 2-hidden layer neural network

layer is denoted the output layer. This is the action a given by the policy π_θ and state x, and will reflect the current best estimate of the optimal behaviour in the environment. NN architecture is a non-trivial process and a hugely complex field [46, 89].

Training NNs to replicate desired behaviour refers to learning the optimal weights for each connection within a NN. This often requires an iterative process due to the complex and nonlinear relationships within a network. Additionally, the selected activation function and number of hidden nodes in each layer often influences the convergence behaviour of the algorithm. Complex solution spaces, for example multi-body dynamics, may best be represented with a large number of hidden nodes and layers. However, this can come at the expense of longer training and difficult convergence. Furthermore, the choice of activation function for a specific problem is nontrivial and still an active area of research [89].

2.4.2.5 Extreme Learning Machines

Conventional NNs use back-propagation to iteratively learn their weights. This is a multiple step process, and is the brain behind the learning process. Extreme learning machines (ELMs) [90] are single-layer feed-forward networks (SLFNs) that remove the need for back-propagation, significantly increasing the learning process speed.

Any continuous target function $f(x)$ can be approximated by a SLFN with a set of random hidden nodes, activation functions $h(x)$ and learnt parameters β_{ELM} [90]:

$$f_L(\boldsymbol{x}) = \sum_{i=1}^{L} \beta_{ELM\,i} h_i(\boldsymbol{x}) = \mathbf{H}(x)\beta_{ELM}. \tag{2.76}$$

The output parameters are β_{ELM} and the input weights and biases $\mathbf{w}$ and $\mathbf{b}$ are used in the activation function $\mathbf{H}(x) = \sigma(\mathbf{w}x + \mathbf{b})$. Conventionally each of $\mathbf{w}$, $\mathbf{b}$ and β_{ELM} are learnt via back-propagation. However, in an ELM, only β_{ELM} is learnt, whilst the weights and biases can be randomly assigned and kept constant for the entire process. This is known as the universal approximation theorem and works if the number of features L is sufficiently large.

Given N training samples $\mathbf{Y}$, and a known $\mathbf{H}$, the output parameters β_{ELM} can be learnt:

$$\beta_{ELM} = \tilde{\mathbf{H}}\mathbf{Y}. \tag{2.77}$$

where $\tilde{\mathbf{H}}$ is the Moore-Penrose generalised inverse matrix and is required because $\mathbf{H}$ is not a square matrix. This approach allows for quick learning and has been demonstrated to be effective in control law weight tuning for rendezvous problems [85].

2.4.3 Reinforcement Learning in Astrodynamics

The use of deep learning and RL in decision-making systems has produced increasingly exciting results over the past decade and in diverse applications ranging from robotics, to self-driving cars and unmanned air vehicles, and spacecraft [29, 46, 91]. The major draw for RL algorithms is their performance in unfamiliar environments and their similarities with the mathematics of both optimal control theory and dynamic programming [46]. Both in optimal control theory and RL, Bellman's principle of optimality can be applied to solve for the optimal control/policy. Usually the difference is RL approximates the objective in a sub-optimal fashion using a NN, and searches for the solution by stochastic exploration. The ideas of value iteration and policy iteration in dynamic programming inspire the value and state-value functions in RL. However, unlike in optimal control theory, where the dynamics are known through the physics of the problem, in RL the dynamics are given by the MDP and the probability of transition from one state to the other through the policy.

Modern RL techniques can train NN controllers without direct knowledge of the dynamical environment - see the very extensive survey by Shirobokov *et al.* [29]. This ability to learn uncertain and stochastic environments where the dynamics are unknown makes these techniques very attractive. There is significant interest in applying RL in astrodynamics, particularly for spacecraft guidance [30].

Gaudet *et al.* [92, 93] have demonstrated the capability of RL for use in asteroid hovering missions and Mars landers. They extended this work to the Meta-learning for asteroid hovering [94], demonstrating impressive stability to uncertainties in the grav-

itational field. Scorsoglio and Furfaro have also investigated autonomous lunar landing using image-based Deep-RL [95]. The same authors have used it to make the parameters of a zero-effort-miss/zero-effort-velocity (ZEM/ZEV) feedback algorithm state-dependent [85]. This was done in the CR3BP for near-rectilinear halo orbits (NRHOs) and improved the capabilities of the ZEM/ZEV approach. Radial basis functions (RBFs) and ELMs were used to make the learning process lightweight, and the guidance problem was able to handle a variety of constraints. Hovell *et al.* [96] have used Deep-RL for pose tracking and docking scenario in spacecraft proximity operations, however they also combine it with a guidance, navigation and control (GNC) approach. Conventional controllers are able to handle dynamic uncertainties and modelling errors that typically plague RL policies that attempt to learn the entire GNC routine. In the GNC world it is becoming standard to combine both RL and conventional guidance approaches

Miller and Linares [97] provided insights into using RL in the low-thrust CR3BP. Lots of research focuses on developing closed-loop guidance approaches around stable orbits in the CR3BP, with LaFarge *et al.* [98, 99] investigating both guidance between Libration point orbits and station-keeping on NRHOs. Sullivan and Bosanac [100] investigate low-thrust transfer design between libration point orbits using a multi-reward PPO algorithm. Bosanac *et al.* [101] use PPO to design reconfiguration manoeuvres between two spacecraft along an L_2 halo orbit, whilst Bonasera *et al.* [102] use PPO to design the station-keeping manoeuvres for the same problem.

Yanagida *et al.* [103] explore long time-of-flight transfers in the CR3BP using a DDPG and ΔV impulses. Interplanetary transfer design has also been considered by Miller *et al.* [104] and Zavoli and Federici [105]. Both are using PPO and demonstrate robustness to errors. Kwon *et al.* [106] are using RL for autonomous guidance for multi-revolution low-thrust orbit transfers. This is very exciting work where the RL is not combined with any controller, although the optimality and stability are lagging behind conventional techniques considerably at the moment.

Although not along the same lines as the research in this thesis, there is lots of research using NNs for discovering the governing equations for dynamical systems from data using ML [91]. In addition, NNs are being used to approximate optimal control solutions [107] and the solution of Hamilton-Jacobi-Bellman equations using physics-informed learning [108]. In the field of space mission design, (arguably as equally challenging as trajectory design), Harris *et al.* [11] use the RL framework to address high-level mission planning and decision making problems.

Clearly there is significant and growing interest to use RL with astrodynamics. Perhaps one of the reasons is RL has similarities with both optimal control theory and certain global optimisers such as ACO [78]. The extensive survey by Shirobokov *et al.* [29] highlights two key areas for future development with RL in astrodynamics: the lack of stability guarantees within many RL approaches thus far and investigating the performance of NN techniques in unmodelled environments.

2.5 Discussion and Summary

Chapters 1 and 2 have discussed current literature in the field of trajectory design and RL. Conventional direct and indirect methods for computing low-thrust transfers still present many challenges. They remain computationally expensive as they require solving complex optimisation problems. The emergence of heuristic control methods, particularly those involving CLFD control laws, has allowed the computation of sub-optimal trajectories with minimal computational cost [35, 39, 57]. These can still provide results suitable for initial mission planning, initial guesses for more complete methods [37, 38], and potentially for on-board use. Lyapunov control laws are particularly attractive due to their stability properties. Combining these with techniques to vary their user-defined parameters would increase their performance. Evolutionary algorithms such as GAs and PSOs have effectively demonstrated this [60, 74].

When perturbing accelerations are included, the most exciting work comes from Pontani *et al.* [109], where real consideration of the impact of the perturbing accelerations are made, and their impact on Lyapunov stability. This is applied to station-keeping problems with promising results. However, they make no attempt to utilise the perturbation acceleration to their advantage.

Table 2.1: Overview of where the proposed approach fits in with the Literature

Author	Control Law	Optimisation	Stability	Perturbations	Comment
Ruggiero *et al.* [39]	DAG	~	×	×	Thrust Blending
Fummenti *et al.* [59]	Quasi-Impulsive	~	×	×	
Petropoulos *et al.* [36]	Q-law	Constant [60, 62]	Lyapunov	×	Estimated time-to-go
Maddock *et al.* [65]	Q-law	~	Lyapunov	✓	cancel $\boldsymbol{a}_p$
Epenoy *et al.* [67]	Lyapunov	~	Lyapunov	✓	cancel $\boldsymbol{a}_p$
Pontani *et al.* [109]	Lyapunov	~	Lyapunov	✓	Stability Analysis with $\boldsymbol{a}_p$
Locoche *et al.* [63]	ΔV-law	Constant	Lyapunov	×	Estimated ΔV-to-go
Peterson *et al.* [69]	Lyapunov	~	Lyapunov	×	Waypoints
Gao *et al.* [37]	Q-law	Time-varying	Lyapunov (t)	×	Initial co-states guess
Shannon *et al.* [38]	Q-law	Constant	Lyapunov	×	Initial co-states guess
Yang *et al.* [74]	Q-law	State-dependent	Lyapunov (X)	×	ANN + GA
Ohndorf *et al.* [28]	Neurocontroller	Evolutionary Neurocontrol	×	×	ANN interplanetary
Kwon *et al.* [106]	Neurocontroller	State-dependent	×	×	ANN direct state-to-control
Shirazi *et al.* [110]	Q-law	Time-varying	Lyapunov (t)	×	
Holt *et al.* [42]	Q-law	State-dependent	Lyapunov	Free to exploit	

Table 2.1 provides a comparison of techniques available in the literature, presenting their strengths and limitations in three categories: optimisation, stability and perturbations. Optimisation refers to the ability for the usually sub-optimal heuristic controller to be optimised and if, and how, it has been done in the literature. All the techniques have the potential to be optimised, but the best results involve either time- or state-varying parameters. Stability refers to the possibility of making claims on the stability of these control laws. Most are Lyapunov functions, which provides a wealth of stability information. However, in the few cases where they are time-varying or state-dependent the stability considerations have been neglected. Finally, the question remains on how these approaches perform in the presence of perturbing accelerations. Most authors are able to show robustness to perturbing dynamics, or incorporate them within analytical expressions for the controller. However, they are unable to exploit the existence of the

perturbations.

To conclude, methods which utilise state-dependent parameters are yet to ensure Lyapunov stability and yet to fully exploit perturbations to the advantage of the control law. To the authors knowledge, no work has combined the powerful learning process and state-dependent actions from RL with the advantages of Lyapunov control laws for trajectory design purposes. In addition, the burden evolving perturbations and constraints such as eclipse effects can put on low-thrust systems means time-varying weights are identified as an area that would improve the current body of work. Whilst there is significant interest in the application of RL in astrodynamics [29, 30], one of the contributing factors around their lack of deployment in space environments thus far is the difficulty in ensuring stability requirements are met [29]. Concerns over neural network stability in case of neuron failure or simply the stability properties of the control output remain. By combining Lyapunov control laws with NNs, the impact of failures can be reduced as the underlying Lyapunov control law can be used as a inbuilt back-up.

Chapter 3

Reinforced Lyapunov Controller

In this chapter the proposed methodology for this PhD research is given. The main purpose of this chapter is to clarify the motivation behind the approach and to outline the theoretical background needed to understand the implementations found in later chapters. The main contributions can be divided into three parts: the state-dependent philosophy, the RL framework, and ensuring Lyapunov stability. This is then tested for trajectory design in Keplerian dynamics. The main purpose is to understand how the Reinforced Lyapunov Controller performs in an environment it has full knowledge of. This work was published in *Acta Astronautica* [42], with H. Holt as the author. The original and novel state-dependent philosophy was envisioned by H. Holt and R. Armellin, and the subsequent RL framework was developed by H. Holt, N. Baresi and R. Armellin, with A. Scorsoglio and R. Furfaro providing expertise on the fundamentals of RL. Lyapunov stability considerations were addressed by H. Holt, N. Baresi and R. Armellin and encompass the major novelty of the work. A. Turconi and Y. Hashida were involved in monthly supervision meetings as the industrial sponsors of the PhD. This chapter allows the most direct comparison between the novel approach and the literature, and paves the way for more in-depth analyses in Chapters 4, 5, 6 and 7.

3.1 Introduction

The goal is to develop a lightweight and closed-loop control law that can be used for both initial trajectory design and on-board guidance. The proposed approach combines Lyapunov control theory with RL techniques. The major draw for RL algorithms is their performance in unfamiliar and stochastic environments, and their similarities to optimal control theory. As discussed in Chapter 2, there is significant interest in the application of RL in astrodynamics, from mission design, operations, guidance and control, to navigation and even the prediction of the dynamics. The issue around RL techniques, and one of the contributing factors around their lack of deployment in space environments thus far, is the difficulty in ensuring stability requirements are met [29]. This is beginning to be addressed for simple benchmark control applications [111, 112].

The work in this chapter presents a stable Lyapunov controller combined with a RL architecture. The advantage of combining these two approaches allows one to eliminate the drawbacks from each approach: namely the sub-optimality of Lyapunov controllers and the unknown stability of RL methods. First a proof-of-concept is provided,

demonstrating how the optimality of Lyapunov controllers can be improved by optimising the user-defined parameters and then making them time-varying. The next section discusses how to extend this from a time-varying approach to a state-dependent one, which will discuss the importance of NNs and the fundamental RL architecture that is used to learn the mapping from state to action. A state-dependent controller is presented which enforces stability without compromising optimality through the Jacobian of the state-dependent parameters. The results for transfers from GTO-GEO and LEO-GEO are presented for both time and mass-optimal transfers in Keplerian dynamics. These transfers are selected to offer comparison with previous work in the literature.

Note that this approach will be introduced using the Lyapunov-based Q-law, as discussed in Chapter 2. However, it extends to other Lyapunov functions. Performance for both time and mass-optimal transfers are discussed and the influence of the RL architecture on both optimality and stability is presented for the first time. To the author's knowledge the addition of stability through the Jacobian term has not been explored elsewhere. The importance of this step can extend beyond trajectory design and more generally to optimisation of non-linear controllers.

Although out-of-scope for this work, this approach also offers substantial potential for on-board use. Currently a classical Lyapunov controller could be implemented on-board as a guidance closed-loop control law. What is proposed is to train a NN on-ground and upload the trained network alongside the Lyapunov controller. This retains a closed-loop controller with much improved optimality, saving on either time-of-flight or propellant mass. The computationally expensive learning process is thus separated from the resulting on-board controller thanks to the trained NN. Once trained, the closed-loop nature of the NN offers the ability to quickly and autonomously evaluate a control history for the spacecraft with basic linear algebra operations.

3.2 Proof of Concept: time-varying parameters

As discussed in Chapter 2, Lyapunov control theory offers a way of developing control laws that provide a guaranteed stability in non-linear dynamical environments. For the purposes of this chapter, it is important to realise that control laws derived in this fashion are functions of the current state $\boldsymbol{X}$, the target state $\boldsymbol{X_T}$, a set of weights $\boldsymbol{W}$, and a further set of user-defined parameters, p_L:

$$V = V(\boldsymbol{X}, \boldsymbol{X_T}, \boldsymbol{W}, p_L). \tag{3.1}$$

If V is a Lyapunov function then any control $\boldsymbol{u}$ which can ensure $\dot{V} < 0$ is considered Lyapunov stable, and conventionally $\boldsymbol{u}$ is chosen such that $\boldsymbol{u} = \text{argmin}(\dot{V})$. Unless otherwise stated, the results presented use the Lyapunov-based Petropoulos Q-law. In the case of the Q-law, the parameters p_L can affect the performance and are sometimes optimised

in the literature. However, $\boldsymbol{W}$ have a direct relationship with the current state and characterise the controller's intrinsic behaviour. The p_L parameters are secondary, and adjust terms such as the altitude penalty function, a trait which is desirable in almost all cases. As such, a choice is made to use the standard values for p_L and only consider optimising $\boldsymbol{W}$, unlike in Varga *et al.* [113] and Lee *et al.* [62].

In order to replicate the potential performance of state-dependent parameters, one can simulate time-dependent parameters using cubic spline interpolations. As an example test case, consider low-thrust time-optimal and mass-optimal transfers from a GTO-GEO and LEO-GEO in Keplerian dynamics. The goal is to compare three distinct benchmark cases to highlight the strength of allowing time-varying parameters. The first is the "classical Q-law" where $\boldsymbol{W} = \mathbf{1}$ are selected arbitrarily, which is the case first presented in the literature. However, as many authors have found before, tuning these weights can significantly improve the performance [60, 62, 74].

Two further benchmarks are considered, both utilising a PSO to tune the weights. First, a PSO is used to select more optimal $\boldsymbol{W}$ that remain fixed throughout the transfer (hereafter known as "PSO fixed"). This is a standard approached used for CLFD control laws. When targetting a, e, and i the optimisation vector is:

$$\vec{x} = [W_a \quad W_e \quad W_i \quad \eta_a^t], \tag{3.2}$$

which includes the normalised state weights $W_X = [W_a \quad W_e \quad W_i]$ and the absolute effectivity threshold η_a^t. For the test cases presented in the next section, at least three simulations were run with a population of 200 particles, with the best result selected. This exceeds to rule of thumb of 10 times the number of optimisation variables. Max stall iterations was set to 25.

An alternative is to simulate time-dependent parameters using cubic spline interpolations. Here this is referred to as the "PSO spline" approach, and is used to make the weights time-dependent. For simplicity, the transfer is divided into 3 time-intervals where coefficients are determined by the PSO. This leads to an optimisation vector of:

$$\vec{x} = [c_{a1} \quad c_{e1} \quad c_{i1} \quad c_{\eta_a^t 1} \quad c_{a2} \quad c_{e2} \quad c_{i2} \quad c_{\eta_a^t 2} \quad c_{a3} \quad c_{e3} \quad c_{i3} \quad c_{\eta_a^t 3}]. \tag{3.3}$$

Using these coefficients, cubic splines are used to interpolate the value of $\boldsymbol{W}$ at any time, thereby parametrising the weights as piece-wise cubic polynomials over a grid of time-intervals. Simulations were run with a population of 500 particles. Again, the best from at least three simulations was selected.

3.2.1 Geostationary Transfer Orbit to Geostationary Orbit

Here a GTO-GEO transfer with an inclination change is considered in Keplerian dynamics. The parameters are chosen to compare with Yang *et al.* [74] and Geffroy *et al.* [114]

and were considered also considered in [40], [41] and [42]. Yang *et al.* [74] used a NN and ICEA optimiser to make the design parameters of a Lyapunov control law state-dependent. As such, their approach provides a great benchmark for testing the optimality of our RL enhanced Lyapunov control approach due to the overlap in methodologies. Geffroy *et al.* [114] used a generalised averaging method and an indirect formulation to compute minimum-time and fuel-saving transfers, and was also the benchmark considered in Yang *et al.* [74]. Whilst neither can claim to be the true optimal control solution, performance on par with Yang *et al.* [74] would present a significant improvement with respect to classical Lyapunov control laws.

The modelled spacecraft has a mass of 2000 kg, a thrust of 0.35 N and an I_{sp} of 2000 s, giving an initial thrust-to-mass ratio of 0.000175 m s^{-2}. The initial and target orbits are shown in Table 3.1. The convergence criteria is placed on the Q-law residual rather than individual elements. This is also done by Varga *et al.* [113] and reduces the likelihood of control chatter. A 0.25 day residual is chosen as it provides comparable results with previous criteria and because it is the decision time-step in the RL algorithm - see Section 3.3.2. These convergence criteria prevent the Q-law from chattering close to the target orbit.

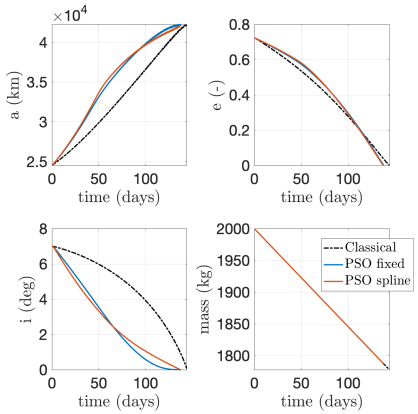
Table 3.1: Initial and target orbital elements for a GTO-GEO transfer.

	a (km)	e	i (°)	Ω (°)	ω (°)	ν (°)
Initial	24,505.9	0.725	7	0.0	0.0	0.0
Target	42,165.0	1e-5	0	*free*	*free*	*free*

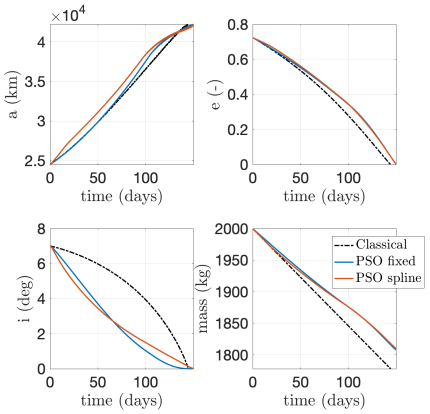
Table 3.2: Comparison of the time-dependent Q-law performance for GTO-GEO transfer.

	Method	Time (days)	Propellant (kg)
	Time-optimal Classical	144.03	222.06
Time	Ref. [74]	137.3	211.72
	Ref. [114]	137.5	212.00
	PSO fixed	137.90	212.61
	PSO spline	137.16	211.47
Mass	Ref. [74]	150.00	187.97
	Ref. [114]	150.00	192.00
	PSO fixed	149.75	192.14
	PSO spline	149.75	190.03

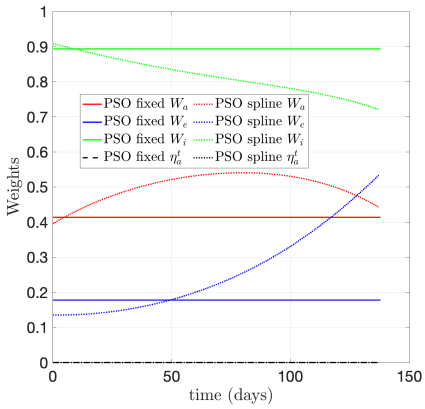
Results for both time and mass-optimal solutions are given in Table 3.2. It is clear from the PSO fixed simulations that the optimality in terms of time-of-flight can be improved compared to the time-optimal classical Q-law by tuning the weights, as many have found before [60, 62, 74]. The PSO spline makes the weights time-dependent and for time-optimal transfers should provide an indication of the best possible solution available to the RL approaches. For the mass-optimal simulations, the absolute effec-



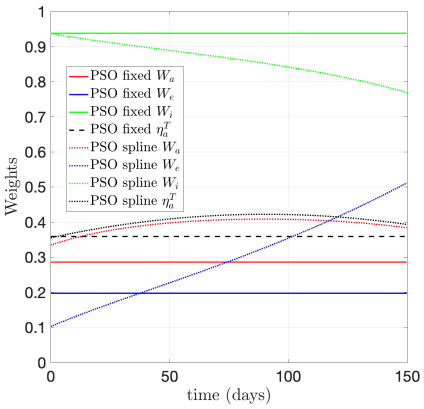
(a) Time-optimal Orbital Parameters.



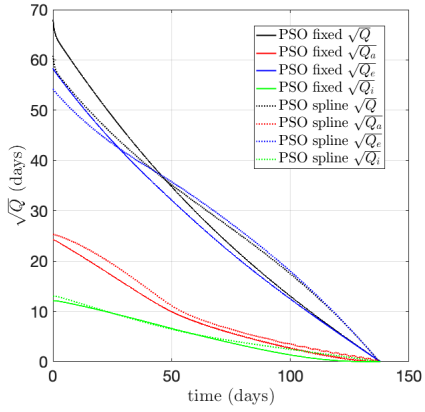
(b) Mass-optimal Orbital Parameters.



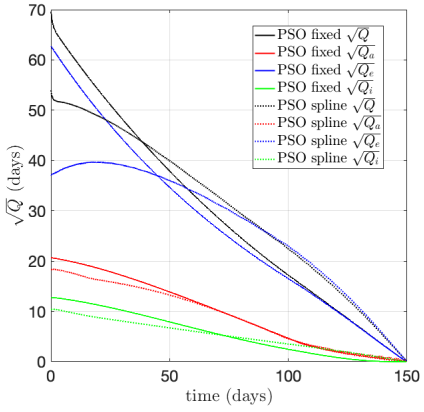
(c) Time-optimal weights.



(d) Mass-optimal weights.



(e) Time-optimal Q-law contributions.



(f) Mass-optimal Q-law contributions.

Figure 3.1: PSO Q-law results for GTO-GEO transfer. Both time-optimal and mass-optimal results are shown for both the PSO fixed and PSO spline approaches.

tivity threshold η_a^t was used and a target time-of-flight of 150 days was set.

Compared to the time-optimal classical Q-law, the best solution can save 32.03 kg (14.0%) at a cost of 5.72 days (3.9%) - as seen in Table 3.2. Compared to the classical Q-law ($\boldsymbol{W} = \mathbf{1}$), Fig. 3.1c shows the PSO fixed simulation prioritises the inclination term which matches the intuitive understanding as this is the most expensive orbit element (out of a, e and i) to change. This is also clearly visible in Fig. 3.1a, where the inclination decreases more rapidly at the start of the transfer for both PSO solutions compared to the classical Q-law.. However, this does not reveal the full picture. The Q-law formulation combines W_X and the ratio of $\delta X = X - X_T$ to $\max_\nu(\dot{X})$, which represents the squared time-to-go to change that orbital element. From Fig. 3.1e it is clear the eccentricity term accounts for the the majority of the control, despite being weighted the least for the transfer. Here $\sqrt{Q_X}$ indicates the contribution of element X to the Q-law Lyapunov function. At each instance in time, $Q = \sum_X (\sqrt{Q_X})^2$. The square-root is taken to adjust the units to time-to-go. Thus, it indirectly indicates the contribution element X is having to the control computation. The greater the ratio of Q_X to Q, the greater the contribution of that orbital element. As such, Figs. 3.1e and 3.1f reveal the most information of the combination of W_X, $\delta X = X - X_T$ and the orbital dynamics at play. These results suggests the classical Q-law over-prioritises the eccentricity and hence the PSO fixed reduces the contribution of this term. Again, this can be seen around 50 days into the transfer in Fig. 3.1a. The PSO spline starts with very similar values to the PSO fixed, but then prioritises the eccentricity further towards the end.

In the mass-optimal transfer the effectivity threshold lies around 0.4 for the duration of the transfer. Again the priority lies with the inclination whilst the eccentricity accounts for the majority of the control contribution throughout. Figure 3.1d shows a stronger weighting on the inclination, despite the inclination still having the smallest overall contribution to the control, as seen in Fig. 3.1f. Figure 3.1b also reflects both PSO solutions changing inclination more rapidly at the start, and correcting the eccentricity later. There is a small difference in the propellant consumption, which likely comes from the increased effectivity threshold in the middle of the transfer. In both the time- and mass-optimal transfers, Figs 3.1a and 3.1b show the PSO spline changes the inclination more rapdily than the PSO fixed controller at the start, and then more slowly towards the end. Looking at Figs. 3.1c and 3.1d, it is clear W_i starts of at a similar value but the time-dependence in the PSO spline solutions allows the W_i to be decrease and W_e to be increased in both cases towrards the end of the transfer. From Figs. 3.1e and 3.1f this can be seen in the contributions from $\sqrt{Q_e}$, which are greater towards the end for the PSO spline solutions.

3.2.2 Low Earth Orbit to Geostationary Orbit

Here a LEO-GEO transfer with an inclination change is considered in Keplerian dynamics. The parameters are chosen to allow comparison with Falck *et al.* [58] and Chapter 5 of Conway *et al.* [14]. Similar ones were considered by the authors in [40], [41] and [42]. Their reference solution was obtained using a direct method, where the thrust-steering is parameterised based on the necessary conditions from optimal control theory and the subsequent parameter-optimisation problem is solved using NLP methods. However, orbital averaging methods were used to quickly and efficiently compute multiple powered trajectories during the optimisation process. As in Section 3.2.1, this cannot claim to be the true optimal control solution, but it is a benchmark which is already $\sim 6\%$ better than the classical Q-law and DAG results also presented in Falck *et al.* [58].

The modelled spacecraft has a mass of 1200 kg, a thrust of 0.4017 N and an I_{sp} of 3300 s, giving an initial thrust-to-mass ratio of 0.00033475 m s^{-2}, approximately double the previous test case. The initial and target orbits are shown in Table 3.3. Again a 0.25 day residual is chosen to define convergence to the target orbit.

Table 3.3: Initial and target orbital elements, for a LEO-GEO transfer.

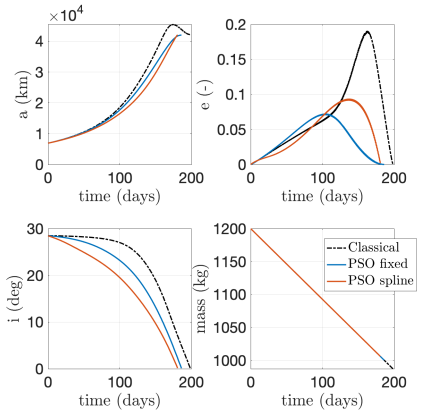
	a (km)	e	i (°)	Ω (°)	ω (°)	ν (°)
Initial	6927	1e-5	28.5	0.0	0.0	0.0
Target	42,164	1e-5	0	*free*	*free*	*free*

Table 3.4: Comparison of the time-dependent Q-law performance for LEO-GEO transfer

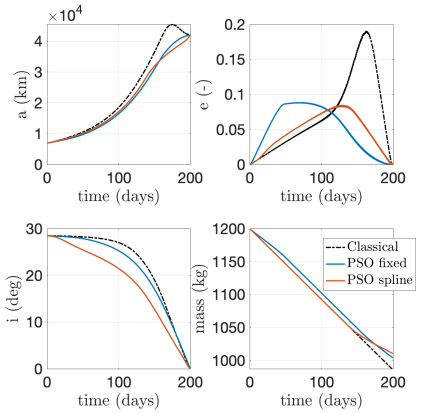
	Method	Time (days)	Propellant (kg)
	Time-optimal Classical	198.32	212.70
Time	Ref. [58]	198.99	-
	PSO fixed	186.10	199.58
	PSO spline	180.68	193.77
Mass	PSO fixed	199.67	196.52
	PSO spline	199.74	189.43

Results are given in Table 3.4. Compared to the GTO-GEO test case the improvement here is very noticeable, with a 5.91 day improvement for the time-optimal transfer, and a 7.08 kg improvement for the mass-optimal case - as seen in Table 3.4. The variation in the PSO fixed solutions was 186.10, 186.10, and 186.23 days. Meanwhile for the PSO free solutions it was 180.68, 180.89, and 180.83 days. Compared to the classical Q-law ($\boldsymbol{W} = \mathbf{1}$) the PSO fixed simulation prioritises the eccentricity term, which is interesting as $\Delta e = 0$ at the start of the transfer. This suggests the PSO is anticipating the need to adjust the eccentricity again later in the transfer, attempting to prevent it from increasing.

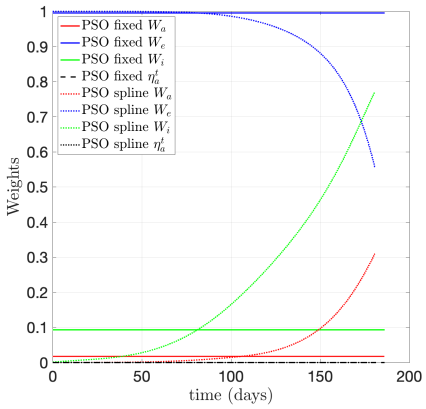
Time- and mass-optimal weight profiles appear similar, with the effectivity accounting for the main change in behaviour. As in the GTO-GEO transfer case, the PSO spline



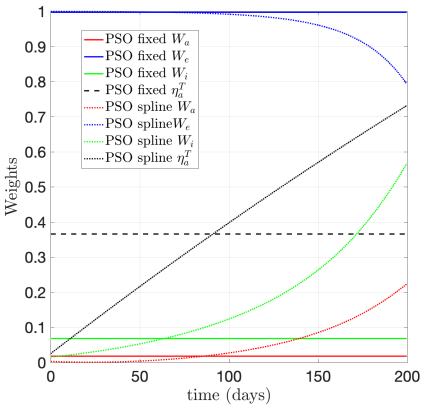
(a) Time-optimal Orbital Parameters.



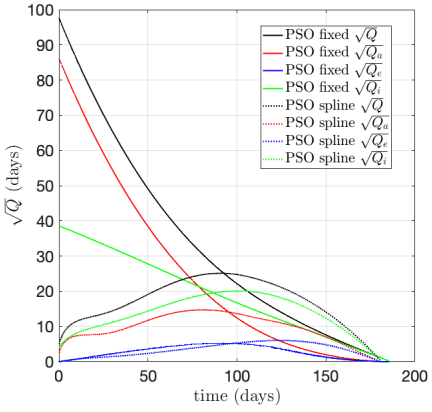
(b) Mass-optimal Orbital Parameters.



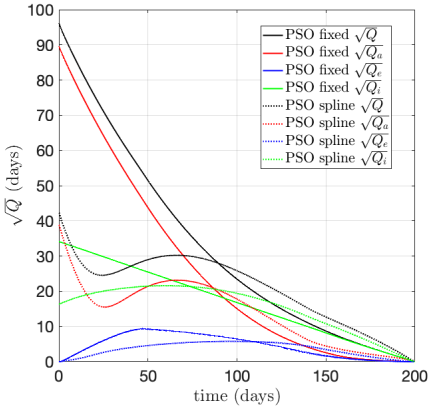
(c) Time-optimal weights.



(d) Mass-optimal weights.



(e) Time-optimal Q-law contributions.



(f) Mass-optimal Q-law contributions.

Figure 3.2: PSO Q-law results for LEO-GEO transfer. Both time-optimal and mass-optimal results are shown for both the PSO fixed and PSO spline approaches.

starts at the same values as the PSO fixed and then changes the behaviour later in the transfer. However, unlike the GTO-GEO transfer, where the PSO increased the contribution from the inclination through W_i, here the PSO prioritises the eccentricity term W_e in order to allow an easier increase in semi-major axis. The dramatic increase in performance between the PSO fixed and spline controllers can be attributed to the change in behaviour of W_i throughout the transfer. This can be seen in the weight profiles in Figs. 3.2c and 3.2d. Figs. 3.2e and 3.2b further confirm this where the fixed cases show the dominance of the semi-major axis to begin with and then a transition to the inclination after ~ 80 days. The PSO spline is able to prioritise the inclination more than the fixed case at the end, whilst also increasing the contribution of the eccentricity towards the end, ensuring a faster convergence. In the mass-optimal case, the plot of η_a^T in Fig. 3.2d demonstrates where the propellant savings are achieved. The engine is on at the beginning to encourage the required increase in semi-major axis, and then propellant is saved whilst the spacecraft converges to the target orbit towards the end. This can also be seen in mass plot in Fig. 3.2b.

3.3 State-dependent parameters

The results in Section 3.2 demonstrate that the optimality of a Lyapunov control law can be improved by first optimising the weights $\boldsymbol{W}$ and secondly allowing these weights $\boldsymbol{W}$ to vary throughout the transfer, $\boldsymbol{W}(t)$. However, a major limitation remains with this approach: it is dependent on the epoch of the transfer. This is a concern from a mission design perspective, and for guaranteeing stability. For mission designers, if the transfer needs to be recomputed at a particular point along the original transfer, it is complicated to extract the value of $\boldsymbol{W}$ at the given state, and replicate the subsequent behaviour and evolution of $\boldsymbol{W}(t)$. In addition, if there is a malfunction during the transfer, and the spacecraft results in a new state $\boldsymbol{X}^*$ which is close but not on the original trajectory, no information remains on the optimal value of $\boldsymbol{W}^*$. Likewise, if there is a missed-thrust event, or another instance where the spacecraft is unable to fire, then the values $\boldsymbol{W}(t)$ will be out-of-sync with the optimal set. No information relating $\boldsymbol{W}(t)$ to the current state $\boldsymbol{X}(t)$ or target state $\boldsymbol{X_T}$ is retained. In short, spacecraft trajectories tend to deviate from the nominal ones due to thrust realisation errors and environmental and platform model discrepancies. These deviations can grow exponentially and require updated guidance frequently.

A suggested solution to this is to make $\boldsymbol{W}$ a function of the state $\boldsymbol{W(X)}$ instead. This retains the advantage of allowing $\boldsymbol{W}$ to vary throughout the transfer, and enables the controller to remain closed-loop whilst independent of the epoch. The trajectory therefore should become less impacted by errors and extends the validity of information supplied from the ground [115]. The issue remains: how to find the functional dependence of $\boldsymbol{W(X)}$.

It was proposed to use a subset of RL algorithms known as PPO. These deal with a continuous state and action space and are based on AC algorithms. A critic evaluates the quality of a given action by approximating the value function using a NN. The actor learns the policy and maps the state to the action the agent should take, often also using a NN.

In the remainder of this section, the following will outlined:

- The impact state-dependent parameters have on Lyapunov stability.
- The proposed actor network and nature of the policy $\boldsymbol{u} \sim \pi_\theta(u|X)$.
- Analytical expression and numerical validation of the state-weight Jacobian.
- Different realisations of the policy $\boldsymbol{u} \sim \pi_\theta(u|X)$ and how they might impact Lyapunov stability.
- Specific details on the RL implementation, along with a pseudocode of the approach.

3.3.1 Impact on Lyapunov stability

As discussed in Section 2.2.2.3, the Q-law is best thought of as a weighted, squared summation of the time required to change the current state $\boldsymbol{X} = [a, e, i, \Omega, \omega]^\mathrm{T}$ to the target state $\boldsymbol{X}_T = [a_T, e_T, i_T, \Omega_T, \omega_T]^\mathrm{T}$. Following on from Section 2.2.2, a stable control is one that ensures $\dot{Q} < 0$ throughout the transfer. One way of doing this is to select a controller that minimises the rate of change of the Lyapunov function (in this case the most negative value). Classically, this derivative is only a function of state $\boldsymbol{X}$, but that is no longer the case as soon as the Q-law weights are allowed to vary. More specifically, if the weights are assumed to be state dependent, $\boldsymbol{W}(\boldsymbol{X})$, a new term in the total derivative of the Lyapunov function suddenly emerges:

$$\begin{aligned}
\dot{Q} &= \dot{Q}_{\boldsymbol{X}} + \dot{Q}_{\boldsymbol{W}} \\
&= \frac{\partial Q}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\dot{\boldsymbol{W}} \\
&= \frac{\partial Q}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} \\
&= \left(\frac{\partial Q}{\partial \boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}\right)\boldsymbol{B}\boldsymbol{u},
\end{aligned} \tag{3.4}$$

where a control vector $\boldsymbol{u}$ is acting as the perturbing acceleration $\boldsymbol{a}_d$ in Eq. (2.4). When the weights are held constant, $\partial \boldsymbol{W}/\partial \boldsymbol{X} = \boldsymbol{0}$ and the control is derived as:

$$\boldsymbol{u}_{\text{orig}} = -f\frac{\boldsymbol{B}^\mathrm{T}\boldsymbol{M}^\mathrm{T}}{\|\boldsymbol{M}\boldsymbol{B}\|} \quad \text{where} \quad \boldsymbol{M} = \left(\frac{\partial Q}{\partial \boldsymbol{X}}\right), \tag{3.5}$$

with $f = T/m$ where T is the engine thrust and m the spacecraft mass. However, when $\partial \boldsymbol{W}/\partial \boldsymbol{X} \neq \boldsymbol{0}$ the control can be modified to account for this as:

$$\boldsymbol{u}_{\text{jac}} = -f \frac{\boldsymbol{B}^{\mathrm{T}} \boldsymbol{M}^{\mathrm{T}}}{\|\boldsymbol{M}\boldsymbol{B}\|} \quad \text{where} \quad \boldsymbol{M} = \left(\frac{\partial Q}{\partial \boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}} \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}} \right). \tag{3.6}$$

As it can be seen, the inclusion of the second term in Eq. (3.6) may play an important role in determining the sign of $\dot{Q}$ and, therefore, the stability of the Lyapunov controller. However, its inclusion is often overlooked in existing work in the literature that aims at improving the optimality of Lyapunov controllers [74]. A very thorough discussion on the implications of this for trajectory design using nonlinear control can be found in Chapter 2 and [109].

The following section explains how analytical expressions for the Jacobian matrix of the state-dependent weights can be derived using available information from the actor network. The product of this investigation is a state-dependent Lyapunov controller that enforces stability throughout the whole trajectory.

3.3.1.1 Actor Network Structure

Actor networks and their importance to RL algorithms was discussed in Section 2.4.2.1. In this section, the actor network architecture used for this work is described. In general, the actor network is a NN that maps from states x_i to actions a_i. It defines a policy $\pi_\theta(a|x)$ that determines the action $a \sim \pi_\theta(a|x)$ taken by the agent whilst it interacts with the environment. Given the formulation of the Q-law, and other Lyapunov controllers, a natural choice is to have an actor network that maps between input state $\boldsymbol{X}$ and weights $\boldsymbol{W}$, in addition to the effectivity threshold η_a^t. Hence, the states x_i are be given by the COEs $\boldsymbol{X}$ and the resulting actions a_i are given by $\boldsymbol{W} = [W_a, W_e, W_i, W_\Omega, W_\omega, \eta_a^t]^{\mathrm{T}}$. A given realisation of this is presented in the remainder of this section.

The input vector $\boldsymbol{X}$ is first projected onto a grid of RBFs, which act as features/activation functions for the network. These RBFs are Gaussian functions and define a projection of the state onto their centres - see Eq. (3.8) below. These will be discussed in more details in Section 3.3.1.4 - the interested reader can refer to Fig. 3.6a. Each has a centre at a fixed and unchanging location $\boldsymbol{c}_d = [a_d, e_d, i_d, \Omega_d, \omega_d]^{\mathrm{T}}$. As such, there are D features in the NN, each with its own centre:

$$\boldsymbol{C}_{\boldsymbol{X}} = \left[\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \ldots, \boldsymbol{c}_d, \ldots, \boldsymbol{c}_D \right]^{\mathrm{T}}. \tag{3.7}$$

Given they are RBFs, these activation functions can be written as $\boldsymbol{\psi} = [\psi_1, \psi_2, \psi_3, \cdots, \psi_D]^{\mathrm{T}}$ with scalar components and derivatives with respect to state $\partial \boldsymbol{\psi}(\boldsymbol{X})/\partial \boldsymbol{X}$ given by:

$$\psi_d(\boldsymbol{X}) = \exp\left(-\frac{1}{2} \frac{\|\boldsymbol{X} - \boldsymbol{c}_d\|^2}{\sigma_{\text{net}}^2} \right), \tag{3.8}$$

$$\frac{\partial \psi_d(\boldsymbol{X})}{\partial \boldsymbol{X}} = \psi_d(\boldsymbol{X}) \frac{-(\boldsymbol{X} - \boldsymbol{c}_d)^{\mathrm{T}}}{\sigma_{\mathrm{net}}^2}, \tag{3.9}$$

where σ_{net} is the standard deviation of each RBF. The $\boldsymbol{\psi}$ vector is multiplied by a $D \times 6$ matrix of parameters θ to give the first in a sequence of outputs which will result in the weights vector:

$$\hat{\boldsymbol{W}} = \theta^{\mathrm{T}} \boldsymbol{\psi}(\boldsymbol{X}), \tag{3.10}$$

$$\frac{\partial \hat{\boldsymbol{W}}}{\partial \boldsymbol{X}} = \theta^{\mathrm{T}} \left(\frac{\partial \boldsymbol{\psi}(\boldsymbol{X})}{\partial \boldsymbol{X}} \right). \tag{3.11}$$

The output is then bounded between 0 and 1 using the hyperbolic tangent function, and this derivative is also readily obtained:

$$\bar{W}_i = f_{\mathrm{scale}}(\hat{W}_i) = \frac{1}{2} \Big(\tanh(\beta_f \hat{W}_i) + 1 \Big), \tag{3.12}$$

$$\frac{\partial \bar{W}_i}{\partial \hat{W}_j} = \begin{cases} \frac{\beta_f}{2} \left(\operatorname{sech}^2(\beta_f \hat{W}_j) \right) & \text{if } i = j \\ 0 & \text{if } i \neq j, \end{cases} \tag{3.13}$$

where β_f is a scaling factor that can affect sensitivity of θ on $\boldsymbol{W}$, here set to 2π. A stochastic contribution is used during training to allow the agent to explore the environment. This is chosen to be a normal distribution with mean $\bar{W}_i$ and variance σ^2, and is denoted as $\mathcal{N}(\bar{W}_i, \sigma)$. This is switched off during validation/deployment. As this is symmetric about its mean value, the derivative is the identity:

$$W_i \in \mathcal{N}(\bar{W}_i, \sigma) \qquad \text{with} \qquad \frac{\partial \boldsymbol{W}}{\partial \bar{\boldsymbol{W}}} = \mathbf{1}. \tag{3.14}$$

Note that as $0 \leq \boldsymbol{W} \leq 1$ issues arises when $W_i = \mathcal{N}(\bar{W}_i, \sigma) < 0$ or > 1. This is dealt with by re-sampling the distribution until $0 \leq W_i \leq 1$. This is elaborated on further in Section 3.3.2. The structure of the actor network is shown in Fig. 3.3.

Section 2.4.2.2 indicates how the parameters of this actor network, θ, are updated using the PPO algorithm. The term $\nabla_\theta \log \pi_\theta$ indicates the gradient logarithm of the policy and can now be computed as a result of the structure outlined in this section. Following on from Eq. (3.14), the probability of a stochastic action W_i given a deterministic output $\bar{W}_i$ is be given by a Gaussian distribution. Hence, one can write

$$\pi_\theta \propto \exp\left(-\frac{1}{2} \frac{(\boldsymbol{W} - \bar{\boldsymbol{W}}(X, \theta))^2}{\sigma^2} \right), \tag{3.15}$$

$$\nabla_\theta \log \pi_\theta = \frac{\nabla_\theta \pi_\theta}{\pi_\theta} = \frac{(\boldsymbol{W} - \bar{\boldsymbol{W}}(X, \theta))}{\sigma^2} \nabla_\theta \bar{\boldsymbol{W}}(X, \theta). \tag{3.16}$$

This derivative is available from the actor network thanks to Eqs. (3.11) and (3.13).

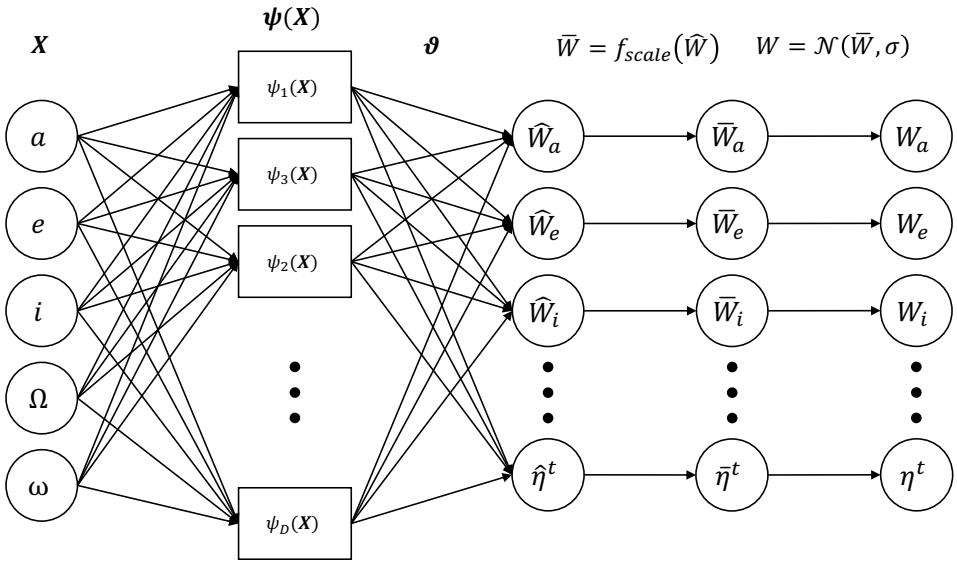


Figure 3.3: Structure of the actor network. It maps from inputs $\boldsymbol{X}$ to output weights $\boldsymbol{W}$ and effectivity threshold η^t.

3.3.1.2 Analytical expression for state-weight Jacobian

This structure enables one to retain the information on $\partial \boldsymbol{W}/\partial \boldsymbol{X}$ analytically, providing access to the state-weight Jacobian matrix throughout the transfer. This is necessary for computing $\boldsymbol{u}_{\text{jac}}$ via Eq. (3.6). The full expression for the Jacobian is determined using the chain rule as:

$$\frac{\partial W_i}{\partial X_j} = \frac{\partial W_i}{\partial \bar{W}_m} \frac{\partial \bar{W}_m}{\partial \hat{W}_n} \frac{\partial \hat{W}_n}{\partial X_j}, \tag{3.17}$$

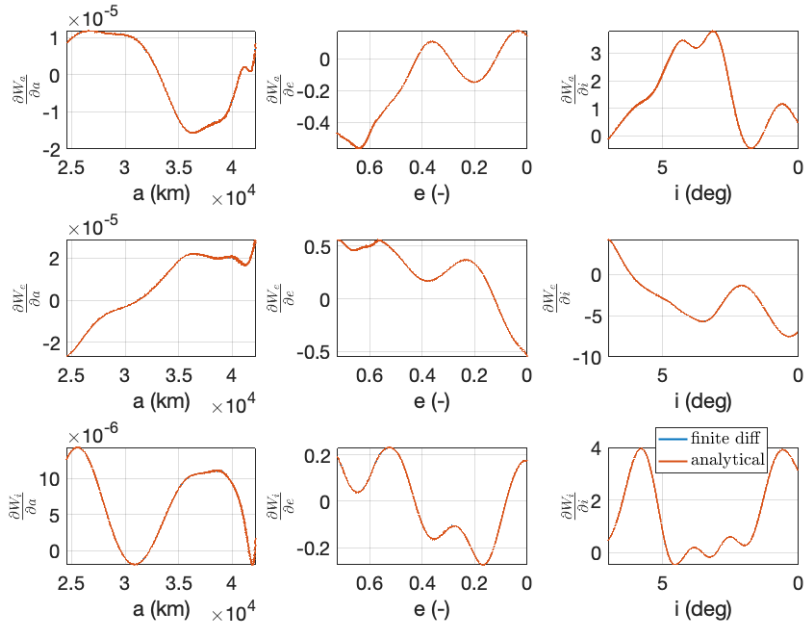
where the Einstein summation convention is used to indicate summing over m and n. Combining the contributions from Section 3.3.1.1, the Jacobian is given by:

$$\frac{\partial W_i}{\partial X_j} = \frac{\beta_f \operatorname{sech}^2(\beta_f \hat{W}_i)}{2} \sum_k \theta_{ki} \left(\frac{\partial \psi_k(\boldsymbol{X})}{\partial X_j} \right). \tag{3.18}$$

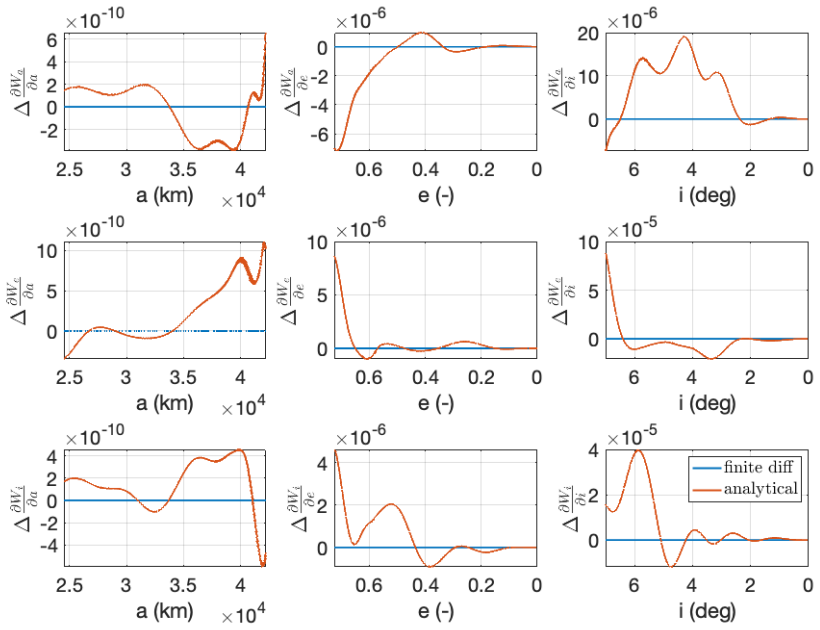
3.3.1.3 Numerical validation for state-weight Jacobian

This expression for the Jacobian was validated in MATLAB using both a finite difference approach, where small changes in state are reflected in the weights, and a simplified lower-dimensional problem.

Figure 3.4a shows both the finite difference and analytical expressions for a GTO-GEO transfers, and Fig. 3.4b shows the different between the two approaches. Both are computed for $\boldsymbol{u}_{\text{orig}}$ because the finite difference approach has to be done *a posteriori*. The likely explanation for any difference on the order of 10^{-5} comes from the inaccuracies of the finite difference approach.



(a) $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$



(b) $\Delta \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$

Figure 3.4: Numerical and analytical comparison for the Jacobian $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$, computed *a posteriori* for a GTO-GEO transfer using $\boldsymbol{u}_{\text{orig}}$ control.

As was demonstrated in Section 3.2, varying the weights $\boldsymbol{W}$ throughout the transfer can improve optimality. It is also important to note the frequency at which $\boldsymbol{W}$ vary does not have to match the control frequency. Given the performance of a classical Q-law, $\boldsymbol{W}$ can theoretically remain fixed for a transfer.

During training there is an interval Δt at which the weights $\boldsymbol{W}$ are updated. Thus, the

values $\boldsymbol{W}$ and $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$ computed at a given state are used along with the Q-law to compute the control. However, if allowed to vary, then $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$ computed at $\boldsymbol{X}_0$ is not necessarily valid for all $\boldsymbol{X}$. There is a range of validity during which $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}|_{t-\Delta t}$ remains a sufficient approximation for the true $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}|_t$. This range of validity, indicated by Δt is considered in Fig. 3.5. The interval during which $\boldsymbol{W}$ and $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$ remain fixed is plotted against the time-of-flight for a pre-trained network. It can be seen that any $\Delta t < 16$ hours has limited effect on the transfer duration. Once training is completed, it is possible to embed the actor network directly with the control and this issue goes away. However, it needs to be considered during training.

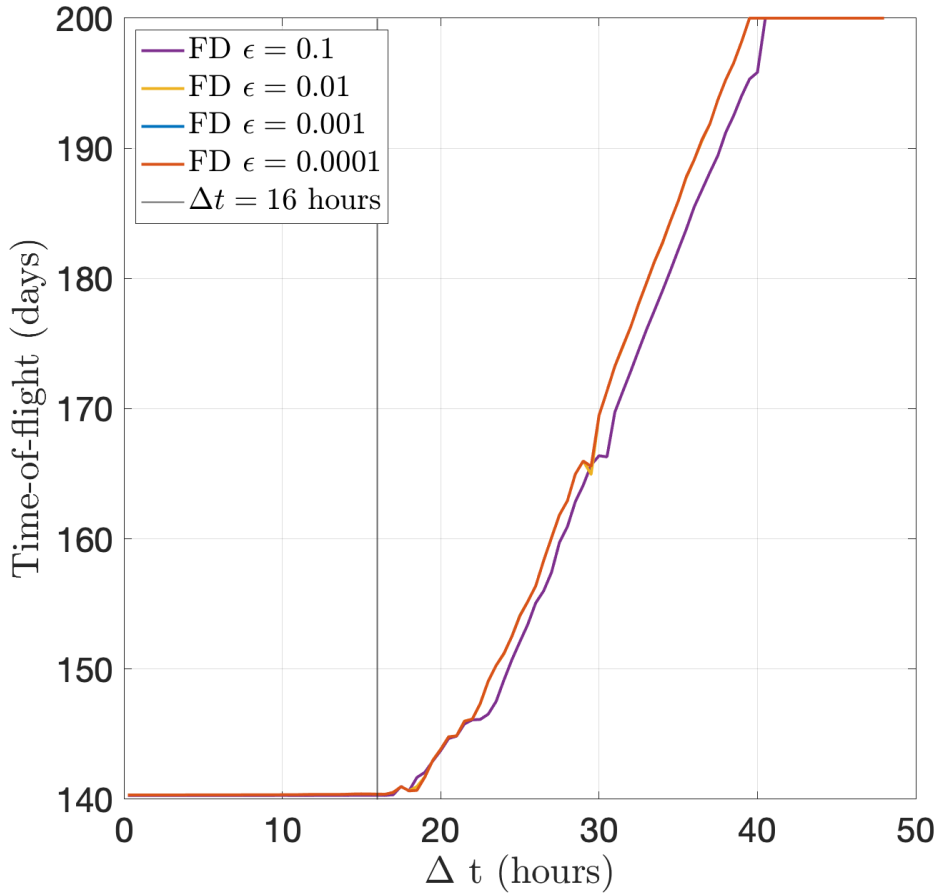


Figure 3.5: Region of Validity for $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$ for a GTO-GEO transfer. Here the Jacobian $\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}$ is computed with a finite difference (FD) method with varying the step size ϵ. All FD methods except $\epsilon = 0.1$ overlap with each other.

3.3.1.4 Effect of the Network Design

As discussed earlier, the actor network governs the relationship between the state and action taken. It is worth stressing the non-negligible effect the architecture has on the performance of the subsequent implementation. The most influential are the type of basis function ψ, the number of neurons (functions) D and the overlap between activation

functions σ_{net}. As indicated by Eq. (3.7), the activation functions are evenly spaced over a grid of orbital elements. As such, it is easier to refer to the number of functions per orbital element, N_X. Two types of activation function are considered, RBFs:

$$\psi_{\text{RBF}}(\boldsymbol{X}) = \frac{1}{\prod_X (N_X - 1)} \frac{1}{\sqrt{2\pi\sigma_{\text{net}}^2}} \exp\left(-\frac{1}{2}\frac{\|\boldsymbol{X} - \boldsymbol{c}_d\|^2}{\sigma_{\text{net}}^2}\right), \tag{3.19}$$

and triangular functions (TRI)

$$\psi_{\text{TRI}}(\boldsymbol{X}) = \frac{(\|\boldsymbol{X} - \boldsymbol{c}_d\| < \sigma_{\text{net}})}{\prod_X (N_X - 1)} \frac{1}{\sigma_{\text{net}}} \left(1 - \frac{\|\boldsymbol{X} - \boldsymbol{c}_d\|}{\sigma_{\text{net}}}\right). \tag{3.20}$$

As seen in Fig. 3.3, the behaviour of $\boldsymbol{W}$ is governed by $\theta^{\text{T}}\boldsymbol{\psi}$. In order to ensure the state-dependent behaviour is dominated by the parameters to be learnt, θ, σ_{net} is calculated to minimise the $L2$-norm between 1 and $\sum_{N_X} \psi$, using an approach similar to [116]. This gives much better performance than using $\sigma_{\text{net}} = 1/(N_X - 1)$ or using the full-width half-maximum. In the triangular-network case, an integer σ_{net} is required, thus, for consistent comparison this value is simply rounded to the closest non-zero integer.

Inputs and outputs are normalised in the network using the range of orbital elements explored by the classical Q-law. If $[a_{\text{class}}(t), e_{\text{class}}(t), i_{\text{class}}(t)]$ is the full set of states visited by the classical Q-law, the ranges are given by

$$a_{\min} = 0.75 \min_t(a_{\text{class}}(t)), \tag{3.21a}$$

$$a_{\max} = 1.25 \max_t(a_{\text{class}}(t)), \tag{3.21b}$$

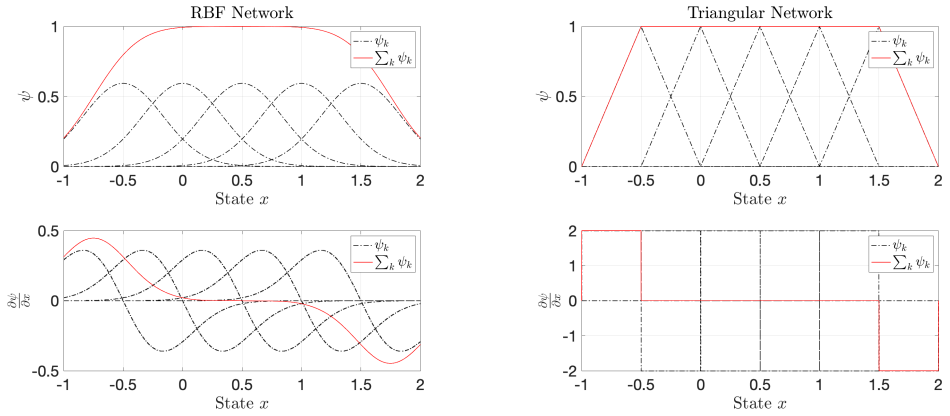
$$e_{\min} = \max(\min_t(e_{\text{class}}(t)) - 0.2, 0), \tag{3.21c}$$

$$e_{\max} = \min(\max_t(e_{\text{class}}(t)) + 0.2, 1), \tag{3.21d}$$

$$i_{\min} = 0.75 \min_t(i_{\text{class}}(t)), \tag{3.21e}$$

$$i_{\max} = 1.25 \max_t(i_{\text{class}}(t)). \tag{3.21f}$$

This ensures resources are not wasted on states that the controller is unlikely to visit. However, in many transfer cases, the initial and target values might be the same. This is avoided by introducing the ±25% margin. If one of the edge neurons C_X is equal to X_T then chattering is observed whilst trying to converge to the target orbit. This is due to the strong influence of the individual θ parameter associated to C_X. This is prevented by defining the network based on Eqs. (3.21) and adding an additional neuron at either end of the network to ensure the observed behaviour occurs inside the network. These additional neurons are added before training starts and can be non-physical, e.g., centred on a negative eccentricity values. A useful byproduct of this is that any edge effects from the sum over ψ_d and their derivative can be reduced. Note, the state of the spacecraft remains physical throughout the learning process.



(a) Radial Basis Function Network (b) Triangular Basis Function Network

Figure 3.6: Actor network architectures visualised in 1-dimension. They show the sum of the basis functions and their derivative with respect to the normalised state input.

Figure 3.6 shows a 1-dimensional representation of these functions and their derivatives when $N_X = 5$. The individual basis functions are shown, along with their sum for $\theta = 1$. By adding an additional neuron outside the domain of interest, the edge effects are reduced within the $0 < x < 1$ domain. Note in the RBF case there is a non-negligible contribution from $\partial\boldsymbol{\psi}/\partial\boldsymbol{X}$ even when $\theta = 1$. As such, when including the Jacobian term in the control, it is not fully determined by θ. This was one of the motivations in exploring the triangular network, where $\partial\boldsymbol{\psi}/\partial\boldsymbol{X}$ is fully determined by the parameters θ.

3.3.2 Reinforcement Learning Framework and Pseudocode

Figure 3.7 illustrates the proposed RL framework diagrammatically. Starting from a state $\boldsymbol{X}_i$ sampled from a distribution $\mathcal{N}(\boldsymbol{X_0}, \sigma_0)$ and desired target state $\boldsymbol{X}_T$, a Lyapunov control law is used to compute the control at a given time $\boldsymbol{u}(t)$. This Lyapunov control law has user-defined parameters $\boldsymbol{W}_X$ and η_a^T. The suggestion is these are made state-dependent using the actor network as described in Section 3.3.1.1, which will take $\boldsymbol{X}_i$ as its input. The relationship between $\boldsymbol{W}_X$, η_a^T and $\boldsymbol{X}_i$ at a given iteration k is given by the policy π_{θ_k} via Eq. (3.15). Once the control has been determined, the state is then integrated in the equations of motion to give the state at the next time step, $\boldsymbol{X}_{i+1}$. This is done iteratively until the controller converges to the target orbit. It is important to note the frequency at which the actor network and the Lyapunov control law are called does not need to match, and instead it is easier to decouple these during the learning process. Once a batch of trajectories have been computed, a critic network evaluates the Value function - seen Eq. (2.66). From here, the Advantage function can be computed using Eq. (2.72) and the PPO update for the parameters θ_k determined via Eqs. (2.74) and (2.75). This outer-loop is then repeated iteratively to learn the optimal policy π_θ. An algorithmic pseudocode is show in Algorithm 1.

During training, two sets of parameters are used: the active θ_k and the newly up-

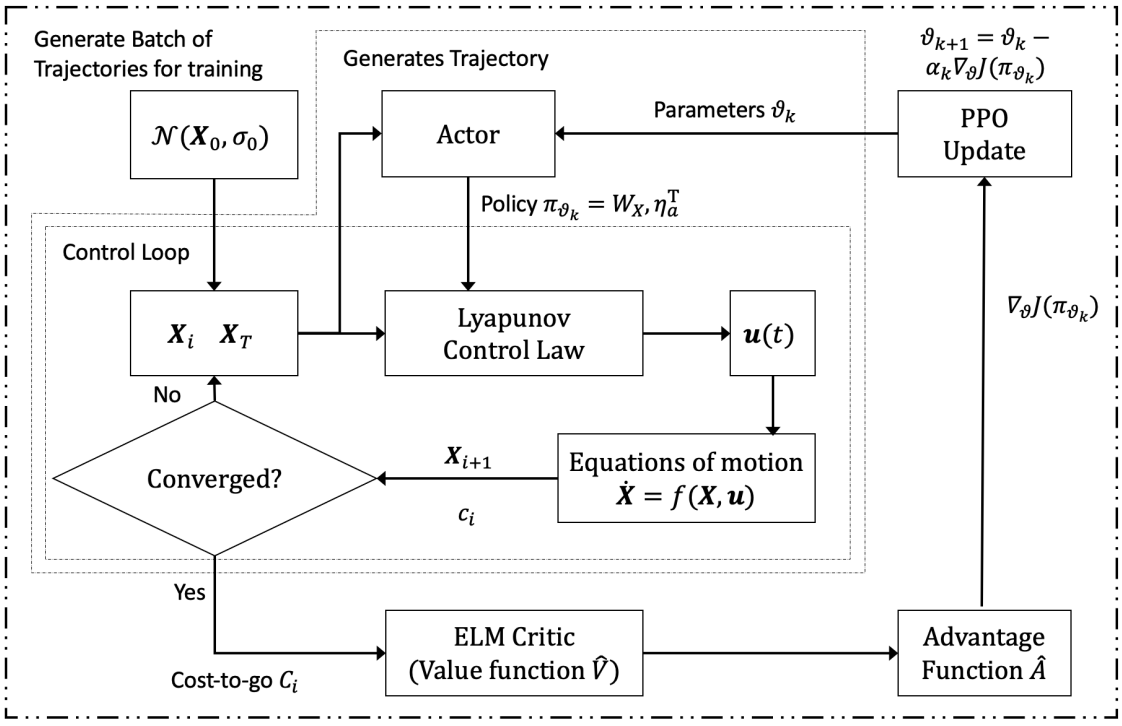


Figure 3.7: Reinforced Lyapunov Controller Flowchart.

dated θ. A batch of N trajectories τ are generated using the stochastic policy π_{θ_k}. In addition, two deterministic trajectories $\tau(\theta_k)$ and $\tau(\theta)$ are computed. See Fig. 3.8 for a visual interpretation. Using the costs associated with stochastic trajectories $\tau(\pi_{\theta_k})$ one can iteratively update θ. Conventionally the updates on θ and θ_k occur after a predetermined *mini-batch size*. This limitation can result in a bad update and inevitably make it difficult for the algorithm to converge to an optimal solution. In this work, the update $\theta \leftarrow \theta - \alpha_k \nabla_\theta J(\theta)$ occurs every iteration (i.e., after a batch of N trajectories). However, the update $\theta_k \leftarrow \theta$ only occurs if $\tau(\theta)$ outperforms $\tau(\theta_k)$ (namely if $C(\tau(\theta)) < C(\tau(\theta_k))$).

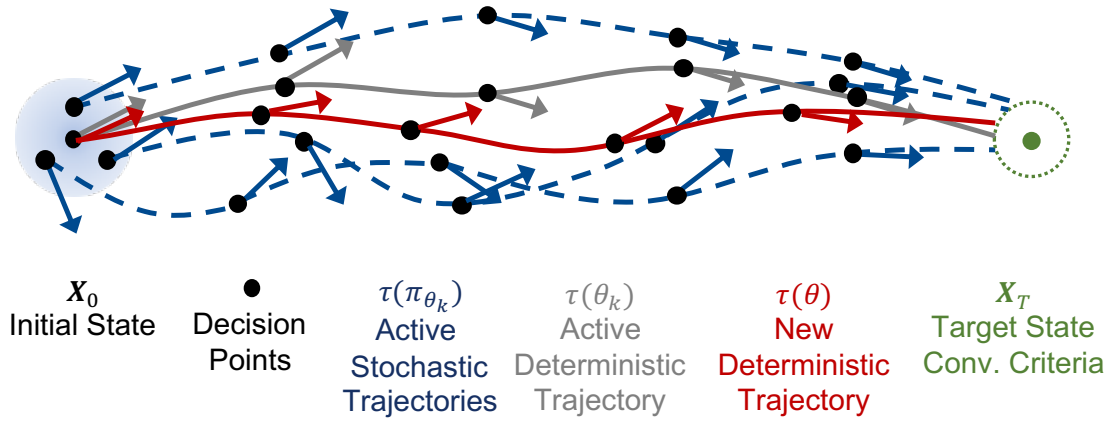


Figure 3.8: Diagram indicating the different trajectories computed during one learning iteration k. Arrows indicate actions taken by the RL algorithm. Although the control vector is computed at each integration step, the RL decisions are only taken every 0.25 days, indicated by the decision points.

Algorithm 1 Reinforcement Learning Pseudocode

Set initial state $\boldsymbol{X}_0$ and target state $\boldsymbol{X}_T$
Set random parameters θ_0
for k = 1: Max Iterations **do** ▷ Learning Iteration
  for n = 1: Number of Episodes + 2 **do** ▷ Minibatch (can parallelise)
    for i = 1: Number of Steps - 1 **do** ▷ RL Decision Steps
      if N ≤ Number of Episodes **then** ▷ Stochastic θ_k
        $\boldsymbol{W}_i$, $\partial \boldsymbol{W}_i/\partial \boldsymbol{X}_i, \nabla_{\theta_k} \log \pi_{\theta_k}$, $R \leftarrow$ ActorNetwork($\boldsymbol{X}_i$, $\pi(\theta_k)$, $\pi(\theta)$)
      else if N = Number of Episodes + 1 **then** ▷ Deterministic θ_k
        $\boldsymbol{W}_i$, $\partial \boldsymbol{W}_i/\partial \boldsymbol{X}_i \leftarrow$ ActorNetwork($\boldsymbol{X}_i$, θ_k)
      else ▷ Deterministic θ
        $\boldsymbol{W}_i$, $\partial \boldsymbol{W}_i/\partial \boldsymbol{X}_i \leftarrow$ ActorNetwork($\boldsymbol{X}_i$, θ)
      while $t < t_{i+1}$ **do** ▷ Integrate
        $\boldsymbol{u}_t \leftarrow$ Lyapunov($\boldsymbol{X}_t$, $\boldsymbol{X}_T$, $\boldsymbol{W}_i$, $\partial \boldsymbol{W}_i/\partial \boldsymbol{X}_i$)
        t, $\boldsymbol{X}_{t+1} \leftarrow$ Dynamics($\boldsymbol{X}_t$, $\boldsymbol{u}_t$)
      Assign costs c_i
    Calculate cost-to-go for each step C_i
    $\hat{V} \leftarrow$ CriticNetwork($\boldsymbol{X}_i$, C_i)
    $\hat{A} \leftarrow$ AdvantageFunction($\boldsymbol{X}_i$, c_i, $\hat{V}$)
  Evaluate $\nabla_{\theta_k} J(\pi_{\theta_k})$ using clipped objective ▷ PPO update
  Update: $\theta \leftarrow \theta - \alpha_k \nabla_{\theta_k} J(\pi_{\theta_k})$
  if $\sum_i C_i(\theta) < \sum_i C_i(\theta_k)$ **then** ▷ Update active policy
    $\theta_k \leftarrow \theta$
  if No change in $\sum_i C_i(\theta_k)$ for K iterations **then** ▷ Sliding window for convergence
    break
Deploy using final θ_k

Each trajectory is divided into fixed time-intervals ensuring the cost associated with each state-action pair is not determined by the time interval. At the start of an interval, the actor network is called with the current state to determine $\boldsymbol{W}$ and $\partial \boldsymbol{W}/\partial \boldsymbol{X}$, which are then kept fixed for the following time-interval. There is a trade-off here: on one hand a smaller time-step allows for more frequent variation of the weights, potentially improving the response by allowing rapid changes in behaviour. However, it also increases the complexity of the learning process. As such, an interval of 0.25 days was found to be most suitable during training. During validation and/or deployment the actor network can be called at the frequency required by the integrator (i.e., embedded) or the satellite operator and, as such, the weights are free to vary at a faster rate. A different transfer problem, particularly one where the expected time-of-flight is much shorter, might require a different time-interval to be selected.

Table 3.5 gives a summary of the parameters used. 1500 was found to be a suitable number of neurons for the critic network and also aligns with a common rule of thumb of 1/10th of samples used [117]. One of the major challenges in RL algorithms is the exploration-vs-exploitation dilemma. Here this is encapsulated in the parameter σ, which governs the size of the Gaussian noise added to $\boldsymbol{W}$ at each decision-step during

the training process. As $0 \leq \boldsymbol{W} \leq 1$, a nominal value of $\sigma = 0.03$ allows the algorithm to explore approximately 10% of the action space at any given moment. Using a constant value works very well and has achieved satisfactory results in previous work [40, 41].

However, improved results can be obtained when using an adaptive approach. Hence, starting with $\sigma = 0.1$, the algorithm explores until 50 consecutive iterations occur without improvement, then σ is reduced to 0.03. Again if 50 iterations occur without improvement then σ is reduced for a second time to 0.01. If this cycle repeats twice without any improvement then convergence is declared and the algorithm terminates. A fixed learning rate is used throughout. Note this does not guarantee convergence of the algorithm, however, it is found to work well in many environments and prevents early convergence [46].

Computationally it was not feasible to run multiple learning processes for each scenario considered in the remained of the thesis. However, five separate learning processes were run for the GTO-GEO and LEO-GEO time-optimal RL Q-law (RBF) and RL Q-Jac (RBF) solutions to assess the variability of solutions and to tune the convergence criteria. For GTO-GEO transfers, this was less than 0.25 days standard deviation, i.e. within one RL decision step, and less than 0.5 days, i.e. within two RL decision steps for the LEO-GEO transfer. Holt *et al.* [41] demonstrated ten training runs gives a range of ±0.26%, ±0.25% for GTO-GEO RL Q-law (RBF) and RL Q-Jac (RBF) respectively, and ±0.40%, ±0.15% for LEO-GEO RL Q-law (RBF) and RL Q-Jac (RBF) respectively. Hence, whilst the best solution might be improved upon, this gives confidence that the performance is repeatable within the confines of a stochastic methodology.

One issue that arises is how to deal with boundaries on the action space. For the Q-law implementation it is necessary to bound $W \geq 0$ and given it is normalised it is preferable to add the additional constraint $W \leq 1$. As such, it is necessary to transform an infinite space in θ to a finite one using f_{scale} in the network architecture. However, taking this output and adding a stochastic element can result in W outside the desired range. Hence, the stochastic noise is generated using σ unless $W < 3\sigma$ or $(1 - W) < 3\sigma$, in which case $\sigma \leftarrow W/3$ or $(1 - W)/3$ respectively. These distributions are re-sampled until $0 \leq W \leq 1$.

Due to the feedback nature of the Lyapunov control laws and the fixed magnitude of the control, defining convergence to the target orbit is required. In the past a maximum residual on the orbital elements was used, however, here a more flexible criteria based on the expected time-to-go is implemented instead. The ratio $\delta(X, X_T)/\max_\nu(\dot{X})$ gives a measure of the time-to-go and simulations show this reduces the likelihood of chattering compared to orbital element residuals.

Table 3.5: Table of parameters for RL Q-law implementation. Standard Q-law parameters from Ref. [36] are denoted with *petro*.

Parameter	Value	Parameter	Value	Parameter	Value
Time-interval, Δt	0.25 days	γ	1	σ_{net}	L2-norm Optimised [116]
Learning rate, α_k	2E-2,2E-3	N	24 (22+2)	Critic activation	sigmoid
σ	0.1, 0.03, 0.01	Grid spacing	5	# critic neurons	1,500
σ interval	50 iterations	# actor neurons	125	ϵ	0.2
r_{petro}	2	k_{petro}	100	b_{petro}	0.01
$r_{\text{petro}}^{\text{p-min}}$	6578 km	m_{petro}	3	n_{petro}	4

3.4 Results

Low-thrust time-optimal and mass-optimal transfers from GTO-GEO and LEO-GEO are considered in Keplerian dynamics. Here, and henceforth in the manuscript, RL Q-law refers to the Q-law with state-dependent weights using $\boldsymbol{u}_{\text{orig}}$ given by Eq. (3.5). Similarly, RL Q-Jac refers to the Q-law with state-dependent weights using $\boldsymbol{u}_{\text{jac}}$ given by Eq. (3.6). The goal is to compare the performance of the RL Q-law and RL Q-Jac state-dependent controllers with the literature, constant and time-dependent Q-law approaches. As such, there are three distinct benchmark cases: the classical Q-law, the PSO Fixed and PSO Spline approaches. The latter provides a suitable benchmark for the RL Q-law approach, as both PSO approaches use Eq. (3.5). The RL control, with its state-dependent weights, is not expected to outperform the PSO spline for a deterministic state - see Section 3.2.

For the RL Q-law and RL Q-Jac approaches, defining the cost function for a particular objective is key. In the time-optimal case, the cost function is simply the final time-of-flight t in days. If the trajectory does not converge within the allowed maximum integration time, then a penalty term on the estimated residual time-to-go is used, t_{res}. This is estimated using the time-to-go formulation of the Q-law - see Eq. (2.42) - from the final orbit element state reached to the target orbit. Using t_{conv} to denote the convergence criteria, the cost function can be written as:

$$c_{\text{time}} = \begin{cases} t, & t_{\text{res}} \leq t_{\text{conv}}, \\ t + 5t_{\text{res}}, & t_{\text{res}} > t_{\text{conv}}. \end{cases} \tag{3.22}$$

In the mass-optimal case, the cost function is written in-terms of propellant mass used. Note that the mass-optimal trajectories presented here have an upper bound on the time-of-flight, and, as such, a penalty term on the final time is included to enforce this. This upper bound is denoted t_{aim} and is simply chosen either from literature for appropriate comparison or by the mission designer for a desirable time-of-flight. For convenience and consistency across all cost terms, this time penalty and the time-to-go residual are converted into propellant mass units:

$$c_{\text{mass}} = \begin{cases} m, & \text{for } t \leq t_{\text{aim}} \\ & \text{and } t_{\text{res}} \leq t_{\text{conv}}, \\ m + \frac{T}{I_{sp} g_0}\Big[(t - t_{\text{aim}})\Big], & \text{for } t > t_{\text{aim}} \\ & \text{and } t_{\text{res}} \leq t_{\text{conv}}, \\ m + \frac{T}{I_{sp} g_0}\Big[(t - t_{\text{aim}}) + 5t_{\text{res}}\Big], & \text{for } t > t_{\text{aim}} \\ & \text{and } t_{\text{res}} > t_{\text{conv}}. \end{cases} \tag{3.23}$$

3.4.1 Geostationary Transfer Orbit to Geostationary Orbit

Here a GTO-GEO transfer with an inclination change is considered in Keplerian dynamics - see Section 3.2.1 for parameters and benchmark discussion. Results are given in Table 3.6. It is clear from the PSO fixed simulations that the optimality in terms of time-of-flight can be improved compared to the time-optimal classical Q-law by tuning the weights. The RL Q-law gives a time-of-flight of 137.14 days. Adding the estimated time-to-go of 0.25 days from the convergence criteria, it matches the result found by Yang *et al.* [74]. The RL Q-Jac gives a time-of-flight of 137.45 days whilst also guaranteeing Lyapunov stability. State-dependent weights result in a minor increase in optimality compared to the optimised constant weight solution. The simulations using the triangular network give very similar results, indicating different network architectures can be accommodated in the algorithm.

Table 3.6: Comparison of state-dependent Q-law performance for GTO-GEO transfer.

	Method	Time (days)	Propellant (kg)
	Time-optimal Classical	144.03	222.06
Time	Ref. [74]	137.3	211.72
	Ref. [114]	137.5	212.00
	PSO fixed	137.90	212.61
	PSO spline	137.16	211.47
	RL Q-law (RBF)	137.14	211.44
	RL Q-Jac (RBF)	137.45	211.93
	RL Q-law (TRI)	137.24	211.59
	RL Q-Jac (TRI)	137.90	212.62
Mass	Ref. [74]	150.00	187.97
	Ref. [114]	150.00	192.00
	PSO fixed	149.75	192.14
	PSO spline	149.75	190.03
	RL Q-law (RBF)	149.68	191.01
	RL Q-Jac (RBF)	149.73	191.82
	RL Q-law (TRI)	149.90	192.33
	RL Q-Jac (TRI)	149.76	191.65

For the mass-optimal simulations, the absolute effectivity η_a was used and a target

time-of-flight, t_{aim}, of 150 days was set. Compared to the time-optimal classical Q-law, the best solution can save 31.05 kg (14.0%) at a cost of 5.65 days (3.9%). Again, there is little difference between the stable RL Q-Jac and RL Q-law. The constant weight solution is 1.13 kg worse than the best state-dependent one.

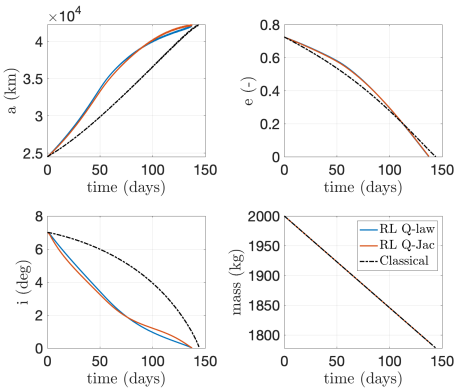
Figure 3.9 shows the RL Q-law and RL Q-Jac results for the time-optimal simulations. Interestingly in Fig. 3.9a it is clear the inclination needs to be prioritised in order to reach GEO with a faster time-of-flight. Figure 3.9c indicates the $\boldsymbol{W}$ learnt by the RL algorithm. However, Fig. 3.9d gives a better indication of which component is dominating the control term. In both cases, the eccentricity is the most influential, followed by the semi-major axis term. Both these have very large changes to undergo. Note the time-history observed in the RL Q-law and RL Q-Jac weights share similarities to the PSO spline approach.

Figures 3.9e and 3.9f indicate the $\dot{Q}$ behaviour throughout the transfer. $\dot{Q}_{\boldsymbol{X}}$ and $\dot{Q}_{\boldsymbol{W}}$ refer to the contributions to $\dot{Q}$ from the state ($\partial Q/\partial \boldsymbol{X}$) and weights ($\partial Q/\partial \boldsymbol{W}$), respectively. Figure 3.9e shows that $\dot{Q}_{\boldsymbol{X}} < 0$ throughout the transfer. However, due to the neglected contribution of $\dot{Q}_{\boldsymbol{W}}$, the true $\dot{Q}$ is greater than zero for short but significant periods during the middle of the transfer. It follows that one of the major requirements of Lyapunov control theory is violated, leading to potentially unsafe behaviour. This lack of guaranteed stability could be a major concern to mission operators, but the introduction of the analytical Jacobian removes this issue with little compromise on optimality.

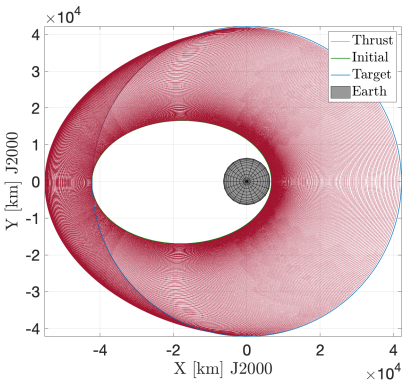
The physical effect of $\dot{Q} > 0$ is an increase in the Lyapunov function value, increasing the expected time-to-go for the transfer. However, this does not always correspond to an increased difference between current and target orbital elements, as the denominator in Eq. (2.42) needs to be considered. In the RL Q-law case it is possible to generate pathological setups of W which causes Q to increase through the transfer and not converge to the target orbit, although this is very rare during the learning process as non-converging transfers are easily discarded. The addition of the Jacobian removes this issue entirely. The current results are obtained using osculating orbital elements. If averaged dynamics are used, it is possible $\dot{Q}$ would appear less than zero for the duration of the transfer for both $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$, as each violation is less than one orbital period - as seen in Fig. 3.9e. In the case of $\boldsymbol{u}_{\text{orig}}$ it is unclear if that would be telling the full story.

In the mass-optimal case, similar plots are given in Fig. 3.10. Figure 3.10b shows coast arcs occurring predominantly at the osculating apoapses, matching the intuitive understanding of these kinds of orbital transfers. Again Figs. 3.10e and 3.10f show that $\boldsymbol{u}_{\text{jac}}$ ensures $\dot{Q} < 0$ throughout.

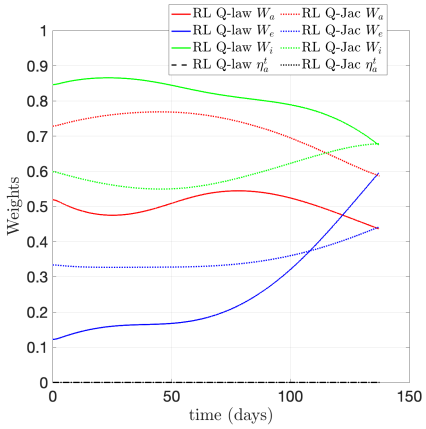
Figure 3.11 shows the evolution of η_a for both the $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$ trajectories. Eq. (2.44) shows that the absolute effectivity depends on the rate-of-change of the Lyapunov function, $\dot{Q}$. Explicitly writing this as a combination of the partial with respect to the state and the weights as $\dot{Q} = \dot{Q}_X + \dot{Q}_W$, it is clear η_a will depend on the available knowledge of $\dot{Q}$. As such, $\eta_a(\cdot)$ can be used to indicate the available knowledge used to calculate η_a.



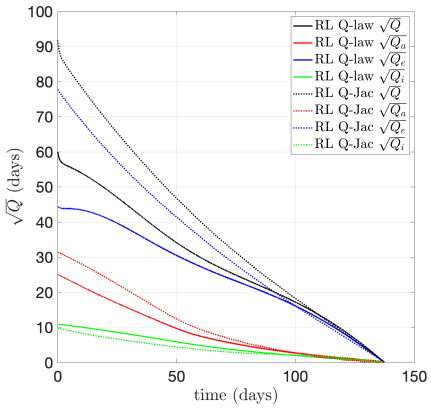
(a) Orbital Parameters from RL Q-law and RL Q-Jac.



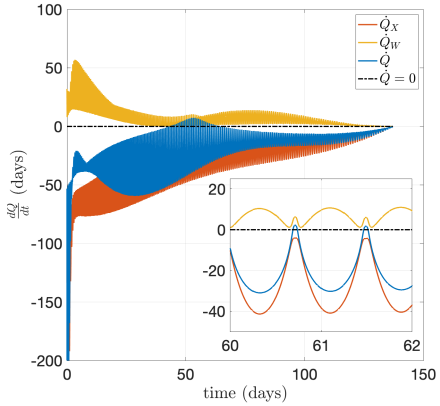
(b) Trajectory using RL Q-Jac.



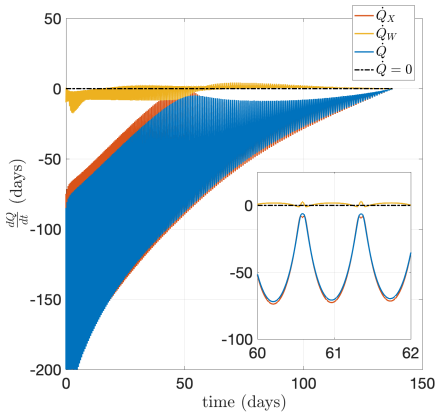
(c) Weights from RL Q-law and RL Q-Jac.



(d) Contributions to Q-law from RL Q-law and RL Q-Jac.

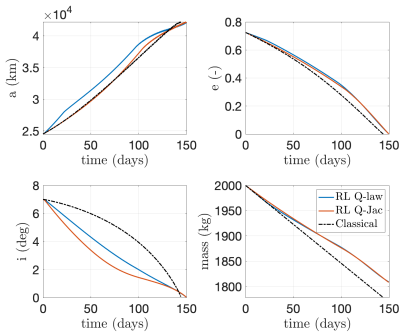


(e) $\dot{Q}$ using RL Q-law.

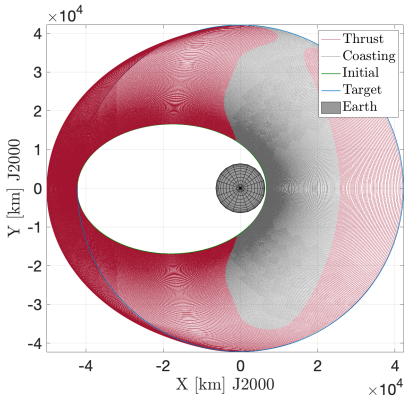


(f) $\dot{Q}$ using RL Q-Jac.

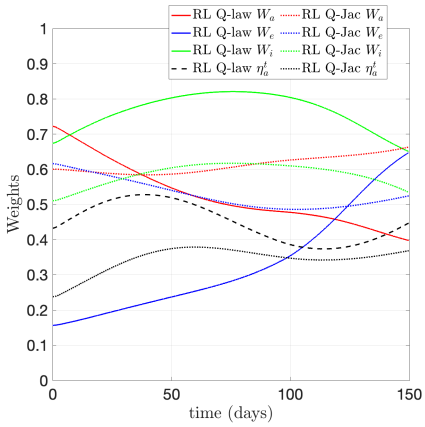
Figure 3.9: Time-optimal GTO-GEO results breakdown for RL Q-law ($\boldsymbol{u}_{\text{orig}}$) and RL Q-Jac ($\boldsymbol{u}_{\text{jac}}$) controls. In Fig. 3.9e and 3.9f, $\dot{Q}_X$ and $\dot{Q}_W$ refer to the contributions to $\dot{Q}$ from the state and weight variations respectively.



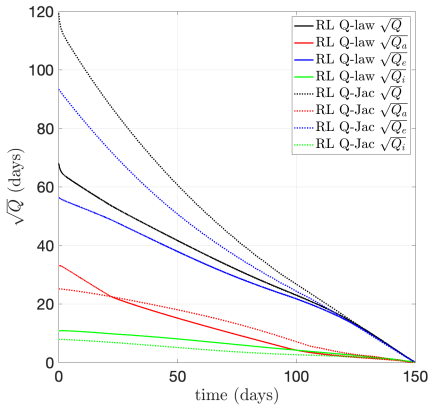
(a) Orbital Parameters from RL Q-law and RL Q-Jac.



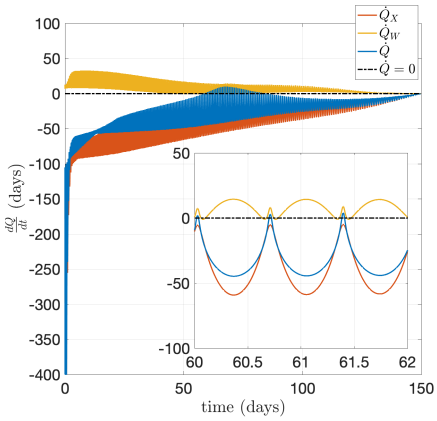
(b) Trajectory using RL Q-Jac.



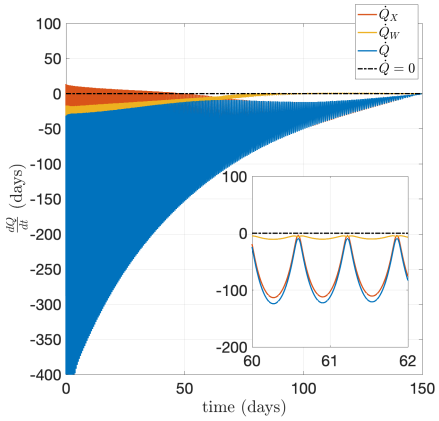
(c) Weights from RL Q-law and RL Q-Jac.



(d) Contributions to Q-law from RL Q-law and RL Q-Jac.

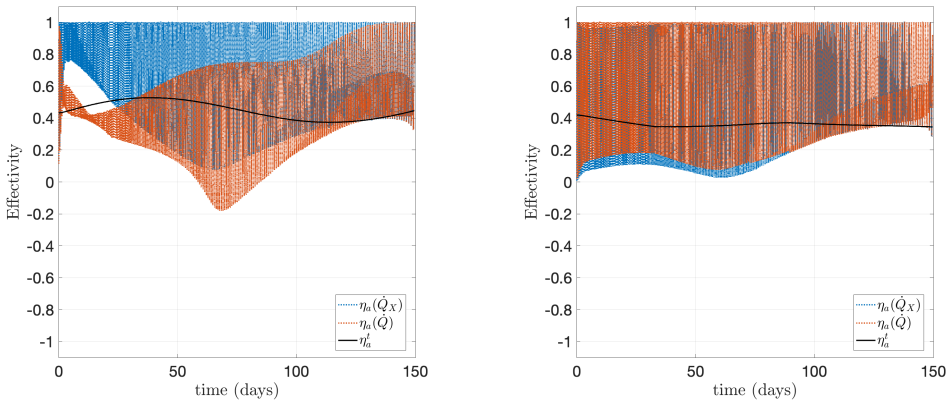


(e) $\dot{Q}$ using RL Q-law.



(f) $\dot{Q}$ using RL Q-Jac.

Figure 3.10: Mass-optimal GTO-GEO results breakdown for $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$ controls. In 3.10e and 3.10f, $\dot{Q}_X$ and $\dot{Q}_W$ refer to the contributions to $\dot{Q}$ from the state and weight variations respectively.



(a) RL Q-law (b) RL Q-Jac

Figure 3.11: Effectivity parameter η_a and threshold η_a^t for the mass-optimal GTO-GEO transfers. In both cases, the effectivity can be calculated using either the full knowledge of $\dot{Q}$ or using the more widely available but lesser knowledge of $\dot{Q}_X$.

Although conventionally $0 < \eta_a \leq 1$, this assumes η_a and the control $\boldsymbol{u}$ are computed with the full knowledge of $\dot{Q}$. This is the case for $\boldsymbol{u}_{\text{jac}}$, as seen in Fig. 3.11b, however, for $\boldsymbol{u}_{\text{orig}}$ a contribution to $\dot{Q}$ is unknown. The effectivity used during the RL Q-law transfer, and thus using $\boldsymbol{u}_{\text{orig}}$, is given by:

$$\eta_a(\dot{Q}_X) = \frac{\min_{\phi_\alpha,\phi_\beta}(\dot{Q}_X)}{\min_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}_X))}, \tag{3.24}$$

where $\dot{Q}_X$ indicates only the derivative with respect to X is used, and the contribution from the dependence on the weights W has to be neglected. Retrospectively, when accounting for the neglected contribution to $\dot{Q}$, the true $\eta_a(\dot{Q})$ appears negative in places, as seen in Fig. 3.11a. Due to this incomplete information on the most efficient places to thrust, a negative retrospective effectivity means the thrust was actually increasing the value of Q rather than decreasing it as desired. Fortunately, although $\eta_a(\dot{Q}) < 0$ for 1.2% of the transfer, this occurs exclusively when the engine is off. In fact, there is only 6.7% of the time when a more efficient direction could be computed, reducing to 5.3% when the engine is on.

3.4.2 Low Earth Orbit to Geostationary Orbit

Here a LEO-GEO transfer with an inclination change is considered in Keplerian dynamics - see Section 3.2.2 for parameters and benchmark discussion. Results are given in Table 3.7. Using $\boldsymbol{u}_{\text{orig}}$, the RL Q-law time-of-flight was 180.38 days. However, using $\boldsymbol{u}_{\text{jac}}$, the optimality increased to 179.77 days. It is also important to note that this result betters the time-dependent PSO, where the control is forced to be $\boldsymbol{u}_{\text{orig}}$ as information of $(\partial Q/\partial \boldsymbol{W})$ $(\partial \boldsymbol{W}/\partial \boldsymbol{X})$ is not available. A key advantage of the RL approach is that the Jacobian matrix can be calculated analytically from the structure of the actor network.

For the mass-optimal simulations, the absolute effectivity threshold η_a^t was used and a target time-of-flight of 200 days was set. Compared to the time-optimal classical Q-law, the best solution saves 26.77 kg (12.6%) at a cost of 1.37 days (0.7%), whilst it improves on the constant weight solution by 10.19 kg. Unlike in the GTO-GEO transfer, there is a significant improvement on the fixed PSO results for both time and mass optimal transfers. This clearly demonstrates the advantage of varying the parameters throughout the transfer. Here the RL Q-Jac not only enforces stability but significantly increases the optimality of the mass-optimal solution. This can be explained by the effectivity below.

Figures 3.12 and 3.13 shows the RL Q-law and RL Q-Jac results for the time-optimal and mass-optimal simulations. Again the plots of $\dot{Q}$ indicate a lack of Lyapunov stability in the RL Q-law case, but this is resolved by adding the Jacobian of the state-dependent weights. Intuition on the time-histories observed in the RL Q-law and RL Q-Jac weights can be drawn from the PSO spline approach.

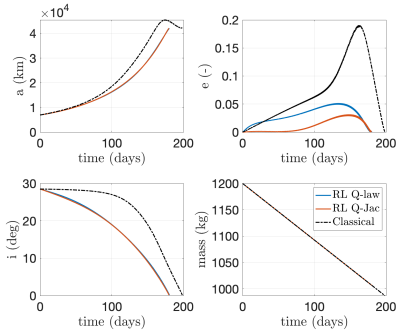
Table 3.7: Comparison of state-dependent Q-law performance for LEO-GEO transfer.

	Method	Time (days)	Propellant (kg)
	Time-optimal Classical	198.32	212.70
Time	Ref. [58]	198.99	-
	PSO fixed	186.10	199.58
	PSO spline	180.68	193.77
	RL Q-law (RBF)	180.38	193.45
	RL Q-Jac (RBF)	179.77	192.80
	RL Q-law (TRI)	181.07	194.20
	RL Q-Jac (TRI)	180.46	193.54
Mass	PSO fixed	199.67	196.52
	PSO spline	199.74	189.43
	RL Q-law (RBF)	198.81	192.10
	RL Q-Jac (RBF)	199.69	185.93
	RL Q-law (TRI)	199.74	190.87
	RL Q-Jac (TRI)	199.59	186.81

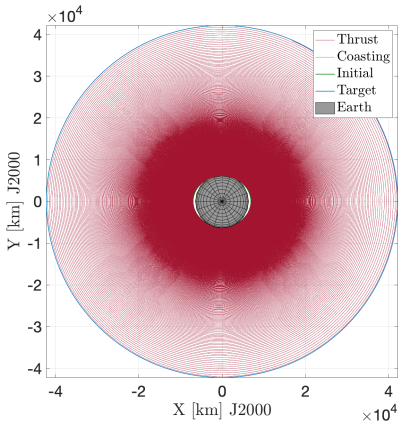
Figure 3.14 shows the evolution of η_a for both the $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$ trajectories. Again Eq. (2.44) is used in both cases, but only with the available knowledge of $\dot{Q}$, and, as such, the RL Q-law is using Eq. (3.24). Unlike in the GTO-GEO case where this had minimal effect, here it can explain the discrepancy between the RL Q-law and RL Q-Jac solutions. In Fig. 3.14a, $\eta_a(\dot{Q}) < 0$ occurs 12.0% of the time, and 11.8% when the engine is on, suggesting thrusting here is actually ineffective. In addition, there is a more efficient control direction for 20.9% of the transfer, reducing to 16.0% when the engine is on.

3.5 Discussion and Summary

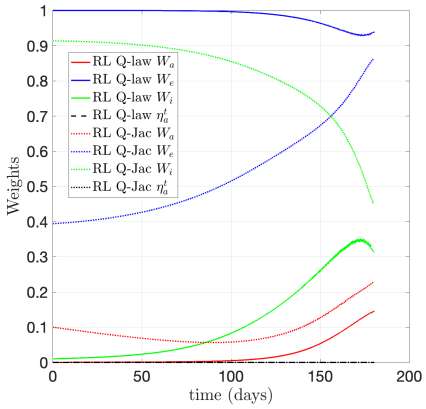
In this chapter, a novel Reinforced Lyapunov Controller framework was outlined. The main purpose was to outline the theoretical background needed to understand the im-



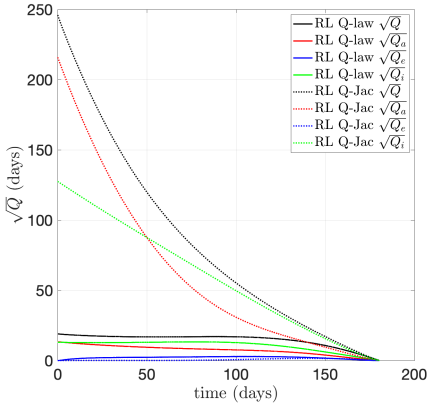
(a) Orbital Parameters from RL Q-law and RL Q-Jac.



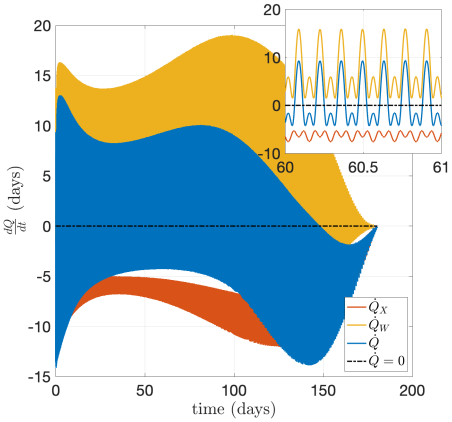
(b) Trajectory using RL Q-Jac.



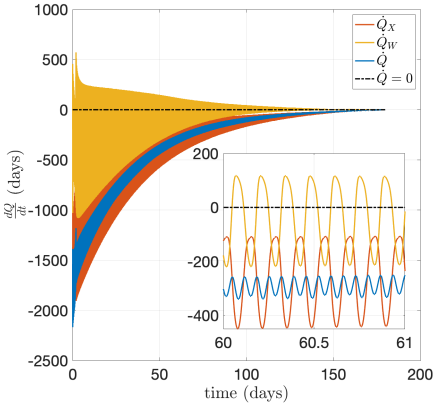
(c) Weights from RL Q-law and RL Q-Jac.



(d) Contributions to Q-law from RL Q-law and RL Q-Jac.

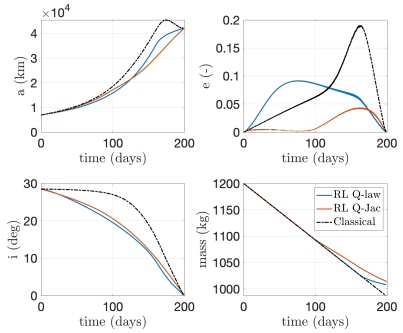


(e) $\dot{Q}$ using RL Q-law.

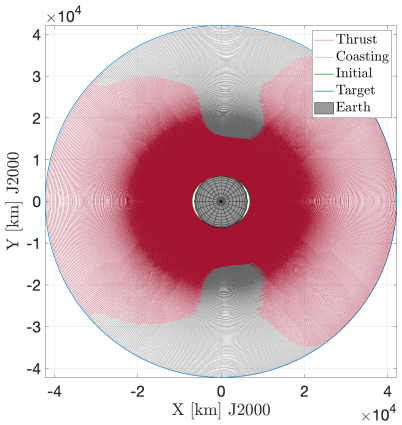


(f) $\dot{Q}$ using RL Q-Jac.

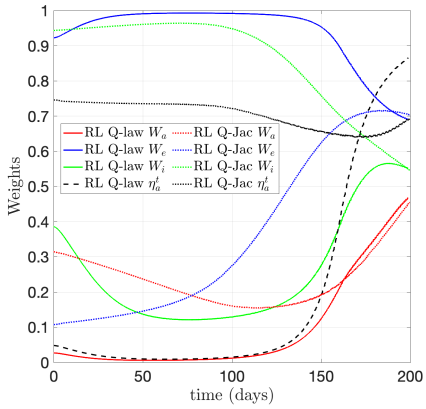
Figure 3.12: Time-optimal LEO-GEO results breakdown for $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$ controls. In 3.12e and 3.12f, $\dot{Q}_X$ and $\dot{Q}_W$ refer to the contributions to $\dot{Q}$ from the state and weight variations respectively.



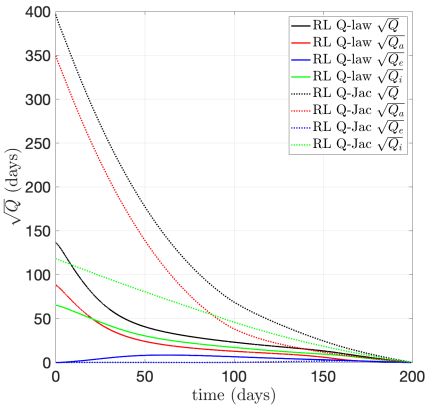
(a) Orbital Parameters from RL Q-law and RL Q-Jac.



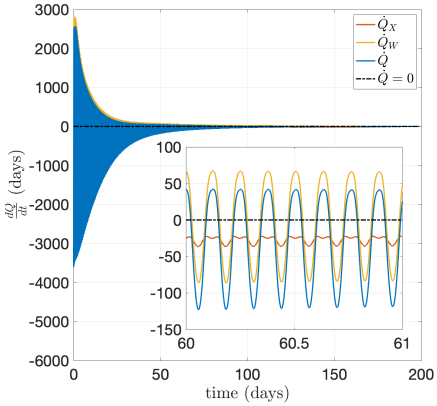
(b) Trajectory using RL Q-Jac.



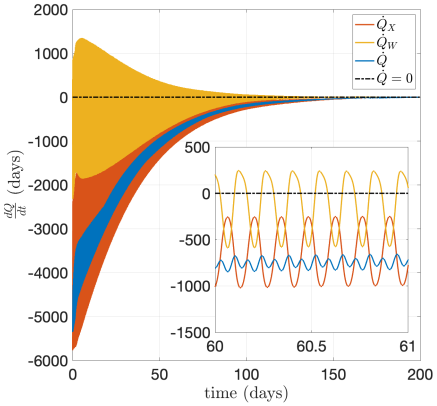
(c) Weights from RL Q-law and RL Q-Jac.



(d) Contributions to Q-law from RL Q-law and RL Q-Jac.

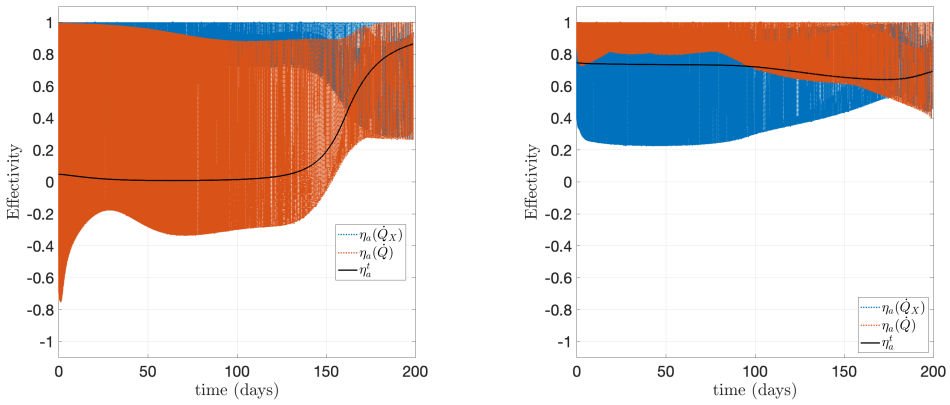


(e) $\dot{Q}$ using RL Q-law.



(f) $\dot{Q}$ using RL Q-Jac.

Figure 3.13: Mass-optimal LEO-GEO results breakdown for $\boldsymbol{u}_{\text{orig}}$ and $\boldsymbol{u}_{\text{jac}}$ controls. In 3.13e and 3.13f, $\dot{Q}_X$ and $\dot{Q}_W$ refer to the contributions to $\dot{Q}$ from the state and weight variations respectively.



(a) RL Q-law

(b) RL Q-Jac

Figure 3.14: Effectivity parameter η_a and threshold η_a^t for the mass-optimal LEO-GEO transfers. In both cases, the effectivity can be calculated using either the full knowledge of $\dot{Q}$ or using the more widely available but lesser knowledge of $\dot{Q}_X$.

plementations found in later chapters. The main contributions can be divided into three parts: the state-dependent philosophy, the RL framework, and ensuring Lyapunov stability. These lay the foundations for developing a lightweight and closed-loop control law that can be used for both initial trajectory design and on-board guidance.

The proposed approach combines Lyapunov control theory with RL techniques. The advantage of combining these two approaches allows one to eliminate the drawbacks from each: namely the sub-optimality of Lyapunov controllers and the unknown stability of RL methods. Time-dependent variations of the user-defined weights in Lyapunov control laws are used to demonstrate a proof-of-concept for the state-dependent approach, and the limitations associated with a purely time-dependent approach are discussed. The state-dependent approach requires a continuous state-action PPO RL algorithm implementation. The fundamental RL architecture that is used to learn the mapping from state to action is discussed. A state-dependent controller is presented, which enforces stability without compromising optimality through the Jacobian of the state-dependent parameters. The necessary Jacobian is available analytically through an actor network, ensuring the system remains closed-loop.

Investigations for transfers from a GTO-GEO and LEO-GEO show the control can ensure stability at each state in the transfer. Making the parameters state-dependent significantly increases the optimality of the solutions found for both mass-optimal transfers and the LEO-GEO time-optimal transfer. Note that this approach is introduced using the Lyapunov-based Q-law approach, as introduced in Chapter 2, however, it extends to other Lyapunov functions.

To the author's knowledge the addition of stability through the Jacobian term has not been explored elsewhere, the importance of which can extend beyond trajectory design and more generally to the optimisation of non-linear controllers.

Chapter 4
Trajectory Design in the presence of Perturbations

In this chapter the techniques developed and tested in Chapter 3 will be extended and investigated in the presence of perturbing accelerations. The main purpose of this chapter is to understand how the Reinforced Lyapunov Controller performs in an environment it does not have full knowledge of. Two approaches are presented: the first demonstrates the robust and closed-loop nature of the Reinforced Lyapunov Controller by deploying it in the presence of perturbations. The second investigates the potential to train the Reinforced Lyapunov Controller in the presence of these perturbations via a novel cone-clock angle approach. Whilst the work presented in this chapter has not yet been published, the cone-clock approach was presented by the authors in [43].

4.1 Introduction

Working in a Keplerian and deterministic environment enables the development and perfection of trajectory design and guidance techniques, and is often the starting point for most approaches. However, further work needs to be done to better replicate the *real-world* environment, where the dynamics are more complicated and full knowledge of the system is not possible. A good approach in Keplerian dynamics is not always guaranteed to achieve good performance, or even converge, in more complex dynamics. The task of an astrodynamicist involves ensuring, to a degree of confidence, that their approach takes appropriate account of the environment it is designed for, and to ensure that any mission analyst, spacecraft operator or future user is aware of the performance and limitations in these environments.

In this chapter Lyapunov control laws are combined with the state-dependence of RL to design preliminary low-thrust transfers in the presence of perturbations. The possibility of training a RL agent in the presence of perturbations is explored. These include J_2, the Sun and Moon's third-body gravity and eclipse effects. This will enable the RL agent to explore the effects of these perturbing forces during training and learn to exploit their existence when designing low-thrust transfers.

The aim is to leverage the state-dependence offered by the RL architecture to allow the Lyapunov control law to evolve as the perturbing dynamics evolve throughout the transfer. A cone-clock angle approach is proposed to free the control direction, allowing

greater freedom in the determined control direction whilst maintaining Lyapunov stability. In this chapter, GTO-GEO and LEO-GEO transfers are chosen as the benchmark cases to begin with. These are well documented in the literature and as such allow easy comparison with existing techniques. In addition, due to the many-revolutions, they are particularly challenging for conventional indirect and direct methods. However, perhaps most importantly, these transfers should highlight two desired aspects of the Reinforced Lyapunov Controller. Firstly, there is sufficient difference in initial and final state to allow an evolution throughout the transfer and enable different controller behaviour to be investigated. Similarly, there is an increased possibility that the dominant perturbing dynamics can differ through the transfer, increasing the potential importance of the state-dependent approach considered here.

As becomes apparent, the effect of the perturbations is not always clear on the transfers. As such, a new transfer scenario is proposed: a LEO SSO-SSO transfer. In this scenario, the spacecraft is tasked with changing its RAAN by 10°, a costly manoeuvre, especially in Keplerian dynamics. However, in reality the non-spherical nature of Earth results in spherical harmonics in the gravitational field. The most dominant is the J_2 perturbation, which, as seen in Section 2.1.4, causes a secular drift in both RAAN and argument of periapsis. As such, it is possible for a spacecraft to exploit this drift in order to change its RAAN, potentially at a different rate to the target RAAN. This, along with the work presented in Chapter 7, explores the controller behaviour in environments more greatly affected by the perturbing accelerations and this allows increased understanding of this aspect of trajectory design.

This chapter is structured as follows. In Section 4.2 the approach and pseudocode for training in the presence of perturbations and eclipse events is presented. Results for GTO-GEO and LEO-GEO are presented and discussed, including J_2, third-body gravity affects, and eclipse effects. This is all done with the osculating expressions for the perturbing accelerations. In Section 4.3 a novel cone-clock angle approach is introduced, attempting to give the control law greater freedom and exploit the existence of the perturbations. Following this, an alternative transfer scenario between two LEO SSOs is considered, where the dominance of the J_2 perturbation strongly influences the possible transfers. Here only the secular contribution from J_2 is considered. A summarising discussion with the conclusions is given at the end.

4.2 Performance in Perturbed Dynamics

The results for the Reinforced Lyapunov Controller in the presence of perturbations is presented. To begin with these are simulated for the same GTO-GEO and LEO-GEO transfers presented in Chapter 3. The results for the Q-law are presented and discussed here, whilst results for a basic Lyapunov controller are given in Appendix A.2. This is to emphasise the approach is flexible to the controller adopted, and can improve the

performance in both cases. However, it must also be noted the underlying controller can limit the optimality of the results, and this choice of controller should be taken into account.

The motivation of this section is three-fold:

- Can an RL agent trained without perturbations remain robust in the presence of perturbations?
- Can the performance of the RL agent be further improved by including perturbations in the training?
- Is the approach able to learn something about the existence of and exploit the perturbations?

In order to address these, two different simulations are compared for each test case. Firstly, the Reinforced Lyapunov Controller trained without the presence of perturbations (as done in Chapter 3) is deployed in the presence of perturbations. In this case the training had no experience of the perturbing accelerations, and hence the controller reacts purely in a feedback manner. This can emphasise the importance of the closed-loop nature of the approach and allows discussion on the suitability of on-board implementation, as no controller can be made aware *a priori* of all the perturbations it might experience. Secondly, the Reinforced Lyapunov Controller will be re-trained in the presence of the perturbations. This allows learning process to experience and potentially adjust its behaviour according the presence of dynamics which, fundamentally, the Lyapunov controller does not have knowledge of. Importantly, information of the perturbation's existence is not provided to the controller *a priori*. Instead, the perturbation's presence must be inferred indirectly through interacting with the environment.

A stable control $\boldsymbol{u}$ in the $R\theta h$ frame is conventionally chosen to minimise rate of change of the Lyapunov function Q. By incorporating the state-weight Jacobian as in Chapter 3, the rate of change is given by:

$$\begin{aligned} \dot{Q} = \dot{Q}_{\boldsymbol{X}} + \dot{Q}_{\boldsymbol{W}} = \frac{\partial Q}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\dot{\boldsymbol{W}} = \left(\frac{\partial Q}{\partial \boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}\right)\dot{\boldsymbol{X}}, \\ \dot{Q} = \boldsymbol{M}\dot{\boldsymbol{X}} \quad \text{where} \quad \boldsymbol{M} = \left(\frac{\partial Q}{\partial \boldsymbol{X}} + \frac{\partial Q}{\partial \boldsymbol{W}}\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{X}}\right). \end{aligned} \tag{4.1}$$

Under a disturbing acceleration $\boldsymbol{a_d}$, $\dot{\boldsymbol{X}} = \boldsymbol{B}\boldsymbol{a_d}$ and hence $\dot{Q} = \boldsymbol{M}\boldsymbol{B}\boldsymbol{a_d}$. It is possible to define a vector $\hat{\boldsymbol{p}}$ as the direction to maximise the rate-of-change of Q:

$$\boldsymbol{p} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{M}^{\mathrm{T}} \qquad \text{and} \qquad \hat{\boldsymbol{p}} = \frac{\boldsymbol{p}}{\|\boldsymbol{p}\|}. \tag{4.2}$$

If it is assumed the disturbing acceleration is all due to a control acceleration ($\boldsymbol{a_d} = \boldsymbol{u}$) then a control in the opposite direction of $\boldsymbol{p}$, $-f\hat{\boldsymbol{p}}$, will minimise the rate-of-change of the

Algorithm 2 Pseudocode: Training with and without Perturbations

Set initial state $\boldsymbol{X}_0$ and target state $\boldsymbol{X}_T$ ▷ Initialise Transfer
Set random parameters θ_k ▷ Initialise RL
while Training **do**
  for Batch N + 2 **do** ▷ N Stochastic $\boldsymbol{W}$ and 2 Deterministic $\boldsymbol{W}$
    while Propagating Transfer **do** ▷ Computing transfer trajectory
      if Training with Perturbations **then**
        $\boldsymbol{a_T} = \boldsymbol{a_p}$ ▷ Use perturbed dynamics
      else if Training without Perturbations **then**
        $\boldsymbol{a_T} = \boldsymbol{0}$ ▷ Use Keplerian dynamics
      $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}, \theta_k)$ ▷ Full info on $\partial \boldsymbol{W}/\partial \boldsymbol{X}$ assumed
      $\boldsymbol{u} \leftarrow \text{Lyapunov}(\boldsymbol{X}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0}))$
      $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{a_T})$
  Update θ_k if $C(\theta) < C(\theta_k)$ ▷ Deterministic Update
while Deploying **do** ▷ Deploy learnt policy
  while Propagating Transfer **do** ▷ Computing transfer trajectory
    $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}, \theta_k)$ ▷ Full info on $\partial \boldsymbol{W}/\partial \boldsymbol{X}$ assumed
    $\boldsymbol{u} \leftarrow \text{Lyapunov}(\boldsymbol{X}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0}))$
    $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{a_p})$ ▷ Use perturbed dynamics

Lyapunov function (make as negative as possible):

$$\dot{Q} = -f\boldsymbol{p}^T\hat{\boldsymbol{p}}. \tag{4.3}$$

Making the weights $\boldsymbol{W}$ state-dependent will make $-\hat{\boldsymbol{p}}$ more optimal, as shown in Chapter 3. However, on its own this RL Q-Jac approach is not guaranteed to be stable in the presence of a perturbing acceleration $\boldsymbol{a_p}$. Reconsidering Eq. (2.4), when the two-body system subject to a disturbing acceleration $\boldsymbol{a}_d$ is made up from the control $\boldsymbol{u}$ and perturbation $\boldsymbol{a}_p$, one can write this as:

$$\dot{\boldsymbol{X}} = \boldsymbol{B}\left(\boldsymbol{u} + \boldsymbol{a}_p\right). \tag{4.4}$$

Note the Lyapunov function Q is a function of both $\boldsymbol{X}$ and $\boldsymbol{X}_T$ from Eq. (2.42). Neglecting the evolution of the target orbit ($\dot{\boldsymbol{X}}_T = 0$), from $\dot{Q} = \boldsymbol{MBa_d} = \boldsymbol{MB}\left(\boldsymbol{u} + \boldsymbol{a}_p\right)$ it follows:

$$\dot{Q} = \boldsymbol{p}^T\left(\boldsymbol{u} + \boldsymbol{a}_p\right), \tag{4.5}$$

and for Lyapunov stability one needs

$$\boldsymbol{p}^T\left(\boldsymbol{u} + \boldsymbol{a}_p\right) < 0. \tag{4.6}$$

The standard procedure to handle this is to rederive $\hat{\boldsymbol{p}}$ with an analytical expression for $\boldsymbol{a}_p$. However, for Lyapunov stability, the best choice of control direction is $-\hat{\boldsymbol{p}}$. Here the

possibility of learning this necessary compensation via RL is explored. Hence, the aim is to investigate whether it is possible to provide an optimal, and Lyapunov stable, control direction without prior knowledge of the analytical expression for the perturbations. The aim is to leverage the state-dependence offered by the RL architecture to allow the Lyapunov control law to evolve as the dynamics evolve throughout the transfer.

Algorithm 2 shows how the training is done both with and without the perturbing accelerations. $f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{a_p})$ refers to the dynamics of state $\boldsymbol{X}$ subject to a spacecraft control $\boldsymbol{u}$ and perturbing acceleration $\boldsymbol{a_p}$ - see Section 2.1.4, specifically Eqs. (2.14), (2.24) and (2.25).

4.2.1 Results

Tables 4.1 and 4.2 show the results for the time- and mass-optimal Q-law simulations for a GTO-GEO transfer - see Section 3.2.1. Throughout this Chapter an epoch of 1$^{\text{st}}$ January 2022 12:00:00 UTC was used. Although the Reinforced Lyapunov Controllers are deployed in the presence of perturbations, comparisons are shown for those trained without and with the perturbations. Only one perturbation was applied within each simulation for simplicity, although it is possible to combine them. Time-of-flight, propellant mass and ΔV are shown. In addition, the *Fraction* $\dot{V} < 0$ indicates the fraction of the whole transfer during which $\boldsymbol{a}_p$ is aiding the decrease of the Lyapunov function. In other words, $\boldsymbol{p}^T \boldsymbol{a}_p < 0$. The purpose of this metric is to understand whether training with the perturbations changes how the controller behaves.

Table 4.1: Comparison of time-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Grey highlights indicate the RL Q-Jac solution trained with Perturbations.

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	144.03	222.06	2.308	-
	PSO Spline	-	-	-	-	137.16	211.47	2.192	-
	RL Q-law	-	-	-	-	137.29	211.67	2.194	-
	RL Q-Jac	-	-	-	-	137.70	212.31	2.201	-
J_2	Classical	-	-	-	-	143.93	221.91	2.307	-
	PSO Spline	137.42	211.88	2.196	-	137.44	211.91	2.197	-
	RL Q-law	137.43	211.90	2.197	0.53	137.64	212.21	2.200	0.53
	RL Q-Jac	137.84	212.53	2.204	0.55	137.98	212.75	2.206	0.54
3^{rd}-body	Classical	-	-	-	-	144.19	222.31	2.311	-
	PSO Spline	137.35	211.77	2.195	-	137.67	212.27	2.201	-
	RL Q-law	137.41	211.86	2.196	0.50	137.81	212.47	2.203	0.49
	RL Q-Jac	138.04	212.82	2.207	0.49	137.97	212.73	2.206	0.49
Eclipse	Classical	-	-	-	-	151.37	224.26	2.333	-
	PSO Spline	138.18	210.87	2.185	-	138.48	211.31	2.190	-
	RL Q-law	138.16	210.83	2.185	-	138.59	211.44	2.192	-
	RL Q-Jac	139.19	212.47	2.203	-	138.82	211.77	2.195	-

In addition to the GTO-GEO transfer above, the LEO-GEO transfer scenario was also considered - see Section 3.2.2. Table 4.3 shows the results for the time-optimal simulations for both the Q-law controller, whilst Table 4.4 gives the mass-optimal results.

The first key message is all the RL controllers trained in Keplerian dynamics but deployed in perturbed dynamics converge to the target orbits. This can be seen by com-

Table 4.2: Comparison of mass-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Here a maximum allowed time-of-flight of 150 days is desired. Grey highlights indicate the RL Q-Jac solution trained with Perturbations.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	-	144.03	222.06	2.308	-	-
	PSO Spline	-	-	-	-	-	149.75	190.05	1.958	190.05	-
	RL Q-law	-	-	-	-	-	149.82	190.78	1.965	190.78	-
	RL Q-Jac	-	-	-	-	-	149.91	191.88	1.978	192.13	-
J_2	Classical	-	-	-	-	-	143.93	221.91	2.307	-	-
	PSO Spline	149.76	190.74	1.966	190.74	-	149.36	191.32	1.966	191.32	-
	RL Q-law	150.23	191.85	1.970	191.85	0.54	149.32	192.51	1.985	192.51	0.54
	RL Q-Jac	150.81	192.32	1.983	193.95	0.55	149.74	193.09	1.991	193.09	0.55
3^{rd}-body	Classical	-	-	-	-	-	144.19	222.31	2.311	-	-
	PSO Spline	149.76	191.04	1.969	191.04	-	150.06	190.27	1.961	190.75	-
	RL Q-law	149.72	191.27	1.972	191.27	0.49	150.10	191.35	1.967	191.35	0.48
	RL Q-Jac	149.82	192.75	1.988	192.85	0.49	150.18	192.05	1.980	192.72	0.48
Eclipse	Classical	-	-	-	-	-	151.37	224.26	2.333	-	-
	PSO Spline	149.77	190.31	1.961	190.32	-	150.61	189.41	1.951	190.73	-
	RL Q-law	149.81	190.98	1.967	190.98	-	150.40	191.30	1.961	191.30	-
	RL Q-Jac	149.75	192.61	1.986	192.61	-	150.50	191.56	1.975	192.72	-

Table 4.3: Comparison of time-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Grey highlights indicate the RL Q-Jac solution trained with Perturbations.

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	198.30	212.68	6.313	-
	PSO Spline	-	-	-	-	182.50	195.73	5.762	-
	RL Q-law	-	-	-	-	181.90	195.08	5.741	-
	RL Q-Jac	-	-	-	-	180.44	193.52	5.691	-
J_2	Classical	-	-	-	-	197.22	211.51	6.275	-
	PSO Spline	183.64	196.95	5.802	-	185.25	198.68	5.858	-
	RL Q-law	186.71	200.24	5.908	0.51	184.43	197.79	5.829	0.51
	RL Q-Jac	180.39	193.46	5.689	0.50	181.59	194.75	5.731	0.48
3^{rd}-body	Classical	-	-	-	-	198.25	212.62	6.311	-
	PSO Spline	182.65	195.89	5.768	-	182.47	195.70	5.761	-
	RL Q-law	181.95	195.13	5.743	0.49	181.89	195.07	5.741	0.49
	RL Q-Jac	180.05	193.10	5.678	0.49	180.45	193.53	5.691	0.49
Eclipse	Classical	-	-	-	-	227.64	205.20	6.069	-
	PSO Spline	216.81	192.88	5.671	-	221.32	196.70	5.794	-
	RL Q-law	219.76	194.04	5.708	-	222.31	198.14	5.840	-
	RL Q-Jac	217.46	193.57	5.693	-	225.81	200.27	5.909	-

Table 4.4: Comparison of mass-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Here a maximum allowed time-of-flight of 200 days is desired. Grey highlights indicate the RL Q-Jac solution trained with Perturbations.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	-	198.3	212.68	6.313	-	-
	PSO Spline	-	-	-	-	-	199.20	188.92	5.544	188.92	-
	RL Q-law	-	-	-	-	-	199.81	190.85	5.603	190.85	-
	RL Q-Jac	-	-	-	-	-	199.75	186.97	5.481	186.97	-
J_2	Classical	-	-	-	-	-	197.22	211.51	6.275	-	-
	PSO Spline	199.75	190.09	5.581	190.09	-	210.56	190.63	5.599	202.22	-
	RL Q-law	199.70	190.87	5.606	190.87	0.50	200.19	191.49	5.611	191.49	0.49
	RL Q-Jac	199.75	190.78	5.603	190.78	0.51	201.87	188.61	5.534	190.89	0.49
3^{rd}-body	Classical	-	-	-	-	-	198.25	212.62	6.311	-	-
	PSO Spline	199.74	188.99	5.546	188.99	-	199.07	188.89	5.543	188.89	-
	RL Q-law	199.39	188.48	5.530	188.48	0.50	199.70	190.74	5.602	190.74	0.50
	RL Q-Jac	199.49	186.80	5.476	186.80	0.50	199.69	186.94	5.480	186.94	0.50
Eclipse	Classical	-	-	-	-	-	227.64	205.20	6.069	-	-
	PSO Spline	217.85	191.37	5.622	210.78	-	248.69	190.59	5.597	243.07	-
	RL Q-law	219.36	192.04	5.643	213.08	-	239.89	186.57	5.469	229.58	-
	RL Q-Jac	215.95	191.74	5.634	209.11	-	260.20	185.63	5.439	250.46	-

paring the simulations in the *Trained without Perturbations* column. In each case the controller is trained in Keplerian dynamics, with the *Keplerian* row demonstrating the

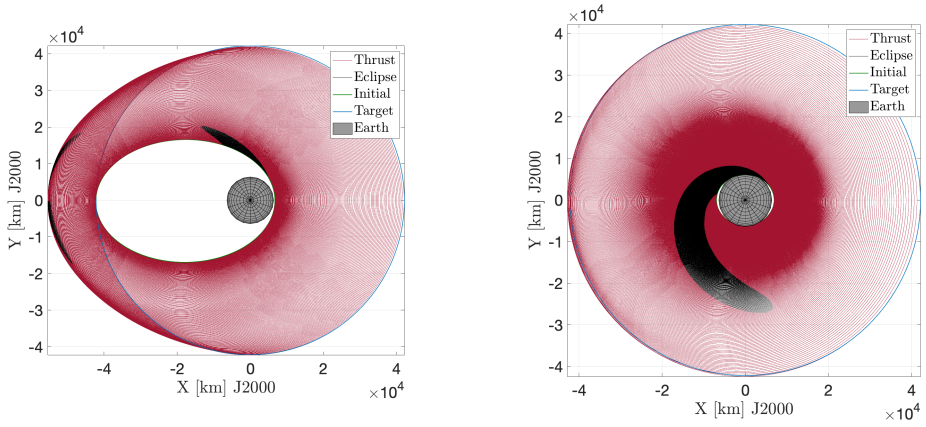
performance when deployed in Keplerian dynamics. However, it can also be deployed in J_2, 3^{rd}*-body* and *Eclipse* environments, where the fully trained network now experiences an environment it was not trained in. Breaking down the effects of the J_2 and 3^{rd}*-body* perturbations, it can be seen the controllers suffer minimal impact on their optimality. For the GTO-GEO transfer with J_2 perturbation, the RL Q-law controller looses 0.35 days and the RL Q-Jac 0.28 days for the time-optimal simulations, whilst they lose 1.73 kg and 1.21 kg in the mass-optimal simulations respectfully. In the LEO-GEO case with the J_2 perturbation, the RL Q-law looses 2.53 days and the RL Q-Jac 1.15 days for the time-optimal simulations, whilst they lose 0.64 kg and 1.64 kg in the mass-optimal simulations respectfully. In all except the last case, the addition of the Jacobian term means the controller is less impacted by the J_2 perturbation. As the *Trained without Perturbations* column shows, for the last case to remain within the desired time-of-flight of < 200 days even more propellant mass must be consumed.

For the 3^{rd}-body case, the RL Q-law looses 0.52 days and the RL Q-Jac 0.27 days for the time-optimal simulations, whilst they lose 0.57 kg and 0.17 kg in the mass-optimal simulations respectfully. In the LEO-GEO case with the 3^{rd}-body perturbation, the both controller actually gain a negligible 0.01 days in the time-optimal simulations, whilst they gain 0.11 kg and 0.03 kg in the mass-optimal simulations respectfully. Its observed that the 3^{rd}-body perturbation has less impact on the transfers than J_2 perturbation, an expected outcome given the difference in strength of these perturbations. Only in the LEO-GEO transfer does the 3^{rd}-body perturbation improve the performance of the controllers, but by an almost negligible amount. Overall it appears the addition of the Jacobian makes the Reinforced Lyapunov Controller less impacted by the perturbations, both positively and negatively. However, as the *Fraction* $\dot{V} < 0$ indicates, in the RL simulations considered thus far, there appears little change in the underlying behaviour which specifically increases the contribution of the perturbing acceleration to the decrease of the Lyapunov function.

One major test for a low-thrust trajectory design algorithm is its ability to handle discontinuities such as eclipse events, which can be very challenging for conventional trajectory design techniques. As mentioned earlier, the existence of the eclipse event is not communicated *a priori* to the agent. Instead, the eclipse event is only computed inside the propagator and results in an inability to fire the engine. Clearly this affects the optimality of the results for the controller trained without eclipse events. The extend to which it impacts the results will depend on both the epoch and the transfer scenario. LEO-GEO transfers spend more time in LEO, where eclipse events are naturally longer given the proximity to Earth. This is reflected in the increased time-of-flight from 180.44 days to 225.81 days for the RL Q-Jac for the LEO-GEO transfer, a 45.37 day penalty, as opposed to just 1.12 days for the GTO-GEO transfer. The epoch affects the transfer because it dictates where the eclipse event occurs in the orbital revolution. For transfers looking to increase their semi-major axis considerably (as is the case here), periapsis is

the key location to thrust. Hence, any eclipse events occurring near the periapsis will hamper the spacecraft in efficiently increasing its orbital semi-major axis.

There is potential to improve this further by providing information on eclipse events to the actor network during training. For example, by adding the locations of eclipse events as inputs to the actor network. This is not done currently, but could facilitate improved learning. The current performance is achieved by the model-free RL algorithm experiencing the environment.



(a) GTO-GEO (b) LEO-GEO

Figure 4.1: Time-optimal transfer trajectories for the RL Q-Jac controller in the presence of eclipse events.

Following the deployment of the controllers trained without perturbations, the possibility of training them with perturbations was considered. By considering the results across Tables 4.1, 4.2, 4.3, and 4.4, one can assess the impact training with perturbations has on the performance of the Reinforced Lyapunov Controllers. Across time- and mass-optimal transfers for both GTO-GEO and LEO-GEO, there are 12 simulation scenarios for each Reinforced Lyapunov Controller. Ten of the base RL Q-law transfers were improved, whilst eight of the RL Q-Jac transfers where improved. However, the margin for improvement was minimal.

There are two suggestions as to why the overall results are not unanimously improved. First, it is possible that a perturbing acceleration complicates the learning process itself. As the underlying control law has no knowledge of the acceleration, it is difficult to distinguish a particular action as a bad action from the agent, or the result of the perturbing acceleration. Given improvements elsewhere in the transfer, these two can become uncoupled on the subsequent learning iteration and as such the agent might experience a different perturbation on the subsequent iteration. Secondly, in the case of eclipse events in particular, it is possible the perturbation makes the path towards the optimal solution much harder to follow. Given the potential for multiple local minima this can result in comparable results with and without perturbations.

This demonstrates the closed-loop nature of the approach, and is one of the advan-

tages of using a model-free RL framework. It is worth remembering that the underlying controller, in this case the Q-law, does not have knowledge of the perturbing accelerations. Due to the closed-loop nature of both the Lyapunov control laws and the actor network configuration, the controller is able to experience these perturbations in a feedback manner, and in all simulated scenarios, was able to converge to the target orbit with minimal impact to the objective, both time-of-flight and propellant mass. As the magnitude of the perturbations is minor, particularly towards the end of the transfer, convergence to the target orbit is not hindered significantly. The most noticeable difference comes in the eclipse simulations, where the low-thrust energy (assumed to be a solar electric propulsion unit) is switched off. As such, it a deterioration in time-of-flight is expected. No failures demonstrate the closed-loop feedback-driven nature of the control laws is maintained, something which is not possible when considering time-dependent weights in Lyapunov control laws. For the J_2 scenario, the addition of the Jacobian proves to be more optimal than RL Q-law formulation. In the 3^{rd}-body case there was no clear distinction, whilst the same is true for the eclipse scenarios.

4.3 Cone-clock approach

Following on from the previous section, it is of interest to further extend this approach exploit the existence of the perturbing accelerations. In this section, an approach for freeing the control direction and allowing increased exploration whilst ensuring Lyapunov stability is proposed. It is noted here that whilst this approach is developed in the presence of perturbing accelerations, it is not restricted to such environments and can be used even when the Lyapunov function has full knowledge of the dynamics.

A new cone-clock angle approach is introduced, adding two additional angular variables to free the control direction. These variables are learnt by the RL algorithm, in addition to the weights $\boldsymbol{W}$, and ensure the resulting control direction has increased freedom whilst crucially remaining stable.

Recall that, whilst designing $-\hat{\boldsymbol{p}}$ one does so in a way that remains Lyapunov stable in two-body dynamics (i.e., $\dot{Q} < 0$), but also one which minimises $\dot{Q}$. As such, any control $\boldsymbol{u}$ in the positive hemisphere projecting onto $-\hat{\boldsymbol{p}}$ must also result in $\dot{Q} < 0$ when $\boldsymbol{a}_p = \boldsymbol{0}$ and is, therefore, also stable - see Eq. (4.5). Defining $\alpha \in [0, \pi]$ as the half-cone angle such that $\boldsymbol{u} \cdot -\boldsymbol{p} = \|\boldsymbol{u}\| \, \|\boldsymbol{p}\| \cos\alpha$ and $\beta \in [0, 2\pi)$ as the clock angle, the control $(\boldsymbol{u})^p$ is given as:

$$(\boldsymbol{u})^p = f \begin{bmatrix} \cos\alpha \\ \sin\alpha\cos\beta \\ \sin\alpha\sin\beta \end{bmatrix}, \tag{4.7}$$

where the frame is defined by the vector $\hat{\boldsymbol{e}}_{p1} = -\hat{\boldsymbol{p}}$, and two orthogonal directions $\hat{\boldsymbol{e}}_{p2} = -\hat{\boldsymbol{p}} \times \hat{\boldsymbol{h}}$ and $\hat{\boldsymbol{e}}_{p3} = -\hat{\boldsymbol{p}} \times (-\hat{\boldsymbol{p}} \times \hat{\boldsymbol{h}})$. Here $\hat{\boldsymbol{h}}$ is the unit vector for the angular momentum of the

orbit given by $\hat{\boldsymbol{h}} = \hat{\boldsymbol{r}} \times \hat{\boldsymbol{v}}$. With a rotation matrix the control vector in the $R\theta h$ frame is:

$$\boldsymbol{u} = f \begin{bmatrix} | & | & | \\ \hat{\boldsymbol{e}}_{p1} & \hat{\boldsymbol{e}}_{p2} & \hat{\boldsymbol{e}}_{p3} \\ | & | & | \end{bmatrix} \begin{bmatrix} \cos\alpha \\ \sin\alpha\cos\beta \\ \sin\alpha\sin\beta \end{bmatrix}, \tag{4.8}$$

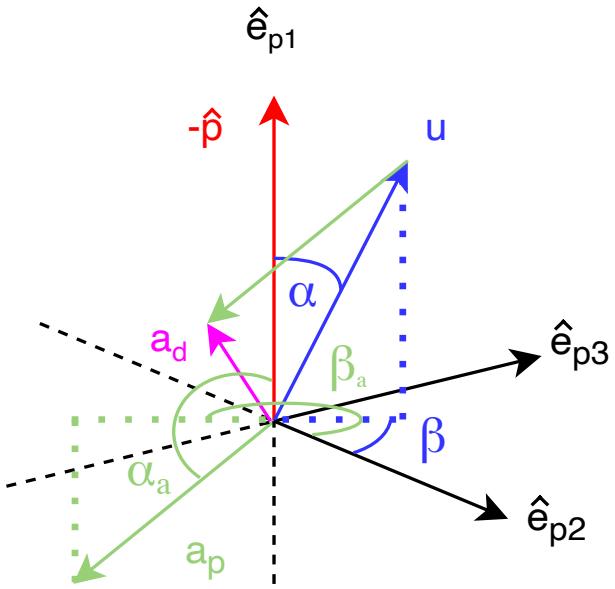


Figure 4.2: Illustration showing the cone-clock angles relative to the stable control direction in a two-body environment, $-\hat{\boldsymbol{p}}$.

This additional freedom allows further exploration of the control space and potential to exploit perturbations. A restriction on the cone angle α can be introduced to ensure Lyapunov stability when $\boldsymbol{a}_p \neq \boldsymbol{0}$ and the control authority is high enough. Equation (4.6) and Fig. 4.2 give:

$$\begin{aligned} \boldsymbol{p}^T\left(\boldsymbol{u} + \boldsymbol{a}_p\right) &< 0 \\ -\boldsymbol{p}^T\boldsymbol{u} &> \boldsymbol{p}^T\boldsymbol{a}_p \\ f\|\boldsymbol{p}\|\cos(\alpha) &> \|\boldsymbol{p}\|\hat{\boldsymbol{p}}^T\boldsymbol{a}_p \\ \cos(\alpha) &> \frac{\hat{\boldsymbol{p}}^T\boldsymbol{a}_p}{f}, \end{aligned} \tag{4.9}$$

This leads to three cases worth considering:

1. If $\hat{\boldsymbol{p}}^T\boldsymbol{a}_p \leq -f$, then $0 \leq \alpha \leq \pi$ will result a Lyapunov stable control.

2. If $-f < \hat{\boldsymbol{p}}^T \boldsymbol{a}_p < f$, then a Lyapunov stable control exists for

$$0 \leq \alpha < \arccos\left(\frac{\hat{\boldsymbol{p}}^T \boldsymbol{a}_p}{f}\right). \tag{4.10}$$

3. If $\hat{\boldsymbol{p}}^T \boldsymbol{a}_p \geq f$, then $\dot{Q} \geq 0$ is unavoidable and the system becomes unstable, but $\alpha, \beta = 0$ is selected to make $\dot{Q}$ as small as possible.

These scenarios are used to restrict the cone angle α. One of the novel ideas presented in this chapter uses $\boldsymbol{u}$ as given in Eq. (4.7) with these inequalities restricting the usable domain on α. The safest choice for Lyapunov stability is always $\alpha = 0$, unlike what is done in the literature. The conventional approach taken in the literature either ignores $\boldsymbol{a_p}$, cancels it out using the control - see the top row of Fig. 4.3, or re-derives the control with a higher-fidelity dynamics in the matrix $\boldsymbol{B}$ in Eq. (4.4). The cone-clock approach has the advantage that it avoids the requirement to adjust the control to all changes in the perturbations. Instead, the perturbation is simply used to constrain the potential control directions.

Alternative formulations for the cone-clock approach may exist. One potential such formulation involves simply multiplying $-\hat{\boldsymbol{p}}$ by a positive-definite rotation matrix. The positive-definite nature will ensure the resulting control $\boldsymbol{u}$ will remain in the positive hemisphere projecting onto $-\hat{\boldsymbol{p}}$. However, this will involve learning more parameters for the control law. It would also be challenging to impose the same limitations on α to ensure Lyapunov stability in the presence of perturbations in this formulation. Finally, the decoupling on the direction that impacts Lyapunov stability, α, and that which is orthogonal to it, β, can aid the learning process.

Whilst this cone-clock approach is important in determining the Lyapunov stability throughout the transfer, it can also be used to free up the Lyapunov control direction. The optimality of $-\hat{\boldsymbol{p}}$ is dependent on the formulation of the Lyapunov function itself. Thus, this approach is introduced to enable the exploitation of the perturbation without compromising Lyapunov stability.

Algorithm 3 shows how the training is done when the cone-clock approach is included. Figure 4.3 compares this cone-clock approach to the standard way of handling perturbations in the literature (for example [109]). There are four possible scenarios, although two of those collapse into a single scenario in the diagram. These four scenarios are divided into two halves: the perturbing acceleration $\boldsymbol{a}_p$ can be thought of as either aiding ($-\hat{\boldsymbol{p}} \cdot \boldsymbol{a}_p > 0$) or hindering ($-\hat{\boldsymbol{p}} \cdot \boldsymbol{a}_p < 0$) the decrease of the Lyapunov function. Note here that this Lyapunov function needs to be thought of in Keplerian dynamics, and the desired direction is indicated by the direction of $-\hat{\boldsymbol{p}}$. Classing the resulting disturbing acceleration as the combination of the control and perturbing acceleration $\boldsymbol{a}_d = \boldsymbol{u} + \boldsymbol{a}_p$, any projection $-\hat{\boldsymbol{p}} \cdot \boldsymbol{a}_d > 0$ is Lyapunov stable, whilst $-\hat{\boldsymbol{p}} \cdot \boldsymbol{a}_d < 0$ is Lyapunov unstable. Conventionally Lyapunov functions are desired to cancel out the perturbing accelera-

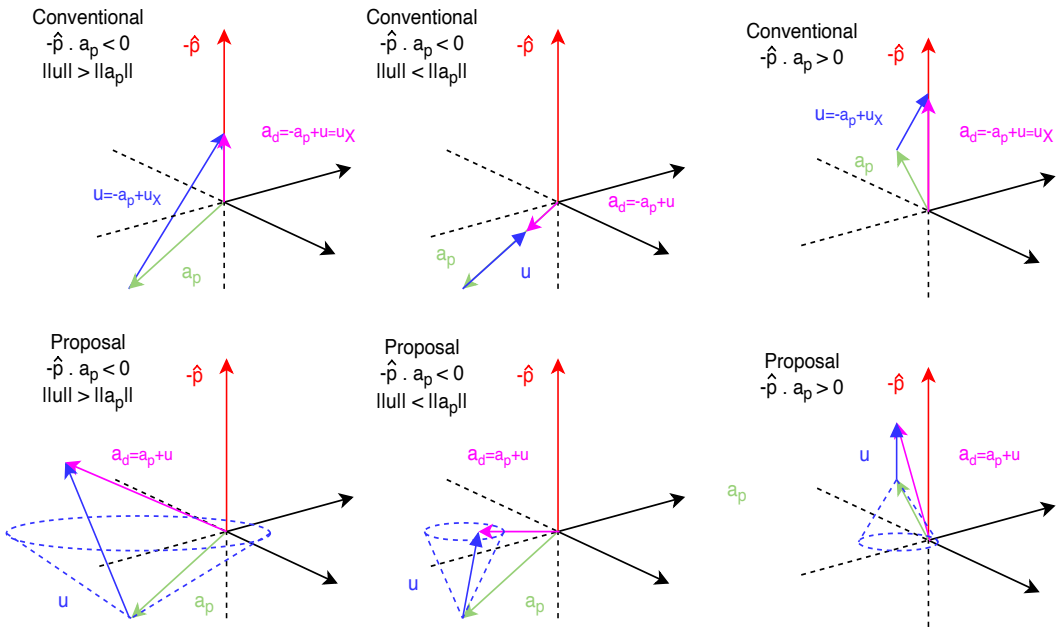


Figure 4.3: Illustration showing the potential configurations of the cone-clock angle with varying perturbing accelerations and control authorities. Dashed lines indicate the limiting cone for Lyapunov stability, always taking the cone angle from the $-\hat{\boldsymbol{p}}$ direction.

tion. The control vector is computed such that, if possible, $\boldsymbol{a}_d$ is parallel to $-\hat{\boldsymbol{p}}$ - see the top row of diagrams in Fig. 4.3.

Consider the four potential scenarios. First, the perturbation is hindering the Lyapunov function, but the magnitude of the control acceleration is greater than the perturbing acceleration and can be used to, in part, cancel out the perturbing acceleration. In this scenario the controller remains Lyapunov stable. In the second scenario, again the perturbation is hindering the Lyapunov function, but the magnitude of the control acceleration is less than the perturbing acceleration. As such, the controller do not have control authority and cannot maintain Lyapunov stability. Lastly, when the perturbation aids the decrease of the Lyapunov function, the control authority is irrelevant and two scenarios merge. The control vector is computed to first cancel the perturbation and then to aid decrease of the Lyapunov function. Pontani *et al.* [109] provide a very comprehensive analysis of these conditions for Lyapunov functions applied to station-keeping scenarios.

The proposed cone-clock approach has two potential modifications to this conventional approach. Firstly, it increases the freedom for the control vector. It is no longer required to cancel the perturbation, and the choice of α and β can be used to potentially exploit the perturbation. Secondly, and perhaps more importantly, this can be used to extend the potential domain over which the controller is Lyapunov stable. As can be seen in the central diagram on the bottom row in Fig. 4.3, what was not Lyapunov stable before is now forced to be Lyapunov stable. This comes from the restrictions on α seen in

Algorithm 3 Pseudocode: Training with Perturbations with cone-clock

Set initial state $\boldsymbol{X}_0$ and target state $\boldsymbol{X}_T$ ▷ Initialise Transfer
Set random parameters θ_k ▷ Initialise RL
while Training **do**
  for Batch N + 2 **do** ▷ N Stochastic $\boldsymbol{W}$ and 2 Deterministic $\boldsymbol{W}$
    while Propagating Transfer **do** ▷ Computing transfer trajectory
      if Training with Perturbations **then**
        $\boldsymbol{a_T} = \boldsymbol{a_p}$ ▷ Use perturbed dynamics
      else if Training without Perturbations **then**
        $\boldsymbol{a_T} = 0$ ▷ Use Keplerian dynamics
      $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}, \theta_k)$
      $-\hat{\boldsymbol{p}} \leftarrow \text{Lyapunov}(\boldsymbol{X}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0}))$
      $\boldsymbol{u} \leftarrow \text{Cone-clock}(-\hat{\boldsymbol{p}}, \boldsymbol{a_T})$ ▷ Using Eqs. (4.8) and (4.9)
      $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{a_T})$
  Update θ_k if $C(\theta) < C(\theta_k)$ ▷ Deterministic Update
while Deploying **do** ▷ Deploy learnt policy
  while Propagating Transfer **do** ▷ Computing transfer trajectory
    $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}, \theta_k)$
    $-\hat{\boldsymbol{p}} \leftarrow \text{Lyapunov}(\boldsymbol{X}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0}))$
    $\boldsymbol{u} \leftarrow \text{Cone-clock}(-\hat{\boldsymbol{p}}, \boldsymbol{a_p})$ ▷ Using Eqs. (4.8) and (4.9)
    $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{a_p})$ ▷ Use perturbed dynamics

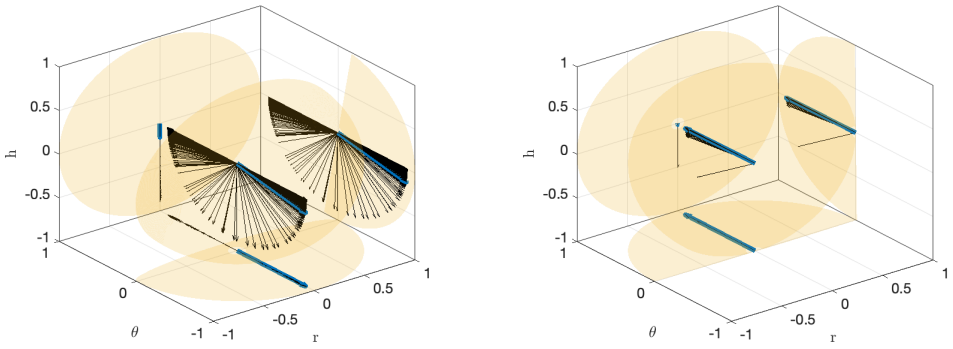
Eq. (4.10). The dashed lines in Fig. 4.3 indicate the limiting cone for Lyapunov stability, always taking the cone angle from the $-\hat{\boldsymbol{p}}$ direction.

4.3.1 Domain and Frequency

One of the concerns around introducing the cone-clock approach is whether it introduces unnecessary degrees of freedom. In Keplerian dynamics, $\boldsymbol{W}$ allow $-\hat{\boldsymbol{p}}$ to vary as they change the makeup of the Lyapunov function. However, in practice this does not mean they will cover the full domain of possible directions. This is instead dependent on the initial and target state, and the dynamical model. α and β add additional degrees of freedom whilst also overlapping with the domain covered by variations in $\boldsymbol{W}$.

Figure 4.4 shows a visual interpretation of the possible domain covered by $\boldsymbol{W}$ and α and β. Here, the blue vector indicates the classical Q-law control vector at the initial orbital state. Both figures target a GEO, starting at a GTO on the left and a LEO on the right. The yellow hemisphere demonstrates the domain that would be covered up to $\alpha = \pi/2$, i.e. the possible directions for $\boldsymbol{u}$ that would remain Lyapunov stable under the original classical Q-law expression. The black vectors are created using a $10 \times 10 \times 10$ grid on $0 \leq W_a, W_e, W_i \leq 1$, indicating the possible directions at that instance in time an optimiser or learning process could use. Clearly the control vectors resulting from this grid over $\boldsymbol{W}$ is a much narrower region and does not necessarily overlap with the hemisphere created by α and β.

Another issue worth discussing is the frequency at which $\boldsymbol{u}$ can vary throughout the



(a) GTO (b) LEO

Figure 4.4: Comparing the control vectors for a grid of $\boldsymbol{W}$ values and the hemisphere $\alpha = \pi/2$ around the classical control direction. Normalised control vectors plotted in the $R\theta h$ frame.

transfer. For simplicity, consider two cases with constant weights and cone-clock angles for the entire transfer. If case 1 uses $\boldsymbol{W} = \mathbf{1}$ and $\alpha \neq 0$ but constant throughout, then $\boldsymbol{u}$ evolves according to the dynamics and the evolution of the Lyapunov function $\boldsymbol{V}(\boldsymbol{W} = \mathbf{1})$. However, if instead case 2 has $\alpha = 0$ and $\boldsymbol{W}$ is chosen such that $\boldsymbol{u}$ matches case 1 at the start of the transfer, then the control will evolve in a different manner, according to $\boldsymbol{V}(\boldsymbol{W})$, than in case 1. Hence, whilst the two cases can represent each other at a given instance during the transfer, they will cover different domains throughout. This has implications in the learning process, where allowing either $\boldsymbol{W}$, α and β, or both to vary, could result in multiple paths to the same solution. This effect is non-trivial and would need significant further investigation to understand fully, but can explain some of the results found in the next section. There is a degree of redundancy where $\boldsymbol{W}$ and α, β overlap and this can lead to surfaces of optimal solutions in the search space.

4.3.2 Cone-clock Results

Using this new cone-clock implementation, two different approaches were considered. Firstly, α, β were trained alongside $\boldsymbol{W}$ simultaneously. Here, and henceforth, RL Q-Jac+CC refers to this RL Q-Jac implementation combined with the cone-clock approach. However, this adds two additional parameters to the learning process and can make it more challenging to find the optimal solution. In addition, as Figs. 4.4a and 4.4b show, it also expands on the design domain at any given moment in time. Hence, a second approach where $\boldsymbol{W}$ remain fixed was considered - here referred to as RL Q-Jac+CC Fixed. In this case, either the policy trained with or without perturbations needs to be used. In an attempt to understand where the perturbations can be exploited, the policy trained in Keplerian dynamics is used. The RL Q-Jac approach from Section 4.2.1 is included for comparison.

Tables 4.5, 4.6, 4.7 and 4.8 give the time- and mass-optimal results for the GTO-GEO

Table 4.5: Comparison of time-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law) with cone-clock approach. Grey highlights indicate the RL Q-Jac+CC solution trained with Perturbations.

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	RL Q-Jac	-	-	-	-	137.70	212.31	2.201	-
	RL Q-Jac+CC Fixed	-	-	-	-	137.70	212.31	2.201	-
	RL Q-Jac+CC	-	-	-	-	137.85	212.42	2.202	-
J_2	RL Q-Jac	137.84	212.53	2.204	0.55	137.98	212.75	2.206	0.54
	RL Q-Jac+CC Fixed	137.92	212.64	2.205	0.57	-	-	-	-
	RL Q-Jac+CC	137.87	212.55	2.204	0.62	138.16	212.89	2.208	0.74
3^{rd}-body	RL Q-Jac	138.04	212.82	2.207	0.49	137.97	212.73	2.206	0.49
	RL Q-Jac+CC Fixed	137.92	212.64	2.205	0.50	-	-	-	-
	RL Q-Jac+CC	137.73	212.36	2.202	0.49	138.12	212.84	2.207	0.54
Eclipse	RL Q-Jac	139.19	212.47	2.203	-	138.82	211.77	2.195	-
	RL Q-Jac+CC Fixed	138.72	211.68	2.194	-	-	-	-	-
	RL Q-Jac+CC	139.15	212.47	2.203	-	138.80	211.71	2.195	-

Table 4.6: Comparison of mass-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law) with cone-clock approach. Here a maximum allowed time-of-flight of 150 days is desired. Grey highlights indicate the RL Q-Jac+CC solution trained with Perturbations.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	RL Q-Jac	-	-	-	-	-	149.91	191.88	1.978	192.13	-
	RL Q-Jac+CC Fixed	-	-	-	-	-	149.91	191.88	1.978	192.13	-
	RL Q-Jac+CC	-	-	-	-	-	149.73	192.24	1.985	192.61	-
J_2	RL Q-Jac	150.81	192.32	1.983	193.95	0.55	149.74	193.09	1.991	193.09	0.55
	RL Q-Jac+CC Fixed	149.75	192.99	1.990	192.99	0.60	-	-	-	-	-
	RL Q-Jac+CC	149.67	193.43	1.995	193.43	0.60	149.70	193.49	1.996	193.49	0.65
3^{rd}-body	RL Q-Jac	149.82	192.75	1.988	192.85	0.49	150.18	192.05	1.980	192.72	0.48
	RL Q-Jac+CC Fixed	149.78	192.38	1.984	192.43	0.49	-	-	-	-	-
	RL Q-Jac+CC	149.78	192.46	1.984	192.46	0.50	150.09	192.64	1.986	193.17	0.50
Eclipse	RL Q-Jac	149.75	192.61	1.986	192.61	-	150.50	191.56	1.975	192.72	-
	RL Q-Jac+CC Fixed	149.74	192.19	1.982	192.19	-	-	-	-	-	-
	RL Q-Jac+CC	149.84	192.20	1.982	192.20	-	150.37	193.11	1.981	193.11	-

Table 4.7: Comparison of time-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law) with cone-clock approach. Grey highlights indicate the RL Q-Jac+CC solution trained with Perturbations.

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	RL Q-Jac	-	-	-	-	180.44	193.52	5.691	-
	RL Q-Jac+CC Fixed	-	-	-	-	180.44	193.52	5.691	-
	RL Q-Jac+CC	-	-	-	-	180.50	193.50	5.691	-
J_2	RL Q-Jac	180.39	193.46	5.689	0.50	181.59	194.75	5.731	0.48
	RL Q-Jac+CC Fixed	181.61	194.64	5.727	0.67	-	-	-	-
	RL Q-Jac+CC	181.53	194.66	5.728	0.63	181.85	194.95	5.737	0.56
3^{rd}-body	RL Q-Jac	180.05	193.10	5.678	0.49	180.45	193.53	5.691	0.49
	RL Q-Jac+CC Fixed	180.44	193.51	5.691	0.49	-	-	-	-
	RL Q-Jac+CC	180.75	193.78	5.700	0.51	180.49	193.49	5.690	0.52
Eclipse	RL Q-Jac	217.46	193.57	5.693	-	225.81	200.27	5.909	-
	RL Q-Jac+CC Fixed	220.08	195.59	5.758	-	-	-	-	-
	RL Q-Jac+CC	217.70	193.67	5.696	-	222.95	198.22	5.843	-

and LEO-GEO transfers respectively. First thing to note is the RL Q-Jac+CC Fixed is always equal to or better than the RL Q-Jac trained in Keplerian dynamics but deployed in perturbed dynamics. This is because when $\alpha = 0$ they are have the same result. All these simulations are initialised with $\alpha = 0$ so they at least reach this result. Brief experiments were also run with different initialisation values and they are able to converge back to this value. Naturally, this could limit the possible exploration of the agent, and perhaps better results can be achieved by initialising with $\alpha \neq 0$. Secondly, adding the additional

Table 4.8: Comparison of mass-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law) with cone-clock approach. Here a maximum allowed time-of-flight of 200 days is desired. Grey highlights indicate the RL Q-Jac+CC solution trained with Perturbations.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	RL Q-Jac	-	-	-	-	-	199.75	186.97	5.481	186.97	-
	RL Q-Jac+CC Fixed	-	-	-	-	-	199.75	186.97	5.481	186.97	-
	RL Q-Jac+CC	-	-	-	-	-	200.06	187.45	5.486	187.45	-
J_2	RL Q-Jac	199.75	190.78	5.603	190.78	0.51	201.87	188.61	5.534	190.89	0.49
	RL Q-Jac+CC Fixed	197.78	187.88	5.511	187.88	0.79	-	-	-	-	-
	RL Q-Jac+CC	196.45	189.88	5.575	189.88	0.67	200.26	187.57	5.501	188.12	0.65
3^{rd}-body	RL Q-Jac	199.49	186.80	5.476	186.80	0.50	199.69	186.94	5.480	186.94	0.50
	RL Q-Jac+CC Fixed	199.70	186.94	5.480	186.94	0.49	-	-	-	-	-
	RL Q-Jac+CC	199.47	187.21	5.489	187.21	0.56	199.74	187.08	5.485	187.08	0.54
Eclipse	RL Q-Jac	215.95	191.74	5.634	209.11	-	260.20	185.63	5.439	250.46	-
	RL Q-Jac+CC Fixed	218.08	191.80	5.636	211.46	-	-	-	-	-	-
	RL Q-Jac+CC	235.35	188.25	5.522	226.43	-	252.80	185.54	5.436	242.43	-

freedom of the cone-clock approach can alter the performance of the controller. It is not guaranteed to result in improvement, and instead can be more complex to learn. It is believed this is due to the partially overlapping domains of the angles α and β with the weights $\boldsymbol{W}$.

In addition to the time-of-flight, propellant mass and ΔV, the *Fraction* $\dot{V} < 0$ indicates the fraction of the whole transfer during which $\boldsymbol{a}_p$ is aiding the decrease of the Lyapunov function. In other words, $\boldsymbol{p}^T \boldsymbol{a}_p < 0$. Clearly the introduction of the cone-clock approach in the J_2 scenario increases the time during which the perturbation aids the decrease of the Lyapunov function. However, this does not consistently reflect an improvement in the objective value. It should be noted that decreasing V as quickly as possible is not guaranteed to lead to a more optimal solution.

Table 4.9: Summary: Training RL with Perturbations. Red indicates less than 50% simulations, orange between 50% and 75% and green for more than 75%. Not a statistical analysis but a summary of Tables 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8.

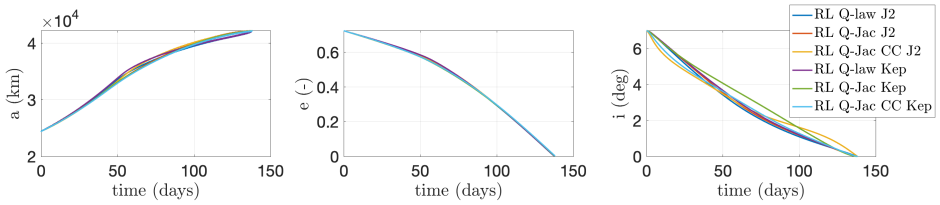
	Reinforced Lyapunov Controller		
Dynamics	All (16)	With Jacobian (12)	With CC (8)
J_2			
3^{rd}-body			
Eclipse			
Objective	**All (24)**	**With Jacobian (18)**	**With CC (12)**
Time			
Mass			

Whist it was not possible to do a full statistical analysis, it is still interesting to summarise the simulations presented in Tables 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8. These tables present a total of 96 simulations, training with and without perturbations for 48 combinations of dynamics, objectives and controllers. There are 24 time- and mass-optimal simulations respectively, and 16 simulations for each of J_2, 3^{rd}-body and Eclipse dynamics. Table 4.9 provides a visual summary assessing whether the new train-

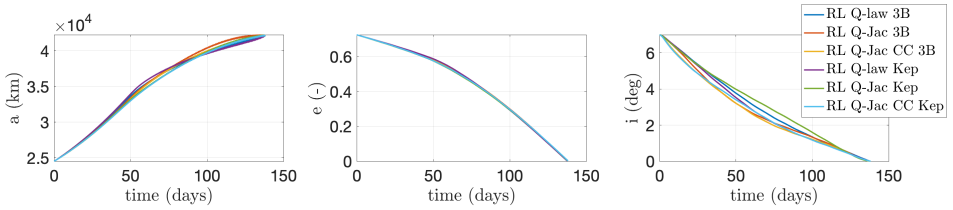
ing approach improves the performance of the Reinforced Lyapunov Controllers. The main take-away is training with the cone-clock appears to have the most success, and scenarios involving eclipse events gain the most by incorporating these into the training regime.

Figure 4.5 shows the evolution of the COEs for the time-optimal GTO-GEO transfer. There is little difference in the observed behaviour, and this is reflected in the minimal variation in the time-of-flight. Figure 4.6 shows the behaviour of the perturbing acceleration and the cone-clock angles for the time-optimal GTO-GEO transfer. The top row plots the angle between the perturbation and the control vector. The central row indicates the magnitude of the perturbation and the bottom plot indicates the restriction this puts on the possible cone-angles whilst attempting to preserve stability.

The noticeable difference between the J_2 and 3rd-body scenarios is the magnitude of the perturbing acceleration and the subsequent restrictions this puts on the cone-clock angle. For the J_2 scenario, the reduction in magnitude during the transfer is also reflected in the reduced limitation on α towards later stages of the transfer. In the 3rd-body case there is limited advantage or disadvantage in utilising the perturbing acceleration as it is simply not strong enough.



(a) J_2

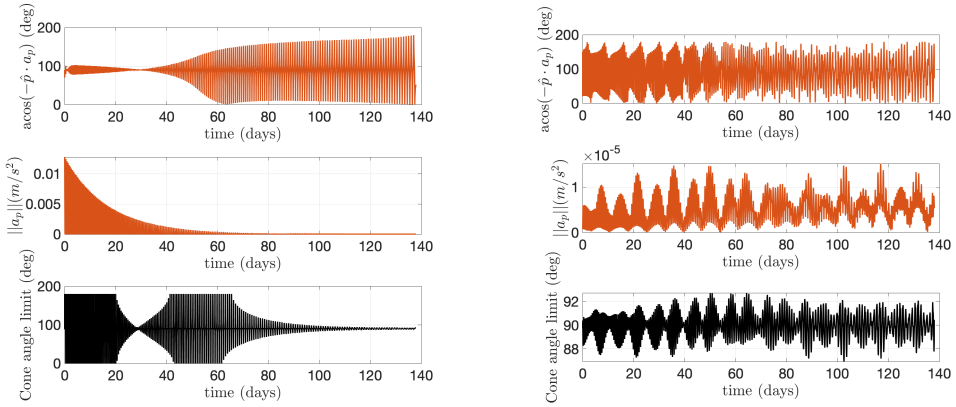


(b) 3rd-body

Figure 4.5: Figure comparing the evolution of the orbital elements for the RL Q-law, RL Q-Jac and RL Q-Jac+CC controllers for a GTO-GEO time-optimal transfer for both the J_2 and 3rd-body perturbations.

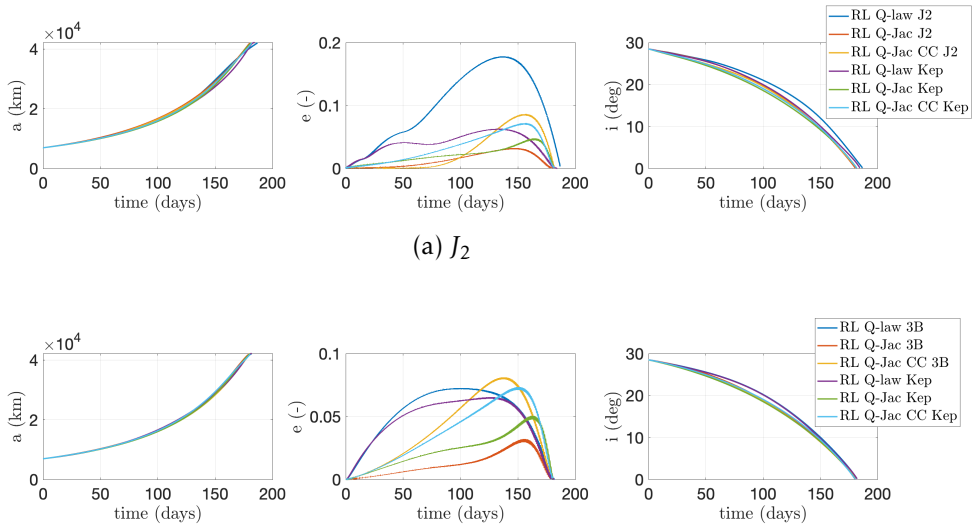
Figures 4.7 and 4.8 give the same plots for the time-optimal LEO-GEO transfers. Compared to the GTO-GEO case, the J_2 perturbation magnitude does not oscillate as much as the spacecraft spends longer in LEO, experiencing similar magnitudes of perturbing acceleration. In the 3rd-body scenario the increasing magnitude is much more noticeable, as is the corresponding restriction on the α-domain.

The eclipse simulations are skewed by the maximum time-of-flight for the mass-



(a) J_2 (b) 3^{rd}-body

Figure 4.6: Plots of the perturbing acceleration $\boldsymbol{a}_p$ in the cone-clock frame for the time-optimal GTO-GEO transfers, relative to the stable control direction in a two-body environment, $-\hat{\boldsymbol{p}}$.



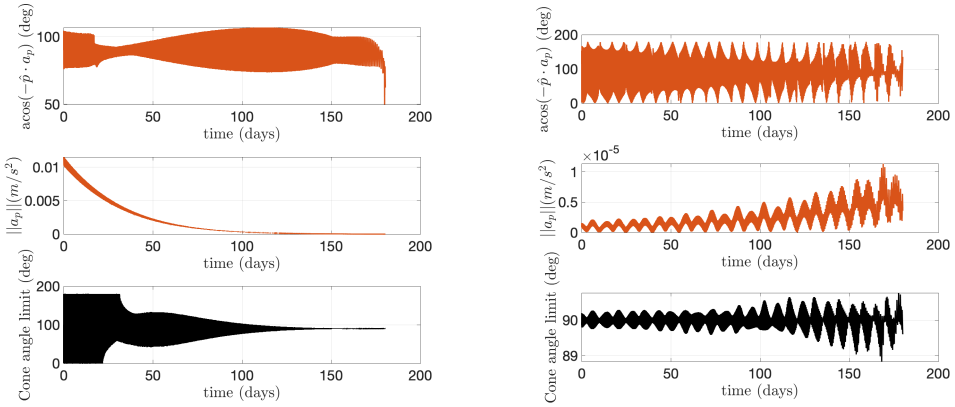
(a) J_2

(b) 3^{rd}-body

Figure 4.7: Figure comparing the evolution of the orbital elements for the RL Q-law, RL Q-Jac and RL Q-Jac+CC controllers for a LEO-GEO time-optimal transfer for both the J_2 and 3^{rd}-body perturbations.

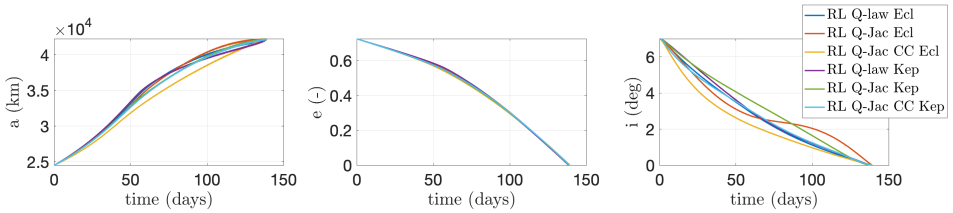
optimal simulations, where the nature of the eclipse event results in the time-of-flight increasing. As such, in order for the spacecraft to arrive before this maximum threshold, more fuel is used to do so, whereas this is not penalised for the controller trained without the eclipse events but deployed with them.

Whilst the first and partially the second objective discussed in Section 4.2 have been investigated, it is still difficult to conclude that the perturbing accelerations have been exploited. Whilst important to consider, J_2 and 3^{rd}-body perturbations have minor effects on GTO-GEO and LEO-GEO transfers, and the discontinuity within eclipse effects means they can be challenging to the control approach without offering the potential to

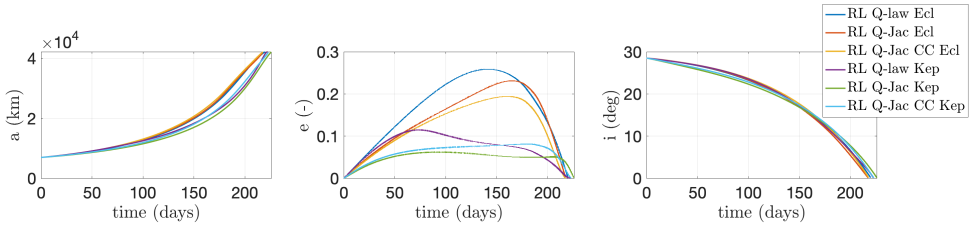


(a) J_2 (b) 3rd-body

Figure 4.8: Plots of the perturbing acceleration $\boldsymbol{a}_p$ in the cone-clock frame for the time-optimal LEO-GEO transfers, relative to the stable control direction in a two-body environment, $-\hat{\boldsymbol{p}}$.



(a) GTO-GEO



(b) LEO-GEO

Figure 4.9: Figure comparing the evolution of the orbital elements for the RL Q-law, RL Q-Jac and RL Q-Jac+CC controllers for a both GTO-GEO and LEO-GEO time-optimal transfer with eclipse events.

be exploited. Thus, increasing the perturbing acceleration magnitude or using a different test case where the perturbed dynamics are more relevant is considered. The 3rd-body acceleration can be increased by going to higher altitudes beyond GEO, and this is looked at extensively in Chapter 7. The J_2 perturbation, on the other hand, is most dominant at lower altitudes. This motivates the transfer scenario in Section 4.4 between two SSOs in LEO.

4.4 Sun-Synchronous Orbit transfer

As mentioned previously, the averaged J_2 causes a secular drift in Ω and ω. This is used to construct SSOs. For a SSO, the drift of Ω needs to match the rate at which the right ascension of Sun moves throughout the year, i.e. $\dot{\Omega} = 360/365.25$ °/year. Using Eq. (2.15), if the semi-major axis is 6927km and the eccentricity 0.005 then the required inclination is 97.5880°. Thus, Table 4.10 gives the orbital parameters for a proposed orbit transfer between two SSOs. Here, all orbital parameters are equal except the RAAN, which is offset by 10°. This is a hypothetical scenario constructed to highlight the importance of the J_2 perturbation, and for simplicity only the secular contribution is included. In Keplerian dynamics, this would require a very large ΔV to achieve the desired $\Delta\Omega$. The modelled spacecraft has a mass of 1200 kg, a thrust of 0.4017 N and an I_{sp} of 3300 s.

The first scenario to consider is what occurs when the target orbit is fixed, as in all simulations until this point. In this instance, one feasible solution for the controller is to do nothing, and coast from the initial state to the target state using the effect of J_2 to change the RAAN. For a $\Delta\Omega = -10°$ change, this should take $350/360 \times 365.25 = 355.10$ days. However, rather than targeting a fixed RAAN at a given epoch, it is more realistic to target a total $\Delta\Omega$ from the initial orbit to the target orbit. If the initial orbit itself is drifting, then so should the target orbit. One might suggest this is an idealised constellation deployment scenario.

Thus, in this section, the target orbit is non-stationary, and will drift thanks to the secular effect of J_2. An important point to note is this will impact the Lyapunov stability of the controller. Reconsidering the derivations in Section 3.3.1, it is clear $\dot{V}$ depends on a contribution from $\dot{\boldsymbol{X}}$, $\dot{\boldsymbol{W}}$ and $\dot{\boldsymbol{X}}_T$. For simplicity, in this section $\dot{\boldsymbol{X}}_T$ is ignored and use the same implementation as before. Hence, the Lyapunov controller is no longer Lyapunov stable - this is discussed further in Section 8.2.2. This does, however, maintain the unknown dynamical nature of the system.

Time- and mass-optimal simulations are considered. In the time-optimal case, the spacecraft engine is switched on for the entire transfer. In the mass-optimal case, the effectivity threshold is used to switch the engine off and allow the spacecraft to coast. The expectation is that, by increasing the semi-major axis and eccentricity of the orbit, the spacecraft can use the reduced impact of the J_2 perturbation to allow the target orbit to catch up with the current osculating orbit before the spacecraft uses its engine again to lower the altitude and converge to the target orbit. In this fashion it should be possible to conserve fuel and allow the spacecraft to exploit the existence of the perturbation.

With this is mind, a PSO is used to provide six different benchmark results. Three of these are with fixed parameters and three are with time-dependent parameters, achieved using cubic spline interpolation. In both cases the three scenarios considered are as follows. Firstly, the PSO optimises the weights $\boldsymbol{W}$ as in all benchmark simulations before. Secondly, the nominal values of $\boldsymbol{W} = 1$ are used and the cone-clock angles α, β are op-

Table 4.10: Initial and target orbital elements for a LEO-LEO Sun-Synchronous Orbit transfer.

	a (km)	e	i (°)	Ω (°)	ω (°)	ν (°)
Initial	6927	0.005	97.588	0.0	0.0	0.0
Target	6927	0.005	97.588	-10.0	*free*	*free*

timised instead. This should indicate the extent to which the cone-clock approach can improve a fixed controller. Lastly, both $\boldsymbol{W}$ and α, β are optimised. This demonstrates the combined potential of the two approaches. Theoretically this last simulation should at least match the first in terms of optimality, as that solution exists within its search space. However, the added degrees of freedom will complicate the optimisation process and could make it more difficult for it to converge to the same solution if that is, indeed, the true optimum. In all mass-optimal simulations, the effectivity threshold η_a^t is also optimised.

Table 4.11: Comparison of time and mass-optimal Sun-Synchronous Orbit transfers using a PSO Q-law with cone-clock approach. Dynamics are Keplerian with secular J_2. Grey highlights indicate the combined $\boldsymbol{W}, \alpha, \beta$ solution.

Objective	Method	Time (days)	Prop (kg)	ΔV (km/s)
Time	Classical	53.93	57.84	1.599
	PSO Fixed $\boldsymbol{W}$	42.75	45.85	1.261
	PSO Fixed α, β	51.82	55.58	1.535
	PSO Fixed $\boldsymbol{W}, \alpha, \beta$	42.87	45.98	1.264
	PSO Spline $\boldsymbol{W}$	36.13	38.75	1.062
	PSO Spline α, β	49.91	53.53	1.477
	PSO Spline $\boldsymbol{W}, \alpha, \beta$	34.44	36.94	1.012
Mass	PSO Fixed $\boldsymbol{W}, \eta_a^t$	99.75	27.27	0.744
(< 100 days)	PSO Fixed α, β, η_a^t	90.38	36.41	0.997
	PSO Fixed $\boldsymbol{W}, \alpha, \beta, \eta_a^t$	99.75	29.10	0.794
	PSO Spline $\boldsymbol{W}, \eta_a^t$	99.75	10.34	0.280
	PSO Spline α, β, η_a^t	99.75	21.44	0.583
	PSO Spline $\boldsymbol{W}, \alpha, \beta, \eta_a^t$	99.75	7.08	0.191

Table 4.11 shows the results for the PSO Q-law simulations. Firstly, in both the time-optimal and mass-optimal cases the PSO Spline results outperform the PSO Fixed simulations. This justifies the need for a time-dependent approach for the Q-law parameters, whether that be the weights $\boldsymbol{W}$, the cone-clock angles α, β or the effectivity threshold η_a^t. In the fixed parameter simulations, the $\boldsymbol{W}$-only simulations outperform the $\boldsymbol{W}, \alpha, \beta$-simulations, possibly due to the increased complexity in the search space. However, in the PSO Spline approach, the most optimal solutions come from the combined $\boldsymbol{W}, \alpha, \beta$ simulations.

The time-optimal benchmark is 34.44 days, using 36.94 kg of propellant mass and 1.012 km/s ΔV. This is 1.69 days better than the $\boldsymbol{W}$ only simulation and 8.31 days better

than the best PSO Fixed solution. In the mass-optimal simulations the benchmark is just 7.08 kg, for a time-of-flight of 99.75 days and using 0.191 km/s ΔV. Again this outperforms the conventional PSO Spline $\boldsymbol{W}$ result by 3.26 kg and the PSO Flat solution by 20.19 kg, although this heavily impacted by the time-dependence of the effectivity threshold. Overall these results confirm the addition of the cone-clock increases the possible control domain and can lead to more optimal solutions in terms of time-of-flight and propellant mass consumption.

Following on from the PSO benchmark simulations, the RL is investigated for the same transfer. Based on the results from the PSO simulations, the domain for the effectivity threshold is increased from $0 \leq \eta_a^t \leq 1$ to $-0.2 \leq \eta_a^t \leq 1.2$. Naturally, only effectivity values $0 \leq \eta_a \leq 1$ have physical meaning. However, as in Section 3.3.2, it is possible to have non-physical neurons in the actor network which can further improve the performance in the physical domain. Allowing $\eta_a^t > 1.0$ will make it easier for the RL agent to experiment with coasting arcs during the transfer. As in previous sections, a comparison between the RL Q-law and RL Q-Jac is given. In addition, the cone-clock angles are included in the RL Q-Jac+CC simulations to investigate whether the current RL architecture can also exploit the additional degrees of freedom.

Unlike in previous chapters and sections, where the initial and target orbits differ significantly, here the only difference lies in RAAN. As such, different inputs to the actor network were considered, given the apparent struggles of the standard input used thus far. Instead of providing $\boldsymbol{X}$ as in Section 3.3.1.1, it is also possible to provide $\Delta\boldsymbol{X}$, i.e. the difference between the current and target state, in a similar fashion to the way the Q-law and other Lyapunov functions handle the inputs ($\delta(\boldsymbol{X}, \boldsymbol{X}_T)$). Alternatively, one can provide both $\boldsymbol{X} + \Delta\boldsymbol{X}$, although this necessitates doubling the network size. Ignoring the impact of $\dot{\boldsymbol{X}}_T$, which is a major limitation but is already done in all the Lyapunov formulations in this section, the derivation of the Jacobian from Section 3.3.1.2 can still be used.

Table 4.12 shows the results for the Reinforced Lyapunov Controller simulations. Six different scenarios were considered: time- and mass-optimal transfers for network inputs of $\boldsymbol{X}$, $\Delta\boldsymbol{X}$ and $\boldsymbol{X} + \Delta\boldsymbol{X}$ respectively. The three different controllers RL Q-law, RL Q-Jac and RL Q-Jac+CC were investigated for each. Four out of the six scenarios considered perform best with the RL Q-Jac+CC controller. This follows the findings with the PSO simulations and, at the very least, demonstrates the cone-clock approach broadens the search space and enables solutions which are otherwise challenging or impossible to achieve.

The PSO Spline $\boldsymbol{W}$ should provide a benchmark for the RL simulations here, however it is clear this formulation is unable to get close to the benchmarks in both time- and mass-optimal simulations. In the time-of-flight case, the RL Q-law takes 42.51 days, a full 6.38 days behind the PSO Spline $\boldsymbol{W}$, and it is only just able to out perform the PSO Fixed $\boldsymbol{W}$ simulation. However, introducing the Jacobian and then the cone-clock

Table 4.12: Comparison of time and mass-optimal Sun-Synchronous Orbit transfers using a Reinforced Lyapunov Controller (Q-law) with cone-clock approach. Dynamics are Keplerian with secular J_2. The best results are highlighted in grey.

Objective	Input	Method	Time (days)	Prop (kg)	ΔV (km/s)
Time	$\boldsymbol{X}$	RL Q-law	42.51	45.59	1.254
		RL Q-Jac	39.15	41.99	1.153
		RL Q-Jac+CC	38.31	41.08	1.127
	$\Delta\boldsymbol{X}$	RL Q-law	42.65	45.75	1.258
		RL Q-Jac	39.40	42.25	1.160
		RL Q-Jac+CC	35.51	38.09	1.044
	$\boldsymbol{X}+\Delta\boldsymbol{X}$	RL Q-law	43.39	46.45	1.278
		RL Q-Jac	36.20	38.83	1.064
		RL Q-Jac+CC	35.92	38.52	1.056
Mass	$\boldsymbol{X}$	RL Q-law	99.81	27.32	0.745
(< 100 days)		RL Q-Jac	93.47	12.76	0.346
		RL Q-Jac+CC	96.28	8.26	0.223
	$\Delta\boldsymbol{X}$	RL Q-law	98.67	20.34	0.553
		RL Q-Jac	98.60	7.35	0.199
		RL Q-Jac+CC	98.99	8.86	0.240
	$\boldsymbol{X}+\Delta\boldsymbol{X}$	RL Q-law	99.66	30.15	0.824
		RL Q-Jac	99.57	8.71	0.236
		RL Q-Jac+CC	98.58	9.57	0.259

approach significantly improves this to, at best, 36.20 days and then 35.51 days respectfully, only 1.07 days off the best performing PSO simulations.

There are three takeaway messages here. Firstly, it is clear the RL is not a global optimiser and struggles to find the global minimum, but rather converges to a sub-optimal local minimum. Next, the addition of the cone-clock helps guide the RL towards the more optimal solution. This is true of all three simulations with $\boldsymbol{X}$, $\Delta\boldsymbol{X}$ and $\boldsymbol{X}+\Delta\boldsymbol{X}$ network inputs. Ultimately the $\Delta\boldsymbol{X}$ network input achieves the best performance. This poses an interesting question on how best to formulate the inputs for the actor network and whether it is transfer dependent. In this scenario, because Ω and Ω_T drift as a result of J_2, it appears $\Delta\Omega$ provides more information on the status of the transfer. The $\boldsymbol{X}+\Delta\boldsymbol{X}$ network likely suffers from the increased number of parameters than need to be learnt to compute the optimal policy.

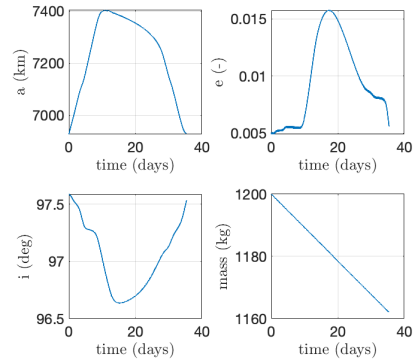
Looking at the mass-optimal simulations, a similar story emerges. Using the original $\boldsymbol{X}$ input, the RL Q-law approach struggles to minimise the propellant mass beyond the PSO Flat simulations, and at 27.32 kg it is more than 2.5 times worse than the PSO Spline $\boldsymbol{W}$. This improves to 20.34 kg with $\Delta\boldsymbol{X}$ network input, although it is still quite a way off the best available solutions, and the $\boldsymbol{X}+\Delta\boldsymbol{X}$ network is even worse at 30.15 kg. As seen in Section 3.4.2, the RL Q-Jac can often handle the mass-optimal simulations better because it provides more accurate information about the effectivity η_a. This is seen here, with 12.76 kg, 7.35 kg and 8.71 kg solutions for network inputs respectfully. Unusually

in the $\boldsymbol{X}$ network scenario, the solution does not utilise the allowed maximum time-of-flight, suggesting it is stuck in a local sub-optimal minimum. This problem is alleviated by the introduction of the cone-clock approach, which uses just 8.26 kg of fuel, 0.223 km/s ΔV to perform the transfer in 96.28 days. The cone-clock approach is not quite as good in the other two network scenarios, but it is still close to the PSO benchmark. Each case outperforms the PSO Spline $\boldsymbol{W}$ benchmark. Whilst that is not the solution provided by optimal control theory (something that was unfortunately not accessible in these simulations), it is an indication of potential solutions that exist for the transfer using conventional approaches in the literature.

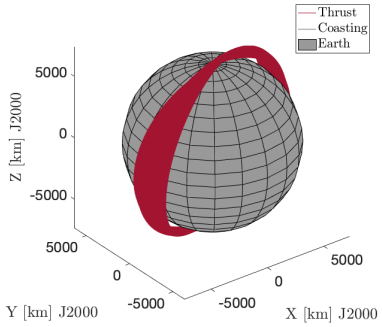
Figures 4.10 and 4.11 show the time- and mass-optimal results for the RL Q-Jac+CC controller. In the time-optimal case, the thrust remains on for the duration of the transfer. The controller increases the semi-major axis and eccentricity, and lowers the inclination during the middle of the transfer. Looking at Fig. 4.10f, this shows that the rate-of-change of the spacecraft RAAN is reduced compared to the target orbit, and this enables the target orbit to catch up with the current osculating orbit. During the middle of the transfer the contribution from J_2, shown in blue, is reduced.

The weights are given in Fig. 4.10c and show a distinct change in behaviour at the beginning at end of the transfer compared to the middle section. The W_Ω profile is perhaps the most interesting, as it is reduced significantly during the middle of the transfer, prioritising the other orbit elements despite being the only non-zero difference at the start. Figure 4.10d shows how the weights combine with the other terms in the Q-law to indicate which have, overall, the greatest contribution to the control direction. Clearly the RAAN has the largest contribution to begin with, but this is eventually super-seeded by the semi-major axis. Figure 4.10e shows the cone angle throughout the transfer, along with the osculating perturbing acceleration that would be present at the spacecraft's orbit, although this is not the case in the dynamics. However it is being used to provide the Lyapunov stability limit on the cone angle, hence the oscillations seen there.

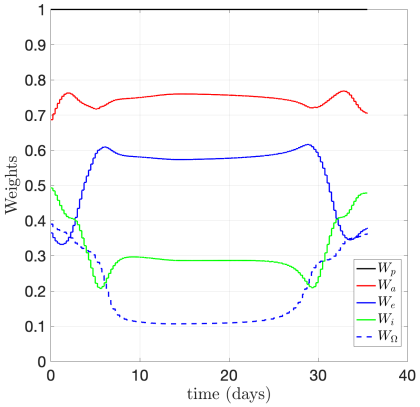
The mass-optimal simulation has similar results, although the behaviour is exaggerated by the effectivity threshold and corresponding coast periods. The desired maximum time-of-flight of 100 days is utilised allow the spacecraft to coast whilst the target orbit catches up to it. In order to do so, and ensure there is a difference in $\dot{\Omega}$, the spacecraft first has to climb to a higher altitude orbit, as can be see in Fig. 4.11a. Then it switches of the engine by pushing the effectivity threshold $\eta_a^t \to 1$. This is clearly seen in Fig 4.11e. At the end of the transfer, after about 90 days, the threshold is lowered and the spacecraft thrusts to converge to the target orbit. Comparing Fig. 4.11f with Fig. 4.10f, you can see that the contribution from J_2 is reduced by a greater amount in the time-optimal simulations. This is because the spacecraft cannot afford to wait for the target orbit to catch up, and instead must attempt to get the largest difference in $\dot{\Omega}$ between the current and target orbit. In the mass-optimal case, the maximum allowed time-of-flight informs the necessary difference between $\dot{\Omega}$ that is required.



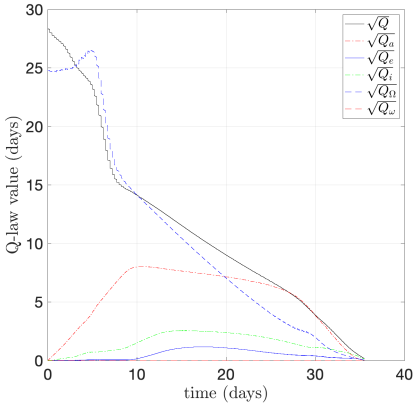
(a) Orbital Parameters



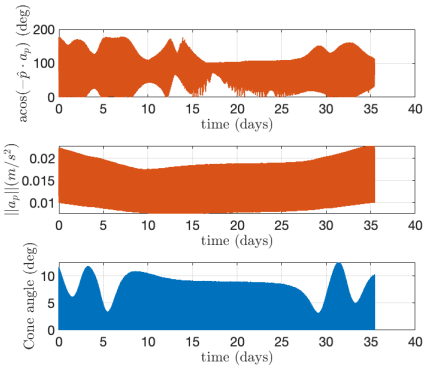
(b) Trajectory



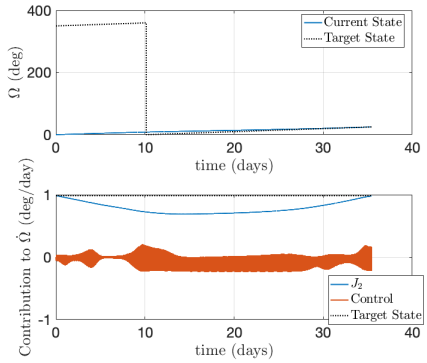
(c) Weights



(d) $\sqrt{Q}$ Contributions



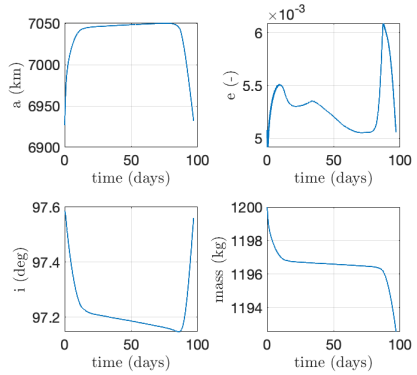
(e) Cone-clock and osculating $\boldsymbol{a}_p$



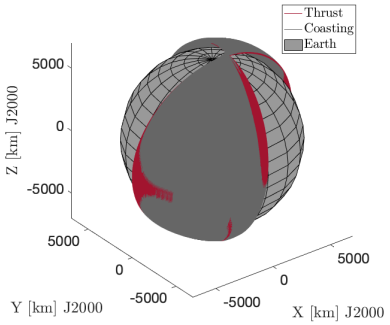
(f) Influence of J_2

Figure 4.10: Time-optimal Sun-synchronous transfers for a RL Q-Jac+CC controller.

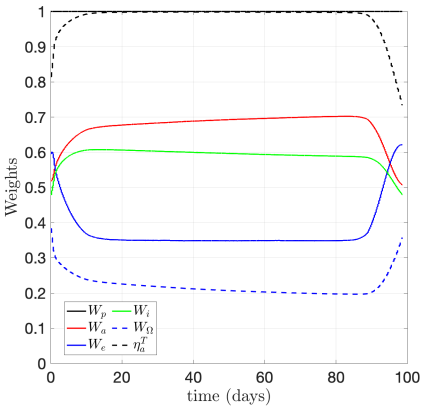
Interestingly, this behaviour can be achieved without informing the Lyapunov controller of the drifting target orbit and the existence of the secular J_2. A potential improvement might be to add a waypoint, similar to the work done by Peterson *et al.* [69] in Lunar orbiting environments, however this appears to be unnecessary in this instance. In addition, by not including a waypoint, one avoids restricting the search space. This trade-off between providing useful astrodynamics insights and potentially restricting the search space would be interesting to explore further in the future.



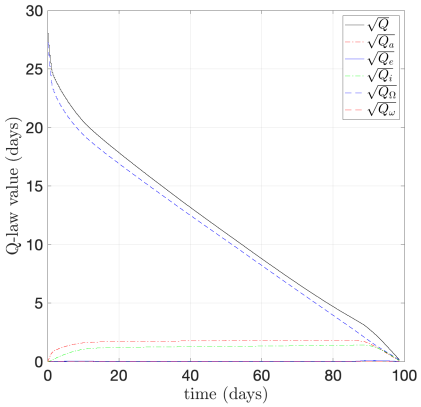
(a) Orbital Parameters



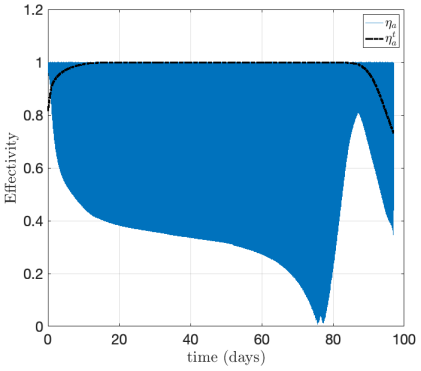
(b) Trajectory



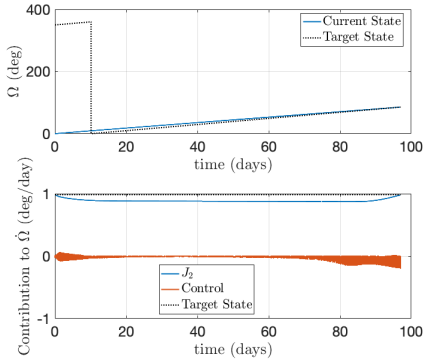
(c) Weights



(d) $\sqrt{Q}$ Contributions



(e) η_a and η_a^t



(f) Influence of J_2

Figure 4.11: Mass-optimal (< 100 days) Sun-synchronous transfers for a RL Q-Jac controller.

4.5 Discussion and Summary

This chapter investigates the robustness of the Reinforced Lyapunov Controller subject to perturbing accelerations and eclipse effects. The combination of the RL framework and the Lyapunov control provides a closed-loop control law which enables it to compute the control given just the current and target state. Firstly, can an RL agent trained without

perturbations perform in the presence of perturbations? Secondly, can the performance of the RL agent be improved by including perturbations in the training? Finally, is the approach able to learn something about the existence and exploit the perturbations?

In an attempt to address these objectives, the GTO-GEO and LEO-GEO transfers were considered with the J_2 and 3^{rd}-body perturbations. In both cases, the Reinforced Lyapunov Controller is robust to perturbing accelerations. Training with these and eclipse effects was not an issue for the closed-loop nature of the approach and a strong degree of optimality was retained. However, it appears the training process struggles due to the difficulty discerning stochastic actions and exploration from perturbing accelerations during training.

A novel cone-clock approach is introduced to allow a greater degree of freedom to the control direction and enable the controller to exploit the existence of the perturbations. This shows potential, improving the performance in several different scenarios. However, the overlap between the weights $\boldsymbol{W}$ and the angles α, β allows multiple learning paths to result in the same solution, over-complicating the learning process. In addition, the perturbing accelerations considered do not affect the Keplerian transfer significantly.

Thus, a new SSO transfer scenario is investigated which is heavily impacted by the J_2 perturbation. In this case the ability for the Reinforced Lyapunov Controller to exploit the J_2 perturbation is more apparent and the impact of the cone-clock is visible across all simulations. The controller learns to utilise the differing drift rates in $\dot{\Omega}$ at different altitudes to aid convergence to the target orbit. It is clear the cone-clock implementation can improve the performance of a nominal Lyapunov controller (without the RL framework), as seen in the PSO benchmark cases.

Results in this chapter show the demonstrated controller is thus robust to perturbations, and the RL architecture is able to handle the dynamics that are unknown to the underlying Lyapunov controller. Future work can investigate more heavily perturbed environments - such as the GTO to LPO spiral transfer considered in Chapter 7. It is noted here that whilst the cone-clock approach is developed in the presence of perturbing accelerations, it is not restricted to such environments and can be used even when the Lyapunov function has full knowledge of the dynamics. Perhaps this could be investigated more thoroughly for a much simpler problem even outside of astrodynamics, such as an inverted pendulum [118].

Chapter 5

Trajectory Design in the presence of Stochastic Errors

In this chapter the techniques developed and tested in Chapter 3, will be investigated in the presence of stochastic errors. The main purpose is to understand how the Reinforced Lyapunov Controller performs in stochastic environments, stressing the closed-loop nature of the control and the potential for on-board guidance applications. This utilises the novel cone-clock angle approach introduced in the previous chapter. Y. Hashida and A. Turconi provided very useful insights and information on the appropriate representation of uncertainties and errors associated with on-board spacecraft guidance. A similar yet simplified analysis was done by H. Holt as part of the Volatile Mineralogy Mapping Orbiter (VMMO) Mission Analysis and Design Phase A study [44].

5.1 Introduction

In many ways, trajectory design and on-board guidance often have conflicting objectives. Trajectory design focuses on optimality, whilst on-board guidance on robustness and stability. The former often makes assumptions on the dynamical model and idealisations on spacecraft model. Taking the control history provided by optimal control solutions and implementing it on-board a spacecraft is a major challenge. Key aspects such as biases, dynamical approximations, engine idealisations and operational constraints can limit the possibility of replicating an optimal trajectory and tracking it using autonomous guidance methods. As such, this chapter attempts to address the robustness of the approach developed so far to a non-idealistic, stochastic and uncertain environment.

One of the major advantages of Lyapunov-based controllers is their closed-loop nature. They only require the current and target states to compute the control direction, making them very attractive for on-board autonomous use. By design this is maintained in the RL formulation. In addition, RL is known for its performance in unfamiliar environments. This motivates the use of a model-free RL algorithm, which learns through experience in the environment. As the agent experiences stochastic behaviour during training process, it has experience of unmodelled dynamics and disturbances.

The motivation here is two-fold. Firstly, can a Reinforced Lyapunov Controller trained without stochastic errors perform in the presence of stochastic errors? In other words, can it converge to the target orbit in an optimal matter despite uncertainties? This in-

herently tests the closed-loop nature of the approach, and its optimality away from the nominal trajectory. Secondly, is there potential to improve the Reinforced Lyapunov Controller if the stochastic errors are included in the training process?

Two approaches are presented. The first involves techniques developed during the VMMO Mission Analysis and Design Phase A study [44], where the control vector is kept constant for a given time-interval and errors are added at these regular time-intervals. Using this *fixed control approach*, the Reinforced Lyapunov Controller presented in Chapter 3 is subjected to orbit insertion (OI), orbit determination (OD) and execution (EX) errors (i.e. thruster misalignment and thrust magnitude errors).

Secondly, a novel error-interpolation approach is presented to enable a *free control approach*. In this fashion, the errors can be included during the RL training process in a computationally efficient yet representative manner. As a result, the second objective can be thoroughly investigated.

This chapter is structured as follows. First, the type and magnitude of OI, OD and EX errors are presented. Next, the implementation and results for the *fixed control approach* are presented in Section 5.3, followed by the error-interpolation and *free control approach* in Sections 5.4 and 5.5. In both cases results for time-optimal and mass-optimal GTO-GEO and LEO-GEO transfer are presented. Unlike in Chapter 4, the dynamics are Keplerian. However, the novel cone-clock implementation is also included here to investigate its robustness to uncertain environments.

5.2 Orbit Insertion, Orbit Determination and Execution Errors

In order to assess the impact of OI, OD, and EX errors on the Reinforced Lyapunov Controller, several 1000-sample MC simulation were initiated. The errors modelled in the simulations are assumed to be:

- OI: zero-mean Gaussian errors with a $1\,\sigma = 10$ km and $1\,\sigma = 1$ m/s spherical standard deviations in the position and velocity components of the spacecraft initial state;
- OD: zero-mean Gaussian errors with $1\,\sigma = 100$ m (position), and $1\,\sigma = 10$ cm/s (velocity) standard deviation;
- Thrust magnitude errors: random noise added to the nominal thrust value. The stochastic component follows a Gaussian distribution with zero-mean, $3\,\sigma = 5$ % of the nominal thrust value;
- Thrust misalignment: uniformly distributed biases of $\pm 10°$ in elevation and $\pm 180°$ in the azimuth angles of the thrust vector. These biases are kept constant throughout the transfer and augmented with a zero-mean stochastic component of $3\,\sigma = 2°$ in elevation and $3\,\sigma = 20°$ in azimuth at each thrust calculation epoch.

OI errors simulate the potential of a launcher malfunction, and could also be used to access the domain of states over which the controller remains optimal. As Lyapunov control laws require the current and target state in order to compute a control, knowledge of the spacecraft's state needs to be provided. The difference between the estimate and true state is known as the OD error. Finally, once a control acceleration vector is determined, it needs to be implemented by the on-board propulsion system. The thrust magnitude errors and thrust misalignment errors are used to replicate any error in this step, and are known collectively as EX errors. The choice of error magnitudes is pessimistic in order to assess the robustness of the Reinforced Lyapunov Controller under extreme circumstances.

The errors are implemented as follows. Starting from the nominal initial orbit at periapsis, OI errors are added to determine the true initial state of the spacecraft. OD errors are simulated next in order to create an estimate for the current state of the spacecraft at the beginning of a "guidance loop". Here, the best estimate of the spacecraft state is passed to the control function for determining the nominal control direction $\boldsymbol{u}$. A δu is added to the magnitude and the thrust direction is also modified. Here, a right-handed frame is defined with respect to $\boldsymbol{u}$ and the angular momentum $\boldsymbol{h}$ of the osculating orbit as follows:

$$\begin{bmatrix} \hat{\boldsymbol{e}}_1 & \hat{\boldsymbol{e}}_2 & \hat{\boldsymbol{e}}_3 \end{bmatrix} = \begin{bmatrix} \dfrac{\boldsymbol{u}}{\|\boldsymbol{u}\|} & \dfrac{\boldsymbol{h}\times\boldsymbol{u}}{\|\boldsymbol{h}\times\boldsymbol{u}\|} & \dfrac{\hat{\boldsymbol{e}}_1\times\hat{\boldsymbol{e}}_2}{\|\hat{\boldsymbol{e}}_1\times\hat{\boldsymbol{e}}_2\|} \end{bmatrix}. \tag{5.1}$$

The elevation ϕ_e and azimuth ϕ_a angles are defined in this local reference frame such that the actual thrust direction is

$$\boldsymbol{u}^* = (\|\boldsymbol{u}\| + \delta u) \begin{bmatrix} \cos\phi_e \\ \sin\phi_e \cos\phi_a \\ \sin\phi_e \sin\phi_a \end{bmatrix}. \tag{5.2}$$

It follows that an error on the elevation angle can have a larger impact on the actual direction of thrust. This is actually defined in the same way as the cone-clock approach presented in Section 4.3, however it is aligned with $\hat{\boldsymbol{u}}$ instead of $-\hat{\boldsymbol{p}}$, in case the two do not overlap. Whilst it can appear counter-intuitive, any error in azimuth is only significant if accompanied by a large elevation error. Hence, the error in azimuth spans the full 360° whilst the elevation error does not.

5.3 Fixed-control Simulations

To begin with, the techniques developed in Chapter 3 are subjected to these stochastic errors at regular intervals in time, with the control vector remaining constant. That is, these errors are added every 1 minute, during which the control direction and magnitude are kept constant. This frequency is selected to introduce issues for the controller.

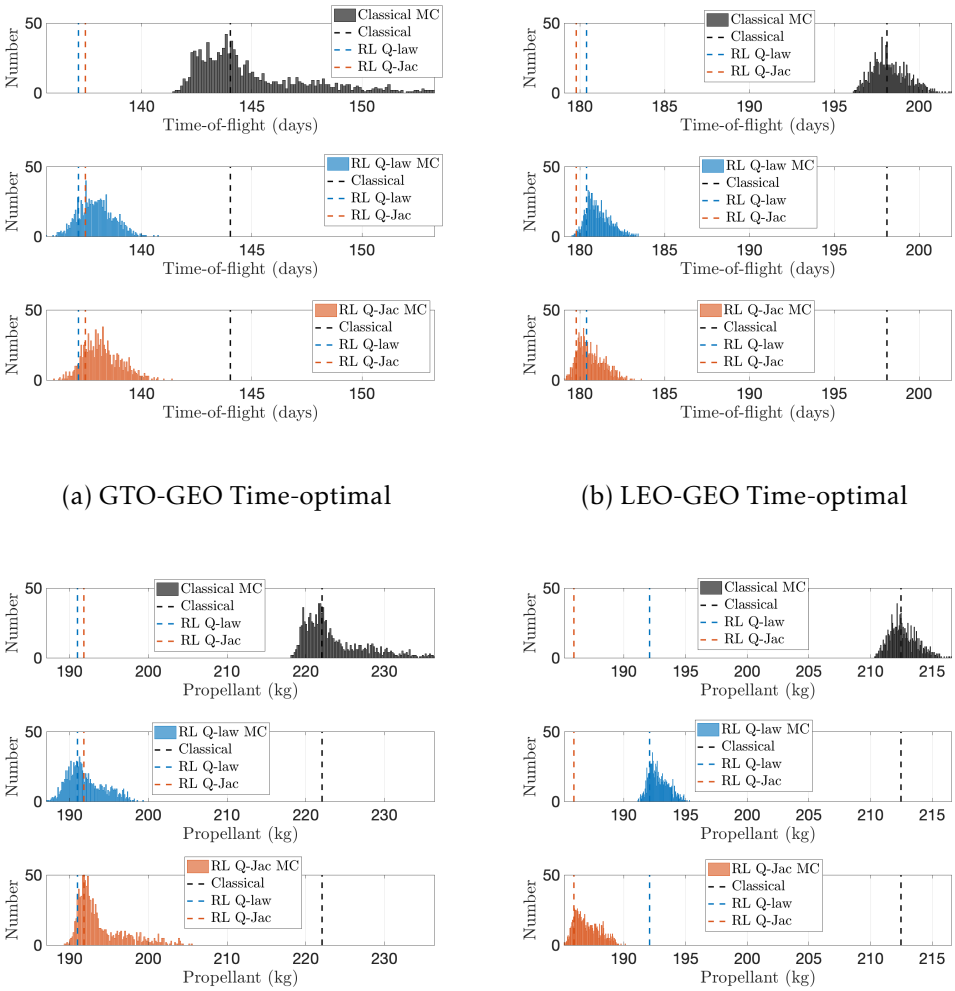
Reducing this frequency reduces both the mean and standard deviation of the objective function (either time-of-flight or propellant mass) in the resulting MC simulations. The RL Q-law and RL Q-Jac controllers are compared, using the classical Q-law as a benchmark to help indicate the effect of the error set. All three controllers experience the same realisation of the error set, although the different transfer scenarios use different realisations. Both RL approaches are pre-trained in Keplerian dynamics, and as such are deployed in identical fashion to the classical Q-law, with the actor network ensuring the state-weight dependence whilst also enabling closed-loop control.

Figure 5.1 shows histograms of the results from these MC simulations. The distribution of the 1000 MC samples is shown against the time-of-flight for the time-optimal transfers and the propellant mass for the mass-optimal ones. In addition, the vertical dashed lines indicate the nominal performance of the controller when no stochastic errors are present. It is clear that the optimality offered by the RL approaches is retained for both the time and mass-optimal transfers. The results highlight the closed-loop nature of this approach, and that it functions away from the nominal trajectory. Even with navigation errors and thruster misalignment, the inherent advantages of using a Lyapunov-based controller are retained. In Table 5.1 it is also clear that the standard deviation of the runs, when subject to identical errors, is smaller for the RL approaches as opposed to the the classical Q-law. Finally, none of the simulated RL approaches under-perform compared to the classical Q-law.

Table 5.1: MC Simulations with stochastic disturbances for the GTO-GEO and LEO-GEO transfers. Nominal results are shown for comparison. Grey highlights indicate the RL Q-Jac solutions.

	Transfer	Method	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile
Time (days)	GTO-GEO	Classical	144.03	144.99	2.55	141.88	144.12	152.73
		RL Q-law	137.14	137.94	0.78	136.29	137.91	139.87
		RL Q-Jac	137.45	138.21	0.84	136.52	138.11	140.28
	LEO-GEO	Classical	198.32	198.27	1.02	196.38	198.12	200.86
		RL Q-law	180.38	181.08	0.72	179.82	180.94	183.07
		RL Q-Jac	179.77	180.61	0.77	179.28	180.45	182.64
	Transfer	Method	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile
Mass (kg)	GTO-GEO	Classical	222.06	223.56	3.93	218.75	222.21	235.48
		RL Q-law	191.01	192.01	2.32	187.97	191.49	197.56
		RL Q-Jac	191.82	193.82	3.12	190.20	192.62	203.89
	LEO-GEO	Classical	212.70	212.64	1.11	210.59	212.48	215.45
		RL Q-law	192.10	192.77	0.78	191.36	192.64	194.83
		RL Q-Jac	185.93	187.02	1.01	185.40	186.81	189.32

The distributions are not Gaussian, with a larger tail towards higher cost values in almost all cases. Hence, the standard deviation provided in the table might give a misleading representation of the true distribution. The 1^{st}, median and 99^{th} percentiles are also given. As expected, the mean and median values for all simulations are worse than the nominal results. However, no failures are observed and in all cases the 1^{st} percentile improves on the nominal result. This is likely due to the OI errors, which can easily make



(a) GTO-GEO Time-optimal

(b) LEO-GEO Time-optimal

(c) GTO-GEO Mass-optimal

(d) LEO-GEO Mass-optimal

Figure 5.1: MC Simulations with stochastic disturbances for the GTO-GEO and LEO-GEO transfers

the transfer physically shorter. The RL Q-Jac approach has a slightly larger standard deviation than the RL Q-law approach, likely due to the fixed control implementation and its impact on the Jacobian contribution to the Lyapunov function, which is trained at a different frequency to the error implementation - see the discussion in Section 3.3.1.3.

Note that the constant control direction and magnitude might be expected to affect the performance, but the results suggest this has little impact for the selected time-step. Including these restrictions and uncertainties in the training process could further extend the validity of this approach, and as such is investigated in the next section.

5.4 Interpolating Errors

In the previous section the robustness of the controller subject to OI, OD and thruster EX errors with a fixed control was investigated. As a result of the previous investigations, it

is of interest to explore whether these errors can be included during the training process and to understand whether this improves the robustness of the controller, or reduce the optimality. Whilst the fixed control approach has its merits, it is quite slow and the zero-error case does not match the default RL approach seen in Chapter 3 due to the 1-minute fixed control vector. In addition, the frequency at which the errors are introduced is fixed to the control acceleration vector. This prevents investigations from decoupling to impact of the errors and the fixed control direction.

As such this section sought to develop a method for implementing the errors during the training process in a computationally efficient manner whilst still remaining representative of the types of uncertainty and error a controller might experience on-board. The key development in this section is the *free control* acceleration vector. The control $\boldsymbol{u}$ is free to evolve as the *ode* integrator and Lyapunov controller see fit. In order to enable this freedom, whilst also ensuring computational efficiency, for-loops with 1 minute intervals are not feasible for 200 day transfers. An approach where the errors are interpolated using cubic splines is proposed. A realisation of the uncertainty set is generated *a priori* at an interval δt using the same values as in Section 5.2. This includes OD and EX errors, whilst the OI error is only required for the initial state.

This results in a series of points at δt intervals for a predetermined duration, always longer than the projected transfer duration. These points are used to create a piecewise polynomial of cubic order to represent these errors at intermediate time-steps. Whilst this is not intended to replicate a physical evolution of the errors, it allows the control to vary at the request of the numerical integration method, which is a variable time-step approach (*ode113*). Alternative methods such as the exponential decay of thruster errors (see Tapley *et al.* [119]) were considered. However, these require a series of parameters to be determined to replicate a physical spacecraft system, which was not the aim of this work. Instead, the focus was on subjecting the controller to uncertainties and to understand its robustness under pessimistic scenarios.

The errors are introduced by passing the piecewise polynomial inside the integration and evaluating the error contribution at any t. A demonstration of the various errors is shown in Fig. 5.2. The blue cross marks indicate a particular realisation of the errors for a $\delta t = 10$ days, and the orange curve the spline interpolation of those points. By storing the coefficients of the piecewise polynomial interpolation, one can pass these inside the integration process and evaluate the error at any instance in time.

It is worth noting this means the frequency of the *a priori* errors is not as important as when using a fixed-control approach, as the OD, control EX errors are added continuously throughout the transfer. What it does affect is the potential for a dramatic change in the applied error. Here, a 1-hour interval is used, allowing the control to vary inside the integration and prevent the RL training from becoming too computationally demanding. The initial distributions are Gaussian, however, due to the bias and interpolation this results in a form of coloured noise [119]. In this instance coloured noise refers to the

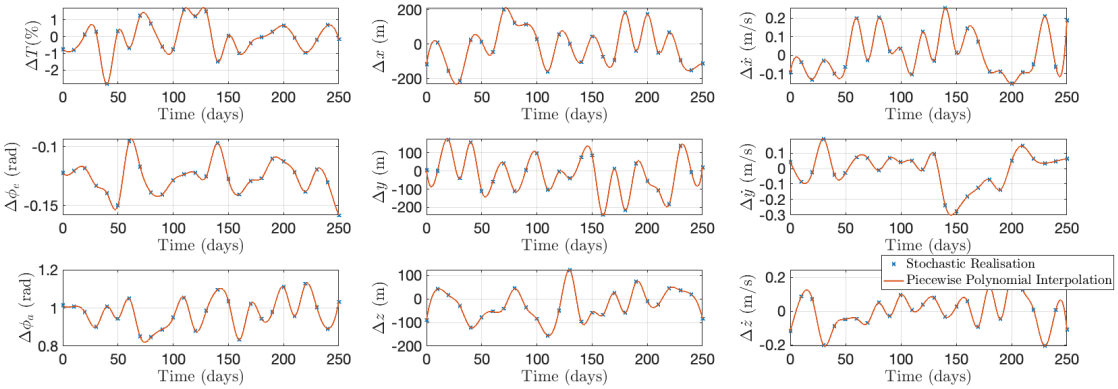


Figure 5.2: Example of interpolating a set of stochastic errors for use inside a variable step integrator. The blue cross marks indicate a particular realisation of the errors for $\delta t = 10$ days, and the orange curve the spline interpolation of those points. A bias can be seen for both ϕ_a and ϕ_e.

correlation between noise at one time step to the next. White noise has no correlation in time. However, due to the cubic spline interpolation used throughout these simulations, there is correlation in the process noise between one time step and the next.

Algorithm 4 shows how the RL training is done in the presence of these stochastic errors. The noticeable difference between previous RL training implementations is the update strategy. Previously, a batch of N + 2 trajectories are generated. N trajectories involve a stochastic policy using the current best policy $\tau(\pi_{\theta_k})$. Two deterministic trajectories are computed, one using the current parameters $\tau(\theta_k)$ and the second using the new policy $\tau(\theta)$. The batch of trajectories provides the exploration around the current policy and is used to update the new θ. The update $\theta_k \leftarrow \theta$ occurs if and only if $\tau(\theta)$ outperforms $\tau(\theta_k)$. See Section 3.3.2 for more details.

Due to the added stochasticity from the inclusion of the errors, using the same learning procedure would not be representative of a controller trained in the presence of stochastic errors. Whilst it is true the errors can easily be introduced to the batch of $\tau(\pi_{\theta_k})$ trajectories, it remains unclear how the update be handled. If $\tau(\theta)$ and $\tau(\theta_k)$ are left deterministic, then they will only improve the deterministic performance of the policy, and not its performance in the presence of stochastic errors. If stochastic errors are included instead, then one cannot include a single realisation of the errors, but rather a batch of errors is required. Essentially, a mini-MC simulation is required such that one can use batch statistics to inform any improvement in performance.

Using this approach, the number of trajectories generated per iteration during training increases from N + 2 to $3N$. The first batch N uses a stochastic policy and stochastic errors. The second and third batch then use a deterministic policy (using θ_k and θ respectfully) but also include stochastic errors. In all three batches the realisation of the errors is the same. Thus, only N realisations of the uncertainty set are required. The median of the two deterministic policy batches is used to determine if and when the update $\theta_k \leftarrow \theta$ is reached. Due to computational limitations, it is not feasible to use a

batch size N = 1000, as might be suggested from the analysis in Section 5.3, as this results in $3N = 3000$ trajectories per iteration. However, it is of interest to increase the batch size compared to the deterministic and Keplerian transfers. Hence, an intermediate value of N = 50 was used. As both deterministic policies receive the same realisation of the uncertainty set, this batch size has less influence than in the final, post-training MC simulations, where an understanding of the true distribution is sought. It is also worth noting that within one iteration the set of errors is fixed, whilst a new realisation of errors is used every iteration. This ensures the controller experiences a large variety of error combinations, and prevents the training of a controller on a particular set of errors only.

Algorithm 4 Pseudocode: Training with Stochastic Errors

Set initial state $\boldsymbol{X}_0$ and target state $\boldsymbol{X}_T$ ▷ Initialise Transfer
Set random parameters θ_k ▷ Initialise RL
while Training **do**
    Realise stochastic errors at δt intervals (N times) ▷ New errors every iteration
    Create piecewise polynomial Interpolation ▷ Ready to pass to integration
    for Batch $3 \times N$ **do** ▷ N stochastic $\boldsymbol{W}$+errors and $2 \times N$ deterministic... ▷ ...$\boldsymbol{W}$ and stochastic Errors
        $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}_0 + \Delta\boldsymbol{X}_0$ ▷ OI Errors
        while Propagating Transfer **do** ▷ Computing transfer trajectory
            Use piecewise polynomial to evaluate $\Delta\boldsymbol{X}_{est}$ and $\Delta\boldsymbol{u}$
            $\boldsymbol{X}_{est} \leftarrow \boldsymbol{X} + \Delta\boldsymbol{X}_{est}$ ▷ OD Errors
            $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}_{est}, \theta_k)$
            $-\hat{\boldsymbol{p}} \leftarrow \text{Lyapunov}(\boldsymbol{X}_{est}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}_{est}, \boldsymbol{u}, \boldsymbol{0}))$
            $\boldsymbol{u} \leftarrow \text{Cone-clock}(-\hat{\boldsymbol{p}}, \boldsymbol{0})$ ▷ Using Eqs. (4.8) and (4.9)
            $\boldsymbol{u} \leftarrow \boldsymbol{u} + \Delta\boldsymbol{u}$ ▷ EX Errors
            $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0})$
    Update θ_k if Median($C(\theta)$) <Median($C(\theta_k)$) ▷ Stochastic Update
while Deploying **do** ▷ Deploy learnt policy
    while Propagating Transfer **do** ▷ Computing transfer trajectory
        $\boldsymbol{W} \leftarrow \text{ActorNetwork}(\boldsymbol{X}, \theta_k)$
        $-\hat{\boldsymbol{p}} \leftarrow \text{Lyapunov}(\boldsymbol{X}, \boldsymbol{X}_T, \boldsymbol{W}, f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0}))$
        $\boldsymbol{u} \leftarrow \text{Cone-clock}(-\hat{\boldsymbol{p}}, \boldsymbol{0})$ ▷ Using Eqs. (4.8) and (4.9)
        $\boldsymbol{X} \leftarrow f(\boldsymbol{X}, \boldsymbol{u}, \boldsymbol{0})$

5.5 Free-control Simulations

In this section, the results for the Reinforced Lyapunov Controller trained and deployed in the presence of stochastic errors are presented. These are simulated for the same GTO-GEO and LEO-GEO transfers presented in Chapter 3. Both the Q-law, and a basic Lyapunov control law, were tested. The results for the Q-law and presented and discussed here, whilst the basic Lyapunov control law results are given in Appendix A.3. This is to emphasise the approach is flexible to the controller adopted, and can improve

the performance in both cases. However, it must also be noted the underlying controller can limit the optimality of the results, and this choice should be taken into account.

Both time- and mass-optimal simulations for GTO-GEO and LEO-GEO transfers are presented. In each case, results are given for OD and EX errors (denoted as OD+EX), and for OI, OD and EX errors (denoted as OI, OD+EX). This distinction is made to separate the errors applied at the beginning of the transfer (the OI ones) to those that are introduced throughout the transfer duration (the OD and EX ones). Finally, for each case, two different training methods are compared. Firstly, a controller trained in a fully deterministic environment with no experience of the errors, hereby denoted by deterministic training (DT). Secondly, the stochastic batch training (SBT) approach is described in Section 5.4. DT allows a comparison with the fixed control approach in Section 5.3. However, in Section 5.3 the control direction remained fixed for 1 minute intervals, whilst for the DT simulations, no restrictions are placed on the control direction. DT explores the robustness and closed-loop nature of the Reinforced Lyapunov Controller, and the SBT can explore any potential improvements training with errors included can achieve. An episode is deemed to have failed if it is unable to converge to the target orbit within the maximum integration time, here set to 300 days. This can either be due to poor performance in terms of optimality, or if there is control law chattering within the *ode113* integration that prevents convergence. In the unlikely event of a spacecraft crashing with the central body or running out of propellant, this is also accounted for.

Table 5.2: Time-optimal MC Simulations with stochastic disturbances for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT. Grey highlights indicate the RL Q-Jac SBT solutions.

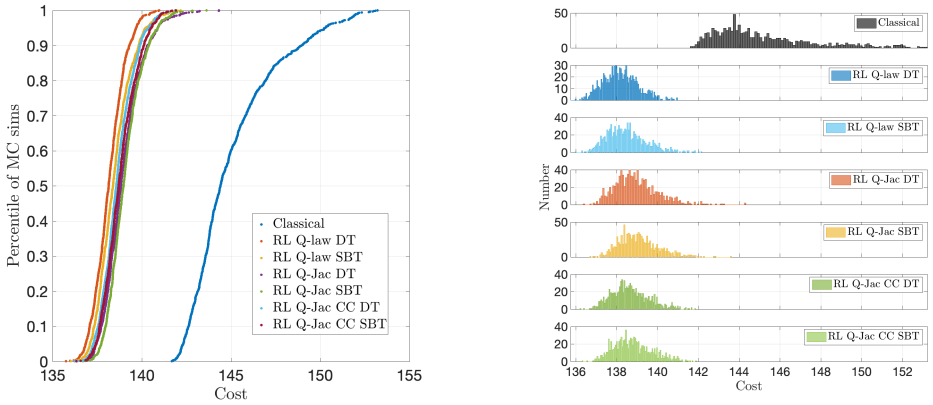
Controller	Errors	Method	Training	Nominal (days)	Mean (days)	σ (days)	1^{st} Percentile (days)	Median (days)	99^{th} Percentile (days)	Failures
Q-law	OD+EX	Classical	-	144.03	144.97	2.433	142.14	144.14	151.86	0
		RL Q-law	DT	137.29	138.19	0.647	137.34	138.01	139.90	0
			SBT	137.25	138.16	0.677	137.30	137.99	140.09	0
		RL Q-Jac	DT	137.70	138.96	1.062	137.74	138.72	143.10	0
			SBT	137.84	138.84	0.864	137.85	138.58	141.55	0
		RL Q-Jac+CC	DT	137.65	138.66	0.816	137.69	138.44	141.31	0
			SBT	137.82	138.88	0.884	137.80	138.62	141.52	0
	OI+OD+EX	Classical	-	144.03	145.11	2.430	141.98	144.41	152.10	0
		RL Q-law	DT	137.29	138.18	0.853	136.47	138.14	140.45	0
			SBT	137.54	138.48	0.981	136.67	138.39	141.41	0
		RL Q-Jac	DT	137.70	138.96	1.071	137.16	138.83	142.28	0
			SBT	138.02	139.07	0.964	137.29	138.98	141.77	0
		RL Q-Jac+CC	DT	137.65	138.66	0.925	136.92	138.57	141.09	0
			SBT	137.78	138.79	0.945	137.02	138.70	141.12	0

Tables 5.2 and 5.3 show the results for the time- and mass-optimal simulations for the GTO-GEO transfer. To begin with, there is a subtle difference between the fixed and free control results. The rows RL Q-law and RL Q-Jac with DT allow a comparison with the fixed control with 1-minute intervals to the interpolated 1-hour errors. In the time-optimal case the mean values differ by only -0.24 days for the RL Q-law and -0.75 days for the RL Q-Jac. In the mass-optimal case this is +1.09 kg and +2.23 kg. This suggests overall the RL Q-Jac, and by extension the RL Q-Jac+CC, is more heavily influenced by

Table 5.3: Mass-optimal MC Simulations with stochastic disturbances for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT. Grey highlights indicate the RL Q-Jac SBT solutions.

Controller	Errors	Method	Training	Nominal (kg)	Mean (kg)	σ (kg)	1st Percentile (kg)	Median (kg)	99th Percentile (kg)	Failures
Q-law	OD+EX	Classical	-	222.06	223.92	4.339	219.28	222.32	238.65	0
		RL Q-law	DT	190.78	191.01	2.127	188.01	190.47	197.26	0
			SBT	195.03	195.13	3.691	191.47	193.60	206.72	0
		RL Q-Jac	DT	192.13	191.55	2.183	189.20	190.68	197.72	0
			SBT	192.53	192.18	2.600	189.63	191.05	200.20	0
		RL Q-Jac+CC	DT	192.61	192.68	2.345	190.22	191.64	199.69	0
			SBT	193.36	193.21	2.949	190.02	192.04	202.29	0
	OI+OD+EX	Classical	-	222.06	223.67	4.157	218.64	222.32	237.70	0
		RL Q-law	DT	190.78	190.92	2.500	186.70	190.35	198.18	0
			SBT	193.97	193.53	3.414	189.11	192.27	204.28	0
		RL Q-Jac	DT	192.13	191.59	2.480	188.25	190.83	199.06	0
			SBT	192.33	191.65	2.693	188.06	190.84	199.40	0
		RL Q-Jac+CC	DT	192.61	192.73	2.660	189.32	191.79	200.94	0
			SBT	192.41	192.05	2.970	187.90	191.36	201.00	0

the interpolated errors. However, in general the results are comparable, and if anything the interpolated errors appear more challenging for the control law to handle, perhaps because control law chatter is increasingly likely compared to the fixed control case.

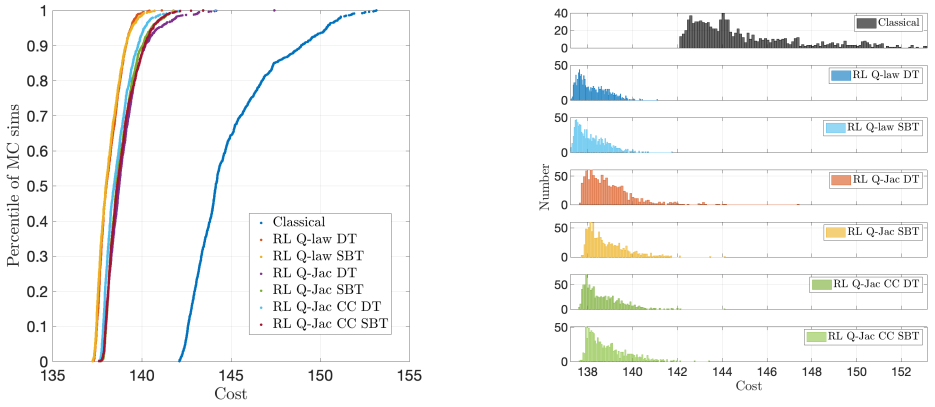


(a) Sorted Costs

(b) Histogram

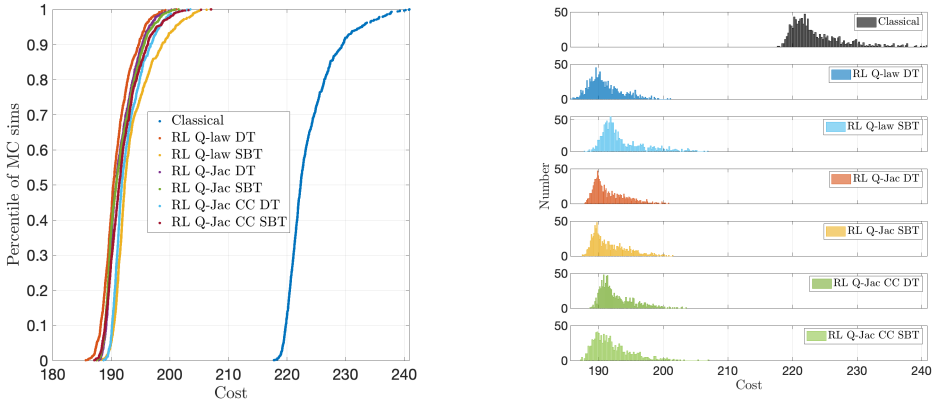
Figure 5.3: Time-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

Figures 5.3 and 5.4 display the results shown in Table 5.2 for the time-optimal GTO-GEO simulations. A sorted set of the MC simulations are shown to allow an easy comparison of the different percentiles for the various controllers. In addition, histograms are shown for the six different RL controllers along with the classical Q-law for comparison. Note in each figure the realisation of the uncertainty set is the same, so every controller experienced an identical set of errors. Clearly the results are not Gaussian, but have a tail towards higher cost values. There is little noticeable difference in the optimality and spread of the six controllers. Including OI errors clearly increases the size of the tail, something which is reflected in the mean and standard deviation values, but much more visible in the plots and median values. In addition, the minimum or best results occur



(a) Sorted Costs (b) Histogram

Figure 5.4: Time-optimal MC Simulations with stochastic disturbances (OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

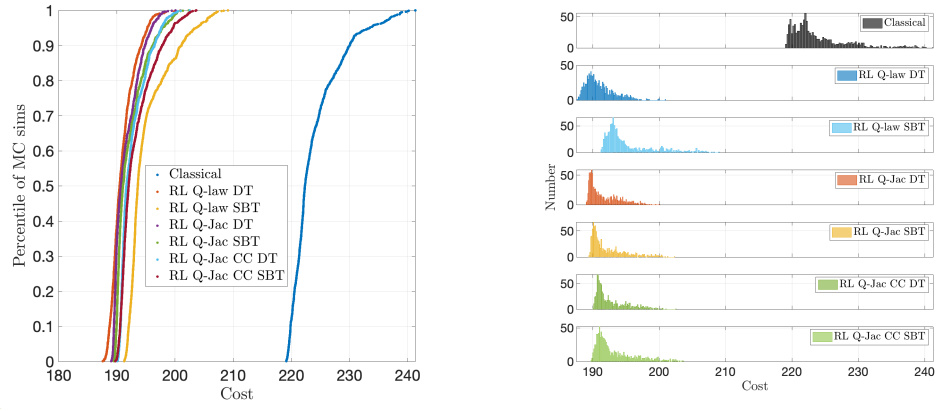


(a) Sorted Costs (b) Histogram

Figure 5.5: Mass-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

when OI errors are included, most likely because it shortens the physical length of the transfer. However, the worst results (indicated by the 99^{th} percentile) are more similar, suggesting the OD+EX errors are most likely to cause the controllers difficulty for these GTO-GEO time-optimal simulations.

Figures 5.5 and 5.6 display the results shown in Table 5.3 for the mass-optimal GTO-GEO simulations. As in the time-optimal case, the optimality gain from the RL simulations over the classical Q-law is preserved when errors are introduced. No failures are observed. Again, the results are clustered around the nominal, zero-error simulation, but have a longer tail to higher cost values. Interestingly, for the time-optimal simulations, the mean and median results are always worse than the nominal case. However, in the mass-optimal scenarios this is no longer true, with all OD+EX and OI, OD+EX simulations having a more optimal median value than the nominal case. However, SBT



(a) Sorted Costs

(b) Histogram

Figure 5.6: Mass-optimal MC Simulations with stochastic disturbances (OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

demonstrates in worse performance compared to DT. This suggests the added stochasticity complicates the learning process too much. This is observed for the mass-optimal case but not the time-optimal case, suggesting it is related to the effectivity term. This is used to estimate how effective thrusting at a particular location along the orbit is. As such, an error in the OD, particularly on the true anomaly, would heavily impact the cut-off location and hence the efficiency of the thrust during the transfer. This does not occur in the time-optimal case and likely explains why the SBT process for the mass-optimal case is so poor: it is unable to learn the effectivity threshold well with all the stochasticities.

Table 5.4: Time-optimal MC Simulations with stochastic disturbances for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT. Grey highlights indicate the RL Q-Jac SBT solutions.

Controller	Errors	Method	Training	Nominal (days)	Mean (days)	σ (days)	1st Percentile (days)	Median (days)	99th Percentile (days)	Failures
Q-law	OD+EX	Classical	-	198.32	198.35	0.629	197.44	198.22	200.23	0
		RL Q-law	DT	181.89	182.93	0.795	181.76	182.76	NaN	344
			SBT	182.92	184.02	0.826	182.89	183.80	NaN	177
		RL Q-Jac	DT	180.44	182.68	3.834	180.45	181.88	NaN	20
			SBT	180.21	181.68	0.930	180.28	181.51	184.06	0
		RL Q-Jac+CC	DT	180.50	183.99	4.675	180.53	182.61	NaN	13
			SBT	180.61	181.50	0.752	180.45	181.32	183.53	0
	OI+OD+EX	Classical	-	198.32	198.37	1.010	196.56	198.25	201.03	0
		RL Q-law	DT	181.89	182.61	0.744	181.46	182.45	NaN	303
			SBT	183.27	184.20	1.022	182.72	183.93	187.00	2
		RL Q-Jac	DT	180.44	182.47	3.327	180.10	181.74	NaN	17
			SBT	180.29	181.72	1.227	180.03	181.49	185.92	0
		RL Q-Jac+CC	DT	180.50	183.54	4.082	180.24	182.40	NaN	15
			SBT	180.62	181.87	1.228	180.16	181.51	185.78	0

Tables 5.4 and 5.5 show the results for the time- and mass-optimal simulations for the LEO-GEO transfer. Comparing these free control simulations against the fixed control ones in Table 5.1, there is a notable difference in performance. In the time-optimal cases the mean values differ by −1.53 days for the RL Q-law and −1.86 days for the RL Q-Jac. In the mass-optimal case this is −1.39 kg and −2.59 kg. As before, this suggests

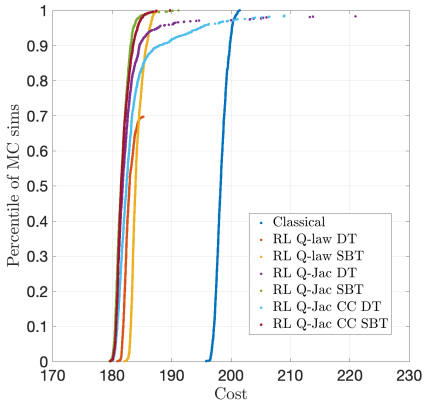
Table 5.5: Mass-optimal MC Simulations with stochastic disturbances for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT. Grey highlights indicate the RL Q-Jac SBT solutions.

Controller	Errors	Method	Training	Nominal (kg)	Mean (kg)	σ (kg)	1^{st} Percentile (kg)	Median (kg)	99^{th} Percentile (kg)	Failures
Q-law	OD+EX	Classical	-	212.70	212.68	0.692	211.72	212.53	214.96	0
		RL Q-law	DT	190.85	195.22	4.974	189.33	194.21	211.37	0
			SBT	205.88	210.32	11.639	201.47	207.97	236.02	0
		RL Q-Jac	DT	186.97	190.03	4.990	184.09	188.87	NaN	138
			SBT	206.53	206.89	4.750	198.92	206.22	216.22	0
		RL Q-Jac+CC	DT	187.45	191.94	5.681	185.72	190.62	NaN	156
			SBT	204.02	201.59	1.928	199.20	200.97	207.06	0
	OI+OD+EX	Classical	-	212.70	212.70	1.209	210.65	212.53	216.66	0
		RL Q-law	DT	190.85	194.16	4.961	188.92	192.42	210.64	0
			SBT	204.34	209.53	4.959	197.86	210.67	218.65	0
		RL Q-Jac	DT	186.97	189.61	5.512	183.68	187.86	NaN	131
			SBT	206.96	205.81	2.352	201.49	205.58	212.09	0
		RL Q-Jac+CC	DT	187.45	191.48	11.544	185.32	189.17	NaN	137
			SBT	202.19	201.56	1.157	199.48	201.38	205.03	0

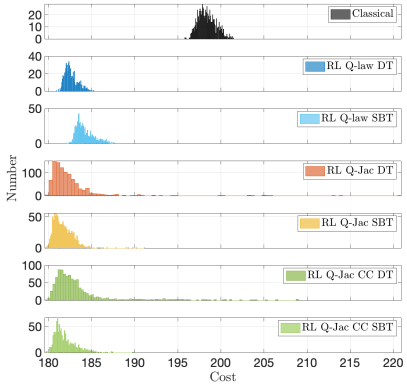
overall the RL Q-Jac, and by extension the RL Q-Jac+CC, is more heavily influenced by the interpolated errors. It appears in both time- and mass-optimal simulations the interpolated errors appear more challenging for the control law to handle.

Figures 5.7 and 5.8 display the results shown in Table 5.4 for the time-optimal LEO-GEO transfer. As above, a sorted set of the MC simulations are shown, along with histograms for each controller. Compared to the GTO-GEO transfer, there are now a sizeable number of failures occurring within several MC simulations. For example, the RL Q-law with DT has 344 failures when subjected to OD+EX errors, and 303 failures with OI, OD+EX errors. Evidently, the LEO-GEO transfer is more sensitive to the OD+EX errors than the GTO-GEO case. A possible explanation is the eccentricity value, which starts with $\Delta e \sim 0$. As was seen in Chapter 3, this needs to increase to $\Delta e \sim 0.2$ before decreasing again. Errors in the estimated state could easily change the priority of the controller and alter the behaviour such that failures occur. It is important to note that failures do not necessarily mean a crash or even an unrecoverable spacecraft. They refer to failures within the limitations of the simulation. One such possibility is if there is a chattering within the *ode113* integration that prevents convergence. Alternatively, if the total integration time is greater than a predetermined maximum time-of-flight, here set to 300 days. Figures 5.7 and 5.8 do not plot these failures, hence why some of the sorted cost curves terminate at around the 70^{th} percentile.

Unlike in the GTO-GEO case, here the SBT has a much greater improvement on the final results. Firstly, the total number of errors always decreases compared to the DT, as does the standard deviation. Interestingly, more failures are observed for the RL simulations without the Jacobian contribution. It is vital to note that the statistical information in the tables neglect the number of failures. Hence, when comparing the results, the first criteria to consider is the number of failures, as this is the key indication of robustness. It explains why the RL Q-law DT approach in OD+EX errors has a relatively good 99^{th} percentile time-of-flight but also observes 344 failures. In this sense the RL Q-Jac and

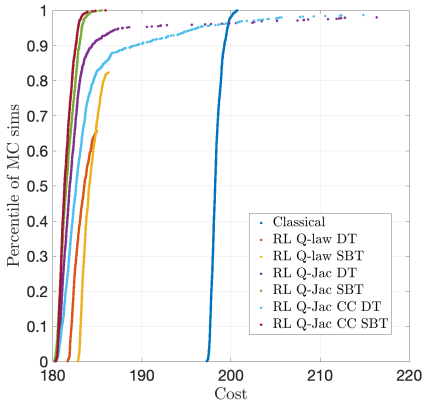


(a) Sorted Costs

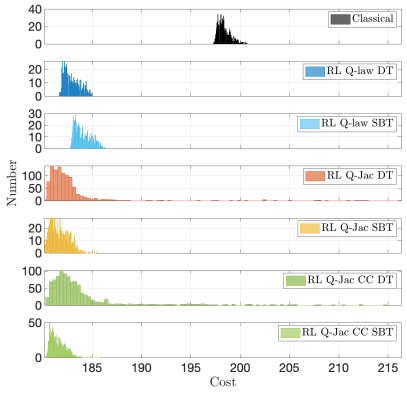


(b) Histogram

Figure 5.7: Time-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.



(a) Sorted Costs



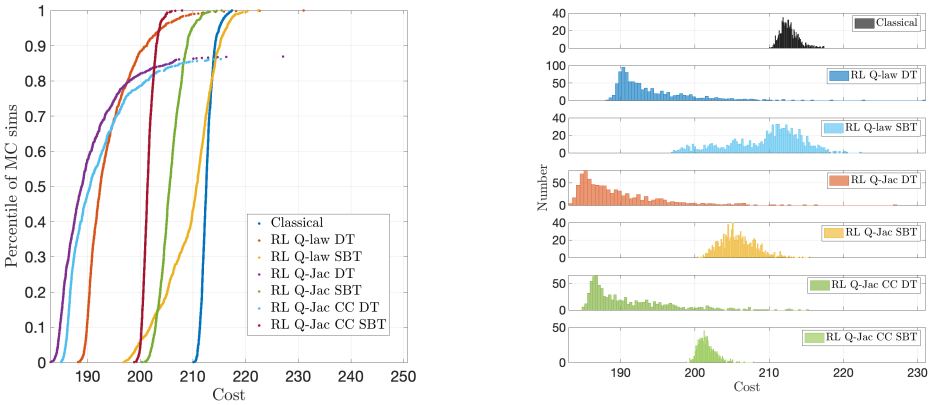
(b) Histogram

Figure 5.8: Time-optimal MC Simulations with stochastic disturbances (OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

RL Q-Jac+CC results are more reflective of the improvements that can be made for time-optimal transfers: namely RL Q-Jac is more robust to unknown errors (DT) and using SBT can reduce the number of errors and standard deviation of the results, demonstrating further increased robustness.

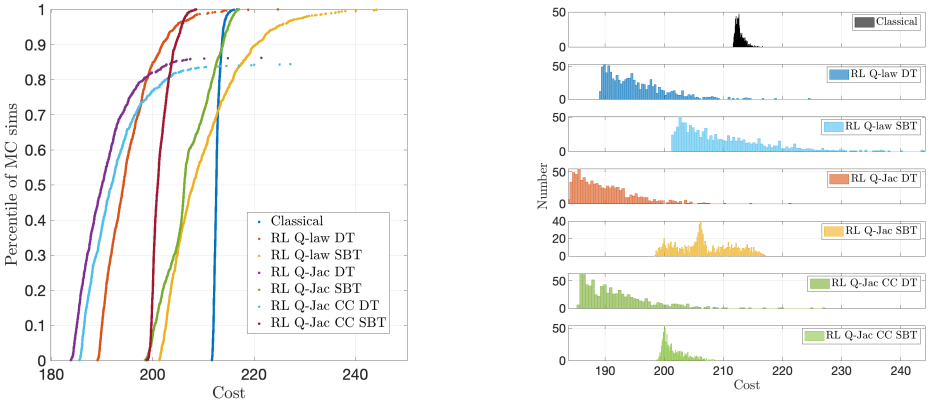
Figures 5.9 and 5.10 display the results shown in Table 5.5 for the mass-optimal LEO-GEO simulations. Interestingly, failures only occur in the RL Q-Jac with DT. These failures are all successfully removed using SBT, at the cost of optimality. As is clearly shown in the histogram plots, the SBT results in a significantly less optimal controller, but all simulations involving the RL Q-Jac have a smaller standard deviation, demonstrating the robustness to errors after SBT. The only distinction between OD+EX and OI, OD+EX simulations case be seen in the histogram plots, where the OI errors can result in a lower minimum cost. However, the OD+EX errors are clearly responsible for the longer tail

and maximum cost results. Finally, the classical Q-law, which is being used here as a benchmark to indicate the effect of the errors on a typical Lyapunov controller, demonstrates no failures and very low standard deviations. This suggests the failures resulting from the errors are either due to the increased optimality and potentially more sensitive transfer, or from the introduction of the actor network. Either way, RL Q-Jac with SBT removes all errors, achieves standard deviations on par with the classical Q-law and results in a more optimal transfers - the key result of these investigations.



(a) Sorted Costs (b) Histogram

Figure 5.9: Mass-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.



(a) Sorted Costs (b) Histogram

Figure 5.10: Mass-optimal MC Simulations with stochastic disturbances (OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (Q-law). Nominal results are shown for comparison. Training is either DT or SBT.

In order to assess these results in relation to the two objectives outlined at the start of the MC analyses, a criteria by which the controllers can be assessed needs to be determined. The primary objective of any autonomous guidance technique is robustness [120]. Thus, any approach which can reduce the number of failures within the MC population

has to be considered a success. If there are equal or no failures, then the second objective would be to remain as optimal as possible. Here this is chosen as comparing the 99^{th}-percentile of the population. Again, the motivation is to account for the simulations where the stochastic errors hamper the performance of the controller, making it more difficult for it to converge optimally. Using these criteria, the results above can be categorised as follows.

5.5.1 Orbit Insertion, Orbit Determination and Execution Errors

Across all the simulations (including the basic Lyapunov control law shown in Appendix A.3), the SBT approach improves the performance compared to the DT more often than not. When considering only those controllers which include the Jacobian term for stability, to implement the proposed Reinforced Lyapunov Controller it is best to train in the presence of stochastic errors as this will significantly improve the performance.

Considering just those results which include the Jacobian stability term, the Q-law controlled transfer results are inconclusive. However, all cases for the basic Lyapunov control law are an improvement - see Appendix A.3. This is likely because the Q-law already includes some knowledge of the orbital mechanics, whilst the basic Lyapunov controller does not, suggesting it is less robust to these stochastic errors.

Perhaps counter-intuitively, the cone-clock contribution improves the the Q-law controller more than the basic Lyapunov control law. The initial expectation would suggest the opposite behaviour. The basic Lyapunov control law has a less optimal $-\hat{\boldsymbol{p}}$ direction, and as such one might expect the cone-clock approach to free this direction to improve the performance. Theoretically the Jacobian+CC transfers should at least match the performance of the Jacobian ones. However, due to the stochastic and local nature of the learning process, along with the complexity of the problem and the number of parameters that need learning, in practice this is hard to ensure.

5.5.2 Orbit Determination and Execution Errors

The same training and MC analyses were done without the OI errors. In practice, this helps indicate the robustness of the controller to the OD and EX errors alone. The injection errors can heavily impact the objective and as such hide some of the influence of the *on-board* errors.

Across all the simulations (again including the basic Lyapunov control law shown in Appendix A.3), the SBT approach improves the performance compared to the DT more often than not, which matches the OI, OD+EX scenario. When considering only those controllers which include the Jacobian term for stability, there is less improvement than in the OI, OD+EX scenario. However, it remains true that if you want to implement the proposed Reinforced Lyapunov Controller, then it is best to train in the presence of stochastic errors.

Considering just those results which include the Jacobian stability term, the results suggest the OI errors are responsible for a large proportion of improvement when using the SBT approach. This makes sense, during DT the initial orbit remains fixed, and as such when a new initial orbit is provided, the optimality of these parameters is reduced. It is thus recommended to include OI errors in the training for potential on-board implementation.

The number of failures is comparable between the OI, OD+EX simulations and the OD+EX only simulations. This suggests the failures arise from the OD and EX errors, likely causing chattering in the control law behaviour which cannot be recovered effectively. Fortunately the trend of the SBT significantly reducing the number of failures remains in both cases.

5.6 Discussion and Summary

This chapter investigates the robustness of the Reinforced Lyapunov Controller in uncertain environments. The combination of the RL framework and the Lyapunov control provides a closed-loop control law which enables it to compute the control given just the current and target state. The motivation here is two-fold. Firstly, an RL agent trained without stochastic errors is robust in the presence of stochastic errors. This was successfully investigated for a fixed control and for a free control in this chapter. Secondly, the performance of the Reinforced Lyapunov Controller can be improved by including the such stochastic errors in the training process, which is a major novel contribution of this chapter. Investigations including OI, OD and EX errors demonstrate the flexibility of the approach. The best performance is achieved when training with the stochastic errors: the optimality is most heavily influenced by the OI errors, whilst the robustness (lack of failures) is dominated by the OD and EX errors. The key conclusion is the SBT can remove all failures for the Reinforced Lyapunov Controller with guaranteed Lyapunov stability. This is vital should this approach be considered for on-board use.

The demonstrated controller is thus robust to uncertainties in state, retaining a high degree of optimality and demonstrating the closed-loop nature of this approach. This paves the way for future work in higher-fidelity dynamics. Although out-of-scope for this work, the method offers a lot of potential for on-board use. The NN can be trained on-ground and the trained network implemented on-board to offer increased optimality along with the desired stability of a classical Lyapunov controller. This closed-loop guidance approach would be able to execute an autonomous transfer, or reconfiguration, in a close-to-optimal manner, with demonstrated robustness to uncertainties in OI, OD and EX errors.

Chapter 6

Trajectory Design for Approximating Finite-burn Manoeuvres

Throughout this PhD thesis, the performance in both modelled (Chapter 3) and unmodelled dynamics (Chapters 4 and 5) has been investigated. The base performance in Keplerian dynamics is used to investigate the optimality and stability of the approach. Then it is exposed to unmodelled dynamics in the form of perturbing accelerations, eclipse events and stochastic errors. This enabled an investigation into the robustness of the approach, the suitability for on-board use, and the potential for exploiting the unmodelled dynamics to the controllers advantage. So far this has focused on low-thrust trajectory design. In this regime, the acceleration provided by the ratio of the on-board thruster to spacecraft mass is often minimal, whilst meaningful cumulative ΔV can only be obtained if this thrust in provided for extended periods of time. However, many spacecraft applications have significantly higher thrust-to-mass ratios, often as a result of chemical propulsion systems. In this chapter, the extent to which the techniques developed so far can be modified to produce finite-burn manoeuvres is investigated.

6.1 Introduction

One of the primary uses of Lyapunov control laws is in preliminary trajectory design, and as initial guess for indirect [37] and direct methods [38]. They can be used to provide sub-optimal but rapid control histories which act as reliable initialisation procedures. Throughout the thesis a major motivating factor has been the potential for more optimal Lyapunov control solutions to help with both mission analysis and as better initialisation methods which are closer to the desired global optimum, or at least within the region of attraction. However, as discussed in Chapter 1, for non-convex problems, it is not possible to attain *a priori* whether a more optimal initial guess will lead to a more optimal solution for indirect or direct methods.

This chapter investigates the potential for the Reinforced Lyapunov Controller to design trajectories at higher thrust-to-mass ratios. The motivation here is three-fold. Firstly, many spacecraft applications have significantly higher thrust-to-mass ratios than those considered so far, often as a result of chemical propulsion systems. Investigating these is of great interest for SSTL because it would allow them to utilise this tool for current mission analysis rather than for the longer term development of low-thrust missions.

Secondly, increasing the acceleration level increases the sensitivity of the problem to the control action selected. In a similar fashion to Chapters 4 and 5, this provides an opportunity to stress the Reinforced Lyapunov Controller in an environment where the control action has a greater influence on the trajectory design. Lastly, there is growing evidence that continuous thrust trajectory design solutions can be used to solve optimal bang-bang [121] and impulsive transfer problems [73] - as discussed in Section 2.2.2.6.

In this chapter, a LEO-LEO transfer scenario is considered. This remains a many-revolution transfer, however in lower altitude environment than the GTO-GEO and LEO-GEO transfers predominantly considered thus far. Whilst they are highly important for electric orbit-raising of, for example, telecommunication satellites, there are extensive number of LEO-based spacecraft which often need to transfer orbits, for example during constellation deployment or reconfiguration. This provides an opportunity to investigate the flexibility for the approach to different trajectory design scenarios. In addition, it is an interesting industry test case which was provided by SSTL and allows comparison to a real-world scenario they have previously considered. This helps access the industry application and impact of the approach presented in this thesis.

The chapter is structured as follows. Section 6.2 discusses the suitability of Lyapunov based control laws for approximating finite-burn manoeuvres. Next, Section 6.3 details the transfer scenario, including the desired operational constraints and preliminary results highlighting the chattering issues of Lyapunov control laws. To solve this, mean orbit elements are used in Section 6.4 and this is followed by the finite-burn approximation procedure. Finally, the results for a grid search, PSO and Reinforced Lyapunov Controller are given in Section 6.5. Discussion and summary are given at the end.

6.2 Methods for Approximating Finite-burn transfers

In this section potential methods for approximating finite-burn manoeuvres are discussed. Although the intention is to use Lyapunov control laws as the basis for this chapter, it is beneficial to consider the alternative thrust-blending approaches first to provide inspiration on how to modify the Lyapunov control law for this function. Thrust-blending control laws appear well suited to approximating finite-burn manoeuvres as they are formulated to provide high rates of change in the spacecraft state at a given instance - a desirable trait for this application. As mentioned in Section 2.2.1 and Section 2.2.2.4, they utilise analytical expressions for the ΔV required to change individual orbital elements.

The cases considered in the thesis have a thrust-to-mass ratio at the beginning of the transfer of 0.000175 m/s^2 and 0.00033 m/s^2 for the GTO-GEO and LEO-GEO cases respectfully. Petropoulous *et al.* [36] introduced a common case later used by Lee *et al.* [62], where the thrust-to-mass ratio at the beginning of the transfer is an order of magnitude greater at 0.0033 m/s^2. Lococh*e et al.* [63, 115] discuss the operational strategy

for a Lyapunov-based control law for a GTO-GEO transfers. They highlight the existing strategies involving ground station communications, orbit determination, guidance and required monitoring. Whilst this is still low-thrust, the introduction of these operational constraints act as necessary interruptions to the computed control.

Lyapunov control theory produces a control direction for each point in time during the transfer. The magnitude of this control vector is defined instead by the on-board propulsion system and naturally the current mass of the spacecraft. As such, it is possible to provide high thrust-to-mass values to replicate chemical propulsion systems. The problem lies in the conventional implementation, where a continuous thrust is assumed. However, the effectivity threshold η_a^t can be used to ensure the thruster is only switched on for short periods of time. As a reminder, the effectivity is calculated using:

$$\eta_a(\dot{Q}) = \frac{\min_{\phi_\alpha,\phi_\beta}(\dot{Q})}{\min_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}))}, \tag{6.1}$$

where ϕ_α and ϕ_β are the in-plane and out-of-plane angles of the thrust vector. Again, $\min_\nu(\min_{\phi_\alpha,\phi_\beta}(\dot{Q}))$ is computed numerically by scanning through the possible ν values to find the maximum and minimum $\dot{Q}$ for the particular osculating orbit. Threshold values can be set and engine thrusting will only occur when $\eta_a > \eta_a^t$. By design this will occur at least instantaneously once per orbit. A very high value is needed to attempt to replicate finite-burn manoeuvres, such as $\eta_a^t \sim 0.95$. As η_a varies throughout the orbit, one possibility of replicating a very short burn arc is to restrict burns to regions where η_a is close to its maximum value. Whilst thrust-blending approaches might be considered more suitable, it is of interest to explore the potential for the Reinforced Lyapunov Controller to generate finite-burn transfers.

6.3 Transfer Scenario and Preliminary Results

The test case considered in this chapter was suggested by SSTL and intended to replicate the deployment of LEO-vantage 1 satellite [122]. The objective of this mission was to design a transfer from a 500 km altitude almost-circular LEO to a 1000 km LEO. This requires an inclination change of approximately 2°. One mission constraint was the need to complete the transfer within a 3-month time-frame. As such, the first burn started on 21$^{\text{st}}$ January 2018, and completed on 6$^{\text{th}}$ April 2018. Table 6.1 gives the orbital configuration used in this chapter to replicate the transfer.

A posteriori data shows an initial spacecraft mass of 169 kg, with an initial propellant mass of 37.6 kg. Although out-of-scope for this shorter investigation, the propulsion system dynamics were also provided. Telemetry data shows an initial and final tank pressure of 19.3 bar, and 6.14 bar. Using the manufacturers thrust model, this equates to a maximum available ΔV of approximately 384 m/s. The I_{sp} goes from 222.5 s to 212.5 s throughout the transfer, with a shallow exponential decay. For simplicity an averaged

Table 6.1: Initial and target orbital elements, and convergence criteria, the LEO-vantage Replication Transfer

	a (km)	e	i (°)	Ω (°)	ω (°)	ν (°)
Initial	6879.138	9.212×10^{-4}	97.5549	*free*	*free*	*free*
Target	7378.258	4.906×10^{-4}	99.4882	*free*	*free*	*free*
Convergence	0.1	1×10^{-4}	1×10^{-3}	*free*	*free*	*free*

value of 217.5 s is used and the tank pressure and thus the thrust provided are kept constant.

Due to the on-board Attitude Determination and Control System (ADCS), several constraints needed to be met during the transfer. These provide a good opportunity to include operational constraints in the test scenario and explore how the Reinforced Lyapunov Controller can be used in the presence of such restrictions. The maximum burn is limited to $\Delta V_{\text{burn}} = 0.6$ m/s and at least 1 orbital period was required between the burns to ensure wheel momentum off-load. The dynamical model used is Keplerian with the J_2 perturbation. The equations of motion are given by Eqs. (2.2) and (2.14) in Chapter 2. For reference, a Hohmann transfer between two co-planar circular orbits of similar values requires $\Delta V = 261.90$ m/s but this is not a fair comparison as it removes the need to compute the inclination change. Whilst a co-planar scenario could be considered, it prevents comparison to the results provided from industry. In addition, co-planar cases are often simpler to solve and out-of-plane corrections such as inclination and RAAN are often the most expensive for a spacecraft to conduct, as such there is greater impact in solving these in a close-to-optimal fashion.

Table 6.2: Approximating finite-burn spacecraft parameters

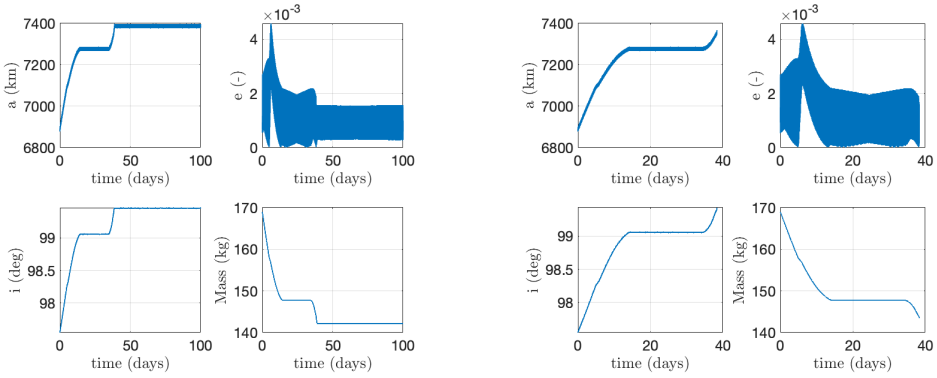
Parameter	Symbol	Value
Initial Spacecraft Mass	m_0	169.0 kg
Initial Propellant Mass	m_0^p	37.6 kg
Constant Specific Impulse	I_{sp}	217.5 s
Total ΔV Budget	ΔV	384 m/s
Individual Burn ΔV Budget	ΔV_{burn}	0.6 m/s
Interval between Burns	$\Delta t_{\Delta V}$	> 1 orbital period
Maximum Engine Thrust	T	0.3 N

Prior to implementing any operational constraints, an investigation was done to understand if the Q-law in its basic form could be used to replicate the transfer. Using the parameters in Tables 6.1 and 6.2, but without any restrictions on the propellant mass or ΔV budget, Table 6.3 demonstrates the difficulties the Q-law has in converging to the target orbit. This is the result of control law chatter.

In Fig. 6.1a, clearly the control law is unable to convergence within the desired tolerance. This is due to the eccentricity value, which is osculating due to the presence of J_2. As can be seen from the evolution of the spacecraft mass, the control law is actually

Table 6.3: Investigating a continuous thrust transfer from a 500km altitude LEO to a 1000km altitude LEO the classical Q-law controller and osculating orbit elements. X_{conv} denotes the default convergence criteria.

Scenario	η_a^{thresh}	Convergence Criteria	Time (days)	Propellant (kg)	ΔV m/s	Mean ΔV_{burn} m/s	Max ΔV_{burn} m/s
Case 1	0.98	$1X_{conv}$	>100	>26.81	>368.5	0.192	1.3
Case 2	0.98	$10X_{conv}$	>100	>26.81	>368.5	0.192	1.3
Case 3	0.98	$100X_{conv}$	38.42	25.62	351.04	0.604	1.19



(a) Case 1 (b) Case 3

Figure 6.1: Figures showing a classical Q-law transfer using osculating orbital elements for different convergence criteria. There appears to be control law chatter near the target orbit.

struggling to compute any control as it is close to convergence, and instead determining no control is required. When the tolerance is relaxed, as shown in Fig. 6.1b, the spacecraft is able to converge. Again the issue appears to be on the eccentricity, with both the semi-major axis and inclination being held constant whilst waiting to adjust the eccentricity. The magnitude of the control acceleration is a cause for concern here.

6.4 Finite-burn Manoeuvre Implementation

As a result of the previous section, it was determined that the control has to be computed in mean orbital elements as opposed to osculating orbital elements. This is a common practice and has often been done for the Q-law [74, 113].

A first-order mapping from mean to osculating elements, and *visa versa*, is briefly explained in Section 2.1.4.1 based on the theory developed by Brouwer and Lyddane [52]. This mapping translates from any osculating elements into mean elements where the short- and long-period oscillations are removed. For the purposes of developing a feedback control law, the matrix $\frac{\partial \xi}{\partial \bar{X}}$ is often approximated as an identity matrix as the off-diagonal terms are all of order J_2 or smaller. This results in an approximated dynamics:

$$\left[\dot{\bar{X}}\right] \approx \mathcal{B}(\bar{X})\boldsymbol{u} + \boldsymbol{A}(\bar{X}), \tag{6.2}$$

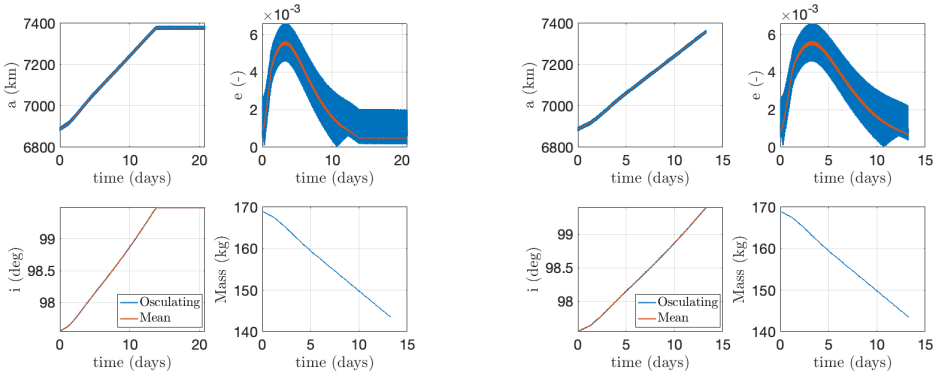
which introduces an error of order J_2. The analytical transformation from the osculating

Table 6.4: Investigating a continuous thrust transfer from a 500km altitude LEO to a 1000km altitude LEO the classical Q-law controller and mean orbit elements. X_{conv} denotes the default convergence criteria.

Scenario	η_a^{thresh}	Convergence Criteria	Time (days)	Propellant (kg)	ΔV m/s	Mean ΔV_{burn} m/s	Max ΔV_{burn} m/s
Case 1	0.98	$1X_{conv}$	20.77	26.63	365.66	0.245	1.03
Case 2	0.98	$10X_{conv}$	13.75	26.35	361.50	0.782	1.03
Case 3	0.98	$100X_{conv}$	13.27	25.44	347.95	0.831	1.03

orbit elements $\boldsymbol{X}$ to the mean elements $\bar{\boldsymbol{X}}$, denoted as $\bar{\boldsymbol{X}} = \xi(\boldsymbol{X})$, is used to transition from osculating dynamics to both the control computation and to determine the convergence to the target elements. For simplicity no initial state offsets are considered.

However, using this assumption, one can use mean orbit elements for both the control law and the convergence criteria around the target orbit. This reduces the control law chattering and the results for a classical Q-law can be seen in Table 6.4.



(a) Case 1 (b) Case 3

Figure 6.2: Figures showing a classical Q-law transfer using mean orbital elements for different convergence criteria. The Brouwer transformation is used in the control computation to convert from osculating to mean orbit elements.

In Fig. 6.2, both cases converge when both the control and the convergence criteria are computed in mean orbit elements. As can be seen, the osculating elements (blue) fluctuate more than the mean elements (orange). This adjustment allows the Q-law to compute a high and continuous-thrust trajectory towards the target orbit, whilst meeting the stricter X_{conv} convergence criteria.

As Table 6.4 demonstrates, the max ΔV_{burn} is greater than the desired individual burn budget. To compensate for this, either a higher η_a^t should be considered or it is necessary to incorporate the operational constraints discussed in Section 6.1. The two most important being the interval between successive burn arcs, $\Delta t_{\Delta V}$, and the individual burn budget, ΔV_{burn}.

Figure 6.3 indicates how this was implemented, and a possible sequence of events during a transfer. There are three possible ways the engine might be switched off. Firstly, if the effectivity $\eta_a < \eta_a^t$ then it is deemed inefficient to thrust and the spacecraft will coast until $\eta_a > \eta_a^t$ before it switches on again. Once the cumulative $\Delta V > \Delta V_{\text{burn}}$ then the

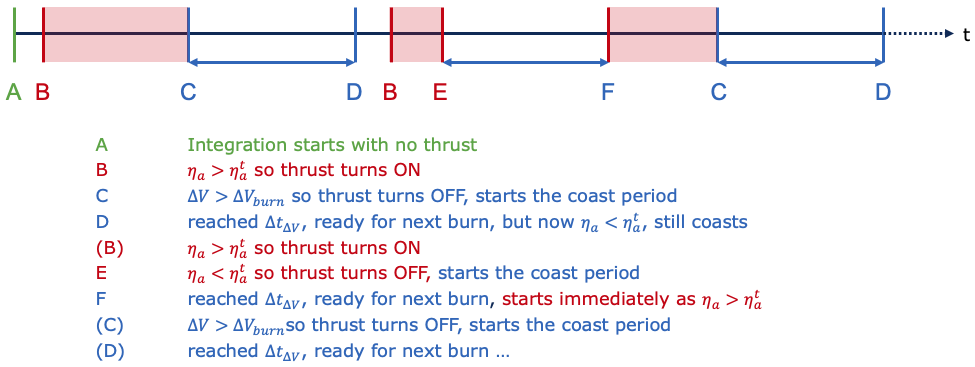


Figure 6.3: Diagram illustrating the a possible sequence of events given the operational constraints on the interval between successive burn arcs, $\Delta t_{\Delta V}$, the individual burn budget, ΔV_{burn}, and the effectivity threshold, η_a^t.

engine is switched off and the spacecraft enters a coast period for $\Delta t_{\Delta V}$ orbital periods. In this time, it would be possible to implement a momentum dumping of the reaction wheels, but this is outside the scope of this work. Hence it is assumed the spacecraft follows the natural dynamics. Once this coast period has passed, it is possible for the spacecraft engine to switch back on again, but this only occurs if and when $\eta_a > \eta_a^t$. Finally, if the spacecraft arrives at the target orbit then the engine is also switched off as no station-keeping procedure is implemented.

6.5 Results

In the following section the results for this investigation are presented. Firstly, a grid search on the effectivity threshold η_a^t is performed to understand the impact of this threshold value. This utilises a classical Q-law controller, and as such is optimised for time-of-flight. Subsequently a PSO is used to optimise the weights $\boldsymbol{W}$ and effectivity threshold η_a^t for both time- and ΔV-optimal transfers. In the ΔV-optimal case the maximum allowed time-of-flight is 90 days, aligned with the desired mission duration. Following this, the Reinforced Lyapunov Control as introduced in Chapter 3 is used to perform the transfer. This includes both RL Q-law and RL Q-Jac controllers. In this instance, the training process includes the operational constraints on both ΔV_{burn} and $\Delta t_{\Delta V}$. Although in practice the interval between successive burn arcs, $\Delta t_{\Delta V}$, is set by the mission operator, here the results for three values: 0.5, 1 and 1.5 orbital periods are presented.

Figures 6.4, 6.5 and 6.6 show these results. The first thing to note is the time-of-flight is always longer as $\Delta t_{\Delta V}$ increases, which makes intuitive sense: the spacecraft is forced to coast for longer. When $\eta_a^t = 0$ this increases from 31.71 days in the $\Delta t_{\Delta V} = 0.5$ case, to 60.79 days in the $\Delta t_{\Delta V} = 1.0$ case, a 91.3% increase, and 90.20 days in the $\Delta t_{\Delta V} = 1.5$ case, a 187.8% increase. The best time-of-flight for constant weights (seen in the PSO time-optimal simulations) also follows a similar but slightly shallower trend. $\Delta t_{\Delta V} = 0.5$

Approach	η_a^t	Time (days)	Propellant (kg)	ΔV m/s
Grid Search	0	31.71	34.77	491.28
	0.5	28.81	31.57	440.99
	0.7	27.29	29.36	406.99
	0.8	26.82	28.03	386.86
	0.9	28.22	27.02	371.53
	0.95	34.27	26.79	368.11
	0.99	42.18	26.39	362.19
PSO Time	-	26.81	28.00	386.32
PSO ΔV	-	44.69	26.34	361.40
RL Q-law Time	-	26.81	28.83	398.83
RL Q-law ΔV	-	38.01	26.61	365.48
RL Q-Jac Time	-	26.78	28.93	400.53
RL Q-Jac ΔV	-	42.35	26.48	363.46

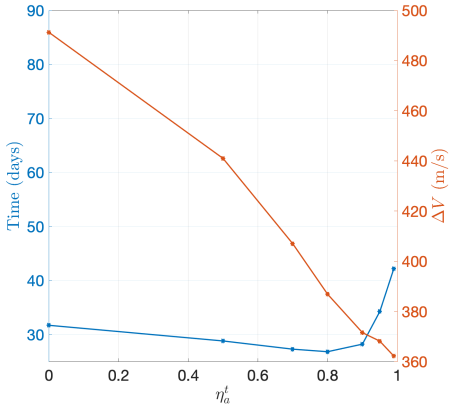


Figure 6.4: Investigating a finite-burn transfer from a 500km altitude LEO to a 1000km altitude LEO. $\Delta t_{\Delta V} = 0.5$ orbital periods and $\Delta V_{\text{burn}} = 0.6$ m/s.

Approach	η_a^t	Time (days)	Propellant (kg)	ΔV m/s
Grid Search	0	60.79	34.88	493.00
	0.5	54.83	31.65	442.35
	0.7	51.13	29.44	408.32
	0.8	49.45	28.14	388.53
	0.9	49.30	27.16	373.67
	0.95	51.09	26.73	367.24
	0.99	61.61	26.42	362.56
PSO Time	-	49.03	27.48	378.60
PSO ΔV	-	72.69	26.39	362.2
RL Q-law Time	-	48.25	27.24	376.85
RL Q-law ΔV	-	63.28	26.31	360.93
RL Q-Jac Time	-	48.54	27.46	378.20
RL Q-Jac ΔV	-	69.45	26.31	360.93

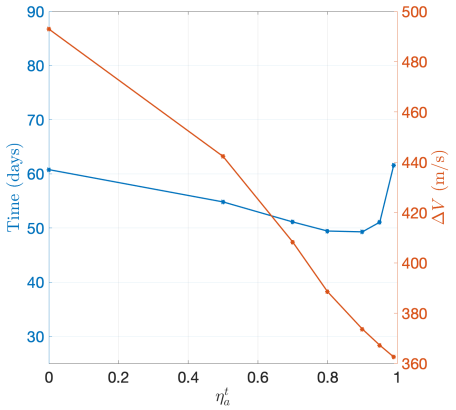


Figure 6.5: Investigating a finite-burn transfer from a 500km altitude LEO to a 1000km altitude LEO. $\Delta t_{\Delta V} = 1.0$ orbital periods and $\Delta V_{\text{burn}} = 0.6$ m/s.

gives 26.81 days, $\Delta t_{\Delta V} = 1.0$ gives 49.03 days, a 82.9% increase, and $\Delta t_{\Delta V} = 1.5$ gives 70.95 days, a 164.6% increase. In all three cases the ΔV are very comparable: 386.32 m/s, 378.60 m/s and 374.84 m/s respectfully. This suggests the limiting factor for the time-of-flight are the coast arcs introduced after the cumulative burn exceeds 0.6 m/s.

Interestingly, the total ΔV used across the ΔV-optimal transfers is very comparable, although it increases slightly as $\Delta t_{\Delta V}$ increases. When $\eta_a^t = 0$, $\Delta t_{\Delta V} = 0.5$ gives 491.28 m/s, $\Delta t_{\Delta V} = 1.0$ gives 493.00 m/s and $\Delta t_{\Delta V} = 1.5$ gives 495.42 m/s. In reality this is not how the transfers would be conducted, so it is better to consider the PSO ΔV-optimal cases, where $\Delta t_{\Delta V} = 0.5$ gives 361.4 m/s, $\Delta t_{\Delta V} = 1.0$ gives 362.2 m/s and $\Delta t_{\Delta V} = 1.5$ gives 363.00 m/s. These values are in good agreement with the observed $\Delta V = 384$ m/s budget from the actual mission. The observed time-of-flight is 44.69 days, 72.69 days and 88.62 days respectfully. This suggests the maximum time-of-flight of 90 days is not required to result in the best ΔV-optimal transfer, and it can in fact be done noticeably faster if the required coasting period is shortened.

In the grid search, different values of η_a^t are compared whilst all other parameters are

Approach	η_a^t	Time (days)	Propellant (kg)	ΔV m/s
Grid Search	0	90.20	35.03	495.42
	0.5	80.83	31.69	443.00
	0.7	74.96	29.44	408.20
	0.8	71.80	28.02	386.64
	0.9	71.22	27.03	371.80
	0.95	82.18	26.75	367.54
	0.99	84.49	26.51	363.88
PSO Time	-	70.95	27.24	374.84
PSO ΔV	-	88.62	26.45	363.00
RL Q-law Time	-	70.43	26.84	368.84
RL Q-law ΔV	-	83.34	26.60	365.32
RL Q-Jac Time	-	71.57	27.25	375.03
RL Q-Jac ΔV	-	82.50	26.58	364.91

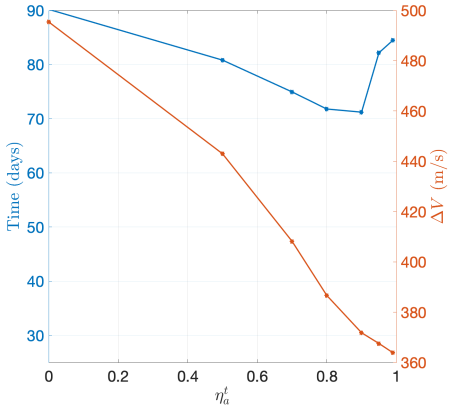


Figure 6.6: Investigating a finite-burn transfer from a 500km altitude LEO to a 1000km altitude LEO. $\Delta t_{\Delta V} = 1.5$ orbital periods and $\Delta V_{\text{burn}} = 0.6$ m/s.

kept constant. The ΔV decreases significantly as η_a^t increases, which matches the intuitive understanding of the effectivity parameter. This should be ensuring the thrust is used at the most optimal locations and hence you need less ΔV to induce a given change in orbital elements. However, the time-of-flight trend is more intriguing. In all three cases the time-of-flight is quite poor as $\eta_a^t \to 1$. Despite being more efficient, the burn arcs are significantly shorter and infrequent. The spacecraft spends more time coasting, waiting for the optimal location to thrust. There appears to be a minimum between approximately $0.8 - 0.9$, after which the time-of-flight increases again. A time-optimal solution might be expected to have the thrust on continuously when the operationally constraints allow it. However, the grid search shows continuous thrusting when permitted by the operational constraints is not the most optimal approach. As a consequence of the constraints imposed by the ΔV_{burn} cut-off, it is suggested there are occasions where coasting for a short period to ensure the burn occurs in the most efficient location is beneficial. This effect would be exaggerated by longer coasting intervals $\Delta t_{\Delta V}$. In addition, perhaps if the truly optimal $\boldsymbol{u}$ was available at each time-step, then the conclusions drawn here might not hold. However, as this is not the case and thrusting throughout the majority of the orbit could also have adverse effects on orbital elements you are targeting. Due to the magnitude of the control acceleration, additional time is then spent correcting any overshooting.

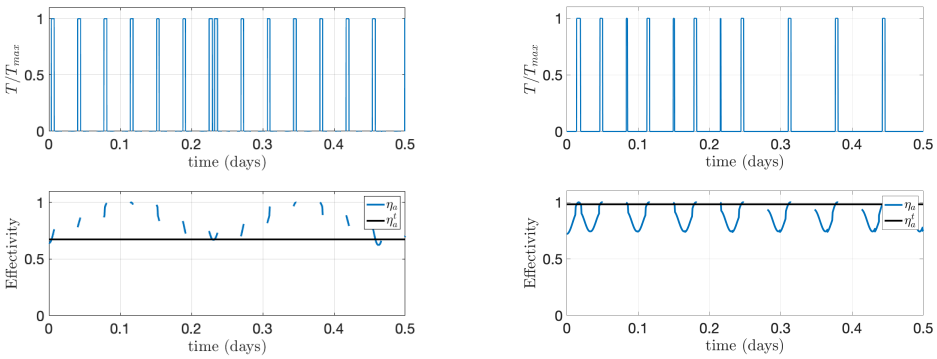
Comparing the time-optimal and ΔV-optimal PSO solutions, in the $\Delta t_{\Delta V} = 0.5$ case one gains 17.55 days at the cost of only 24.92 m/s. In percentage terms this is a 39.3% improvement in time-of-flight at only 6.9% cost in ΔV. For $\Delta t_{\Delta V} = 1.0$, the gain is 23.66 days (32.5%) at the cost of only 16.4 m/s (4.5%) and likewise for $\Delta t_{\Delta V} = 1.5$, the gain is 17.67 days (19.9%) at the cost of only 11.84 m/s (3.3%). In all three cases there is a strong argument to use the time-optimal solution regardless. However, the ΔV-optimal solutions have shorter burn arcs and more accurately recreate finite-burn manoeuvres.

Figures 6.8, 6.9 and 6.10 present the time- and ΔV-optimal RL Q-Jac results. In each

case, three plots are provided: the first is the burn arcs in the ECI frame - subplots a and b; the second shows the weight profile - subplots c and d; and lastly a comparison between η_a and η_a^t and how this corresponds to the engine switching on and off during the transfer (labelled $T/T_{\max}$) is given - subplots e and f. In the ΔV-optimal $\Delta t_{\Delta V} = 1.5$ case, for example, the corresponding figures are Figs. 6.10b, 6.10d and 6.10f respectively.

Considering Figs. 6.8a, 6.8b, 6.9a, 6.9b, 6.10a and 6.10b, it is clear the majority of the burn arcs are located at the nodal points: the ascending and descending nodes. This only changes towards the end of the transfers, when the spacecraft is trying to converge to the target orbit. In the time-optimal cases, the length of each burn is governed by the ΔV_{burn} cut-off. On a few occasions $\eta_a < \eta_a^t$ also breaks the burn into two parts, but this occurs less frequently. In the ΔV-optimal solutions, η_a^t plays a greater role in determining the burn locations, matching intuition and utilising the most efficient locations for the burns. Hence, as the effectivity threshold is lower, the time-optimal transfers have significantly longer burn arcs than the ΔV transfers. Across the three different $\Delta t_{\Delta V}$ scenarios, the average burn duration (cumulative duration until $\Delta V_{\text{burn}} = 0.6$ m/s is reached) is: 5.5, 4.6 and 5.4 minutes for the time-optimal transfers, versus 2.9, 2.7 and 4.9 minutes for the ΔV transfers.

Figures 6.8c, 6.8d, 6.9c, 6.9d, 6.10c and 6.10d show the weight profiles during the transfers. The similar performance between the RL simulations and PSO Q-law can be accounted for by the lack of variation in these weights during the transfers. This suggests the state-dependence is not really used here, perhaps because the orbital elements do not vary enough to result in significantly different $\boldsymbol{W}$. However, similar issues were overcome in Section 4.4, where LEO orbits are also used. In that instance, reducing $\Delta\Omega$ is the key to an optimal transfer. Here Ω is not targeted and the controller is not looking to exploit the effects of J_2. It is possible different actor network inputs could alleviate the lack of state-dependence and this warrants further investigation. However, it is important to consider the additional operational constraints imposed and their impact on the learning process. In Section 5.5, concerns over the effect added stochasticity can have on the learning process were discussed. This could also explain the difficulty observed here. Although the operational constraints are included during the training, they are not something the agent can determine. In fact they are purely events the controller experiences after the actions are taken. In this fashion they resemble the eclipse effects modelled in Section 4.2.1, however, they remain throughout the entirety of the transfer and cannot be avoided, for example, by moving to higher altitudes. This means a stochastic action is selected regardless of whether or not the control acceleration can be applied. Hence, during the learning process, many actions are taken that are hidden behind the operational constraints. From a learning perspective, these actions are interpreted as all having the same value and this could explain the minor variation in behaviour observed throughout the transfers. Developing methods of incorporating the operational constraints more actively in the RL formulation could improve the results seen here.



(a) Time-optimal Effectivity

(b) ΔV-optimal Effectivity

Figure 6.7: Comparing η_a over the first 0.5 days for Time- and ΔV-optimal finite-burn transfers using $\Delta t_{\Delta V} = 0.5$. The operational constraints are reflected in the gabs between burns and a clear link between the engine switching on/off and η_a crossing the η_a^t boundary is observed.

Analogous to the PSO simulations, the argument to use the time-optimal solutions in place of the ΔV-optimal solutions is quite convincing. This is because the percentage gains in time result in smaller ΔV penalties. For $\Delta t_{\Delta V} = 0.5$ there is a 15.57 day (36.8%) improvement at the cost of 37.07 m/s (10.2%). For $\Delta t_{\Delta V} = 1.0$, that is 20.91 days (30.1%) at the cost of 17.27 m/s (4.8%) and similarly for $\Delta t_{\Delta V} = 1.5$, the gain is 10.93 days (13.2%) at the cost of 10.12 m/s (2.8%). Therefore, in all three cases there is a strong argument to use the time-optimal solution.

The plots of η_a and η_a^t are difficult to interpret given the high frequency of the effectivity. Thus, a comparison of the first 0.5 days for the $\Delta t_{\Delta V} = 0.5$ time- and ΔV-optimal transfers is given in Fig. 6.7. The operational constraints are reflected in the gaps between burns and a clear link between the engine switching on/off and η_a crossing the η_a^t boundary is observed. In addition, if $\eta_a < \eta_a^t$ after the mandatory $\Delta t_{\Delta V}$ coast, the engine will remain off as indicated in Fig. 6.3.

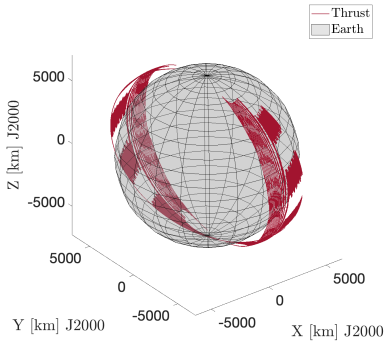
Across the various simulations, the RL time-optimal results outperform the PSO results in all except $\Delta t_{\Delta V} = 1.5$ RL Q-Jac case, whilst the ΔV-optimal only do so in the $\Delta t_{\Delta V} = 1.0$ case. Figure 6.9f shows a very high η_a^t value and indicates the spacecraft engine was off for the majority of the transfer between day 5 to day 40. This is also reflected in Fig. 6.9b, where the coasting can be seen due to lack of burn arcs. Comparing this to Figs. 6.8f and 6.10f, it appears the Reinforced Lyapunov Controller was unable to find the ΔV-optimal strategy for the $\Delta t_{\Delta V} = 0.5$ and 1.5 cases. It is unclear if this is due to the non-integer coast period or not. As seen in Chapter 4 with eclipse effects, coast periods that are not communicated to the agent prove difficult for the Reinforced Lyapunov Controller. In the case of integer coast periods, the controller finds itself in the same true anomaly state as before the coast period. As such, one explanation could be that a choice of η_a^t with the integer coast period has the same effect with or without said coast period, in turn making the learning process much easier for the agent. How-

ever, this does not hold for non-integer coast periods, where a choice of η_a^t will have a different impact because of the different locations along the orbit where the thrust can be activated. Interestingly, Figs. 6.8d, 6.9d and 6.10d show that the best solution prioritises the eccentricity term whilst the other two less-optimal solutions are trying to prioritise the inclination. The PSO indicates that the time- and ΔV-optimal results involve the smallest $\Delta t_{\Delta V}$. The likely explanation for any improved performance observed in the RL Q-Jac over the PSO simulations comes from the slight variation in the weights and the improved understanding of the effectivity parameter. Small changes in $\boldsymbol{W}$ can change η_a enough to cross the boundary with η_a^t and switch the engine on at a crucial time. These small variations in $\boldsymbol{W}$ cannot occur for the PSO.

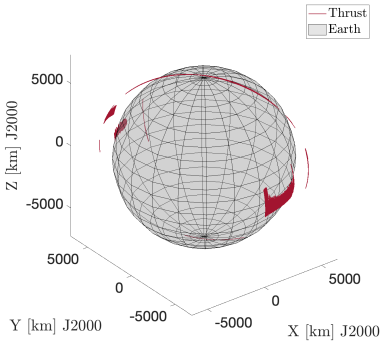
6.6 Discussion and Summary

This chapter investigated the potential for Lyapunov control laws, and the Reinforced Lyapunov Controller, in generating trajectories which resemble those more commonly computed by chemical propulsion systems. Unlike EP systems, where the acceleration provide is relative low, chemical propulsion systems provide a higher unit of thrust and generally do not burn for a long period of time. The Reinforced Lyapunov Controller approach was modified to produce finite-burn manoeuvres. Extending this to higher thrust-to-mass ratios could lead to approximating impulsive trajectory design approaches. In the short term this is of great interest to SSTL because it would allow them to utilise this tool for current mission analysis rather than for the longer term development of low-thrust missions. In addition, there is growing potential for using low-thrust trajectories as the starting point for solving optimal sequence of finite-burn manoeuvres and even for solving N-impulse multi-revolution trajectories.

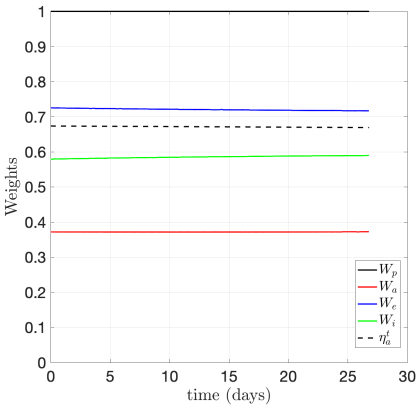
The test case was chosen to replicate the LEO-vantage 1 transfer, and involves transferring between two LEO SSOs, accounting for the J_2 perturbation. Operational constraints are also taken into account, restricting the maximum ΔV for a burn, and the interval between successive burns once this ΔV quota is reached. Replicating these involves adjustments to the implementation from previous chapters. Although not designed for this purpose, it provides a useful exercise pushing the boundaries of the approach developed so far. Results indicate it is possible to obtain comparable time-optimal and ΔV-optimal transfers, with the Reinforced Lyapunov Controller performing comparably to the PSO Q-law approach. However, as a result of the LEO-LEO transfer, there is little change in state during the transfer. This results in almost constant state-weight dependence and does not highlight the potential of this approach. In addition, it is noted this may be the result of the limitations operational constraints can have during the learning process.



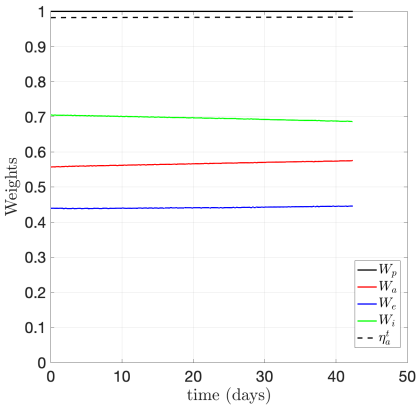
(a) Time-optimal Trajectory



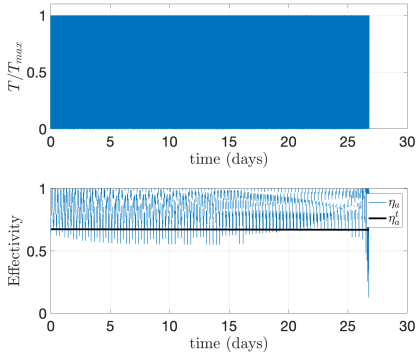
(b) ΔV-optimal Trajectory



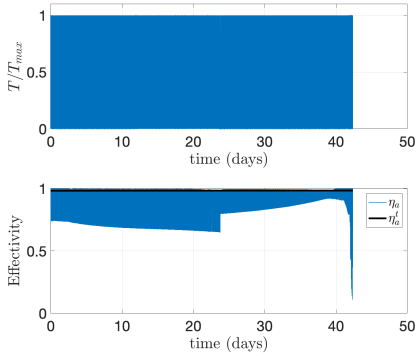
(c) Time-optimal Weights



(d) ΔV-optimal Weights

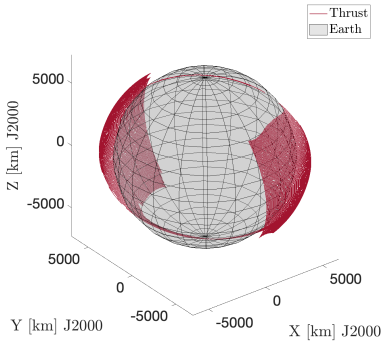


(e) Time-optimal Effectivity

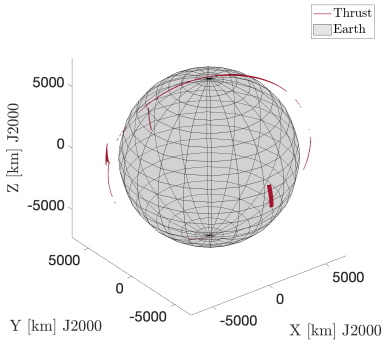


(f) ΔV-optimal Effectivity

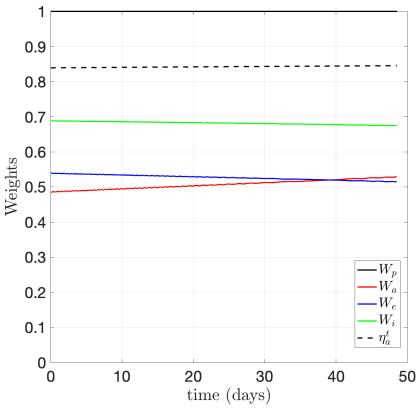
Figure 6.8: Time- and ΔV-optimal finite-burn transfers using a Reinforced Lyapunov Controller. Results for a coast period of $\Delta t_{\Delta V} = 0.5$ are shown. Red arcs indicate when the engine is switched on, whilst grey indicated when the spacecraft is coasting.



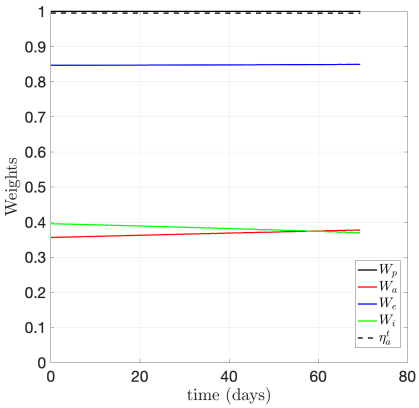
(a) Time-optimal Trajectory



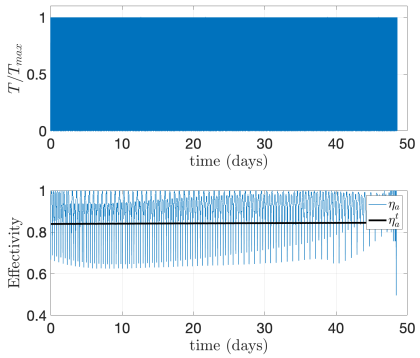
(b) ΔV-optimal Trajectory



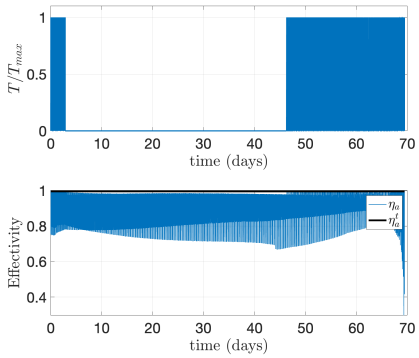
(c) Time-optimal Weights



(d) ΔV-optimal Weights

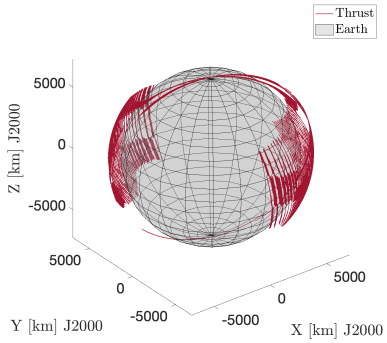


(e) Time-optimal Effectivity

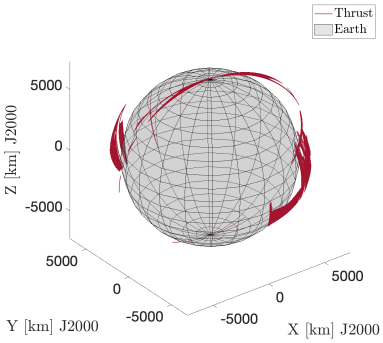


(f) ΔV-optimal Effectivity

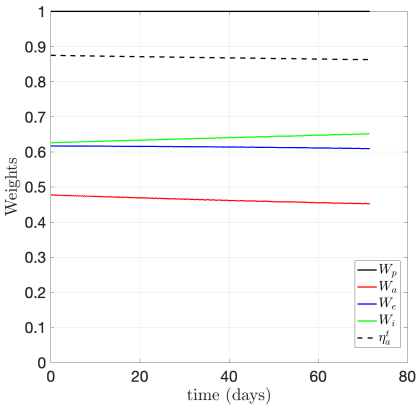
Figure 6.9: Time- and ΔV-optimal finite-burn transfers using a Reinforced Lyapunov Controller. Results for a coast period of $\Delta t_{\Delta V} = 1.0$ are shown. Red arcs indicate when the engine is switched on, whilst grey indicated when the spacecraft is coasting.



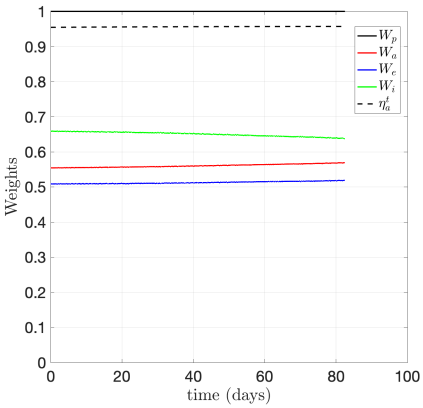
(a) Time-optimal Trajectory



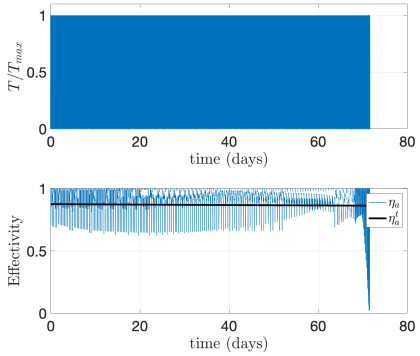
(b) ΔV-optimal Trajectory



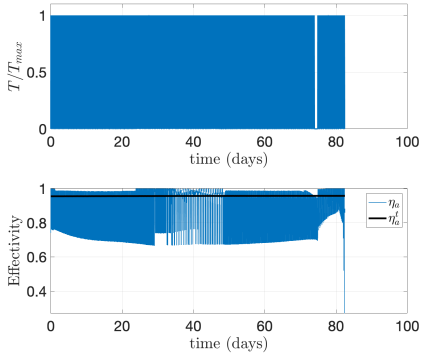
(c) Time-optimal Weights



(d) ΔV-optimal Weights



(e) Time-optimal Effectivity



(f) ΔV-optimal Effectivity

Figure 6.10: Time- and ΔV-optimal finite-burn transfers using a Reinforced Lyapunov Controller. Results for a coast period of $\Delta t_{\Delta V} = 1.5$ are shown. Red arcs indicate when the engine is switched on, whilst grey indicated when the spacecraft is coasting.

Chapter 7
Trajectory Design for Earth-Moon Spiral Transfers

In this chapter the Reinforced Lyapunov Controller is extended to the problem of designing low-thrust, many-revolution spiral transfers from the Earth to the Moon. There are several major novelties over previous chapters. Firstly, the dynamical environment can no longer be approximated as two-body, and instead the perturbing acceleration of the third-body eventually takes over to become the main acceleration. This transition is firmly outside the understanding of the Lyapunov controller and demonstrates the extended capabilities thanks to the RL architecture. Secondly, two actor networks are required instead of one, resulting in a different learning approach. Lastly, constraints such as rendezvous conditions are now required in order to enter the Lunar sphere of influence (SOI). Part of this work was presented at the *AAS Astrodynamics Specialist Conference* [43], and was completed by H. Holt, N. Baresi and R. Armellin.

7.1 Introduction

Future missions to the Moon and beyond are increasingly likely to involve low-thrust propulsion technologies due to their superior propellant efficiency. The higher ΔV enabled by these systems offers exciting opportunities, from Cubesats making use of rideshare opportunities to the upcoming NASA's Lunar Orbital Platform Gateway plans, which will at least partially involve EP technologies [123]. The SMART-1 mission successfully demonstrated the potential for EP as a tool for Lunar and interplanetary missions when it flew to the Moon in 2003 [4]. The technology, along with solar power sails (JAXA's IKAROS and OKEANOS missions [8]), is being increasingly used in both planetary and interplanetary spacecraft (NASA's Dawn and GRAIL missions, JAXA's Hayabusa 1 & 2 and recently ESA's BepiColombo [6, 7, 124, 125]).

Low-thrust many-revolution trajectories in restricted three-body contexts still present a difficult design problem, owing to the near continuous thrust, lack of control authority and chaotic dynamics. As such, mission design often follows a regular pattern: feasibility analysis in a simplified model; initial guess generation; optimal trajectory computation using computationally demanding indirect and direction methods; and developing a guidance approach to follow the reference trajectory. Much work has gone into the feasibility analysis and initial guess generation, as these significantly impact the convergence

and optimality of the final results.

As discussed in Chapter 2, CLFD control laws, and in particular Lyapunov control laws, have been shown to be suitable as initial guesses for indirect and direct methods [37, 38, 70], and they have great potential for on-board use as a guidance law. Lots of recent work has looked to combine these Lyapunov control laws with global optimisers for Earth-Moon spiral transfers - as discussed in Section 2.2.2.6. As such, this research area provides a suitable test bed for the Reinforced Lyapunov Controller.

Previous chapters have demonstrated increased optimality whilst retaining the stability of Lyapunov control laws using RL methods. This has resulted in a lightweight and closed-loop control law that can be used for both initial trajectory design and subsequent on-board guidance, but has so far been limited to geocentric transfers with perturbing accelerations. The major draw for RL algorithms, however, is their performance in unfamiliar environments [46]. Several works use RL as a tool for trajectory design and guidance in the CR3BP [97, 98, 100, 101, 103]. Kwon *et al.* [106] have recently used RL as a tool for low-thrust many-revolution transfers.

This chapter combines the robust nature of Lyapunov control laws with the state-dependence of RL to design preliminary low-thrust transfers to Lunar orbits. The aim is to investigate if the performance of Lyapunov control laws can be increased in an environment where the dynamics are unknown to the controller and constantly evolving. The RL agent is able to explore the effects of the perturbing forces during training and learn to exploit their existence. A long term aim is to use the same methodology to provide an initial guess for subsequent optimisation as well as act as a guidance method for the resulting optimised solution.

This chapter is structured as follows. First, the problem setup is discussed, along with the inertial CR3BP framework. Background theory on the RL architecture is provided, although detailed discussion can be found in Chapters 2 and 3. The application to the Lunar spiral transfers includes an illustrative diagram and algorithmic flowchart. Three different simulation approaches are then discussed in the numerical results, the most promising of which are the forwards propagation simulations. These are then extended further to explore the potential of dynamical systems theory to aid the capture and rendezvous around the Moon.

7.2 Problem setup

The test case considered is the SMART-1 trajectory, initially investigated by Betts [126] using a direct method and subsequently Shannon *et al.* [70]. The time-optimal transfer obtained was 198.38 days, which Shannon *et al.* [70] were able to reproduce using a Q-law, evolutionary algorithm and direct collocation to obtain a transfer in 194.25 days. The departure orbit from Earth is a GTO, which is defined in the ECI frame. The arrival orbit around the Moon is a Lunar polar orbit (LPO) defined in the J2000-centred at the Moon,

here known as the Moon-centred inertial (MCI). The parameters for both are given in Table 7.1. These results provide the benchmark for the later comparisons. The modelled spacecraft has a mass of 350 kg, a thrust of 73.19 mN and an I_{sp} of 1675.8 s.

Table 7.1: Initial and target orbital elements for a the GTO-LPO transfer. Departure epoch Dec 20, 2002.

	a (km)	e	i (°)	Ω (°)	ω (°)	ν (°)
Departure GTO in ECI	24661.14	0.7162279	7.0	*free*	178.0	*free*
Arrival LPO in MCI	7238.0	0.6217187	90.0	90.0	270.0	*free*

Table 7.2: Reference solutions available in the literature for the GTO-LPO transfer.

	Method	Time (days)	Propellant (kg)
Time	Betts [126]	198.38	75.34
	Shannon [70]	194.25	74.75
Mass	Betts [126]	201.28	75.00
	Shannon [70]	247.80	58.74

The proposed approach of combining RL with a Lyapunov controller to tackle this Lunar spiral transfer can be broken down into three sections: the dynamics, the control formulation and the RL formulation. For the dynamics two different models are considered: an inertial CR3BP model for the propagation and a Keplerian model for the Lyapunov control formulation. The inertial CR3BP is used to provide a straightforward continuation to the full ephemeris model in the future, although this is not presented here. The control itself is based on Lyapunov control and uses the well-known and established Petropoulos "Q-law" [35, 36] - see Section 2.2.2.3. An RL framework is then used to improve the performance of this Lyapunov-based controller using state-dependent parameters and guaranteed stability - see Chapter 3. Due to the presence of third-body perturbations, ensuring stability is more difficult, however, the cone-clock approach from Chapter 4 is used to allow increased freedom whilst ensuring such stability when possible.

An inertial CR3BP model is assumed for the dynamics, with the Earth and Moon as the two massive bodies - see Section 2.1.4.2. This is the simplest model to start with and can easily be extended to include either the Sun's third-body perturbation or the spherical harmonics from both bodies. This work is done in an inertial integration frame, avoiding the synodic rotating CR3BP commonly used for such problems, as the Lyapunov control is formulated in an inertial two-body problem. In addition, working in the inertial frame enables an easy transition from a CR3BP model to one where the positions of the Earth and Moon are given by *NAIF SPICE* Ephemeris kernels [54].

7.3 Readjusted RL architecture

Now the application of a Lyapunov controller and RL framework to designing low-thrust spiral transfers from GTO-LPO is considered. Due to the limitations in rendezvous capabilities for Lyapunov controllers, it is often easiest to backward propagate the ECI leg of the transfer [68, 70, 71]. This means the trajectory starts from a predetermined patch point and targets the departure GTO whilst integrating backwards in time. Following the lead of Shannon *et al.* [70], the first approach involves defining a patch point on the Lunar SOI. From here, one can backwards propagate in ECI to reach the departure GTO, and forwards propagate in MCI to the arrival LPO. Unfortunately, this still presents difficulties as a suitable patch point needs to be selected and this will eventually limit the optimality of the results. This works well in Shannon *et al.* 70 because a direct collocation and optimisation takes place at later stages. However, in the initial simulations it appeared very difficult to free this patch point with much success. In addition, whilst the patch point restricts the route of the trajectory, it also influences the geometry of the problem (or the epoch) and means the spacecraft mass at this patch point needs to be guessed, significantly affecting the transfer.

Due to the difficulties in handling the geometry, the patching point and mass, three different approaches are presented, with varying degrees of success:

- Backwards-only propagation directly from LPO to GTO where the geometry is fixed, along with the LPO arrival mass m_{arr}.
- Backwards-only propagation directly from LPO to GTO where the geometry and LPO arrival mass m_{arr} are free.
- Forwards-only propagation starting at GTO, successfully rendezvousing with the Moon and targeting the LPO. The geometry is also freed via the initial GTO departure RAAN Ω_{dep}.

The first approach is the easiest for the RL methodology to solve, as no constraints need to be met. The second approach proved very challenging and highlights the difficulties that RL methods have in handling constraints such as the one on the departure mass m_{dep}. However, it is necessary in increase the generality of the approach beyond the fixed problem. As a result of these difficulties, the problem is flipped and instead the transfer is propagated forward in time. This removes the problem of guessing the LPO arrival mass m_{arr}. As can be see in Fig. 7.1, the trajectory is forward propagated in ECI from GTO to the Keplerian orbit of the Moon. The rendezvous is learnt by including a difference in position between the satellite and the Moon when it arrives at the Keplerian orbit in the cost function, and the Lyapunov controller maintains the orbit until the Moon catches up and it enters the Lunar SOI. Once entered, an LPO is targeted in the MCI frame. In order to encourage rendezvous, a penalty term is included in the

cost function on entering the Lunar SOI, as is the v_∞, which helps prevent subsequent escapes. The following two sections will discuss the training process for the forwards propagation in detail.

7.3.1 Two Actor Networks

During training two networks are used, one in the ECI frame and the other in the MCI frame. For each, two sets of parameters are used: the active θ_k and the newly updated θ. A batch of N trajectories τ are generated using the stochastic policy π_{θ_k}. Unlike in previous chapters where two deterministic trajectories where used, here three are required in order to distinguish the ECI and MCI performances. Hence, one deterministic trajectory is computed with the active policy for both legs $\tau(\theta_k^{ECI}) + \tau(\theta_k^{MCI})$. The second deterministic trajectory involves the newly updated ECI policy with the active MCI one: $\tau(\theta^{ECI}) + \tau(\theta_k^{MCI})$. Finally, the newly updated policy in MCI is combined with the active one in ECI: $\tau(\theta_k^{ECI}) + \tau(\theta^{MCI})$. Figure 7.1 provides an illustration of these deterministic trajectories.

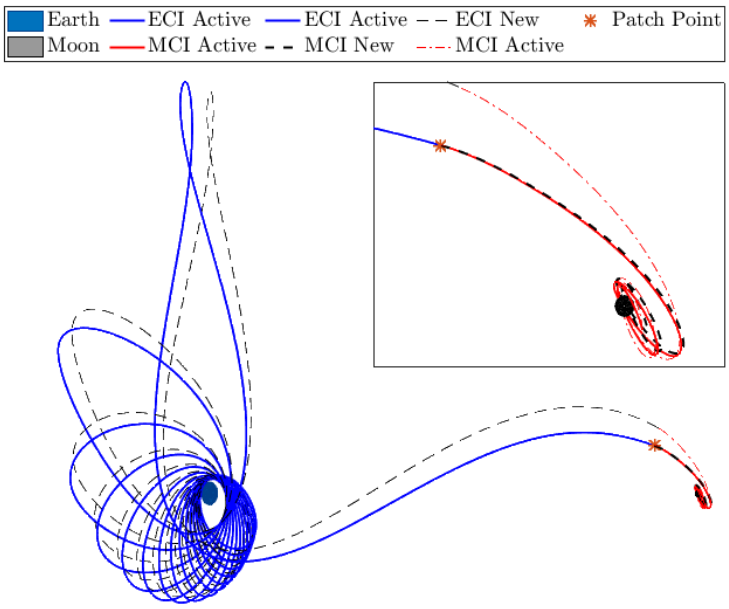


Figure 7.1: Illustration of RL architecture for Earth-Moon spiral transfers. Two networks are used in the ECI and MCI frames respectively. Three distinct trajectories are shown involving ECI Active and MCI Active, ECI Active and MCI New, and ECI New and MCI Active policies. (Note: this is for illustration purposes and not reflective of the spacecraft parameters used.)

Using the costs associated with the stochastic trajectories $\tau(\pi_{\theta_k})$ one can iteratively update θ. However, the three deterministic trajectories enable only updating θ_k if either of the ECI or MCI trajectories outperforms the active trajectory. Hence, update $\theta \leftarrow$

$\theta - \alpha \nabla_\theta J(\theta)$ occurs every iteration for both ECI and MCI networks (i.e., after a batch of N trajectories). However, the update $\theta_k \leftarrow \theta$ only occurs if $\tau(\theta)$ outperforms $\tau(\theta_k)$. An algorithmic pseudocode is shown in Algorithm 5. Once the problem is initialised, line 5 indicates the departure RAAN is determined by the ECI policy $\pi^{ECI}_{\theta_k}$. The trajectory is propagated in the ECI frame using the ECI policy until it enters the Lunar SOI - see lines 6-8. This signifies the patch point, and the state is converted to the MCI frame in line 9. The remainder of the transfer is computed using the MCI policy $\pi^{MCI}_{\theta_k}$. Once both the ECI and MCI legs have been computed, the costs are assigned in line 13. This is repeated for a batch of $N + 4$ trajectories, before the critic networks and advantage functions are used to compute the PPO update in lines 14-16 for both the ECI and MCI policies. The algorithm outlines the forwards propagation approach. The backwards propagation can be achieved by defining a patch point as opposed to a departure RAAN in line 5 and reversing the time and control directions in lines 6-8.

Each trajectory is divided into fixed time-intervals ensuring each state-action pair's influence is not determined by the time interval. At the start of an interval, the actor network is called with the current state to determine the action:

$$\begin{aligned} ECI: &\quad \left[W_a\, W_e\, W_i\, W_\Omega\, W_\omega\, \eta_a^t\, \alpha\, \beta\, \Omega_{\text{dep}}\right] \\ MCI: &\quad \left[W_a\, W_e\, W_i\, W_\omega\, \eta_a^t\, \alpha\, \beta\right], \end{aligned} \tag{7.1}$$

which are then kept fixed for the following time-interval. As in other chapters, there is a trade-off here: on one hand a smaller time-step allows for more frequent variation of the weights, potentially improving the response by allowing rapid changes in behaviour. However, it also increases the complexity of the learning process. As in the previous chapters, an interval of 0.25 days is used. During validation and/or deployment the actor network can be called at the frequency required by the integrator (i.e., embedded) or the satellite operator and, as such, the weights are free to vary at a faster rate. A different transfer problem, particularly one where the expected time-of-flight is much shorter, might require a different time-interval to be selected. The RL parameters used reflect those used in Chapter 3 Table 3.5, however, for the MCI controller $r_{\text{petro}}^{\text{p-min}} = 1,747.5$ km instead of the usual ECI value of 6,578 km.

7.3.2 Cost Functions

As it provides all the information during the learning process, the cost function formulation is key to learning an optimal policy. Two cost functions used for the time-optimal and mass-optimal simulations are presented. Conventionally only one policy is learnt, however here two policies are required, π^{ECI} in the ECI frame and π^{MCI} in the MCI frame. They are not mutually independent of each other as the performance of π^{MCI} is influenced by π^{ECI} because the MCI trajectory follows the ECI one.

For the time-optimal transfers, let t^{ECI} and t^{MCI} be the time of flight for each leg and

Algorithm 5 RL Lunar Spiral Pseudocode: Forwards Propagation

Set t_{dep}, $\boldsymbol{X}_{dep}^{ECI}$, $\boldsymbol{X}_{T}^{ECI}$, $\boldsymbol{X}_{T}^{MCI}$
for k = 1: Max iterations **do** ▷ Learning Iteration
  for n = 1: # episodes+4 **do** ▷ Minibatch (can parallelise)
    Determine stochastic/deterministic policy π_n^{ECI} and π_n^{MCI}
    $\Omega_{dep} \leftarrow \text{Actor}(\boldsymbol{X}_0^{ECI}, \pi_n^{ECI})$
    while $\|\boldsymbol{r}^{ECI} - \boldsymbol{r}_{Moon}^{ECI}\| > r_{SOI}$ km **do** ▷ ECI Integration
      $\boldsymbol{W} \leftarrow \text{Actor}(\boldsymbol{X}^{ECI}, \pi_n^{ECI})$
      $\boldsymbol{X}^{ECI} \leftarrow \text{DynamicsECI}(t, \boldsymbol{X}^{ECI}, \boldsymbol{X}_T^{ECI}, \boldsymbol{W})$
    $\boldsymbol{X}^{MCI} \leftarrow \boldsymbol{X}^{ECI}$ ▷ Convert ECI to MCI
    while $\|\boldsymbol{r}^{MCI}\| < r_{SOI}$ km and not converged **do** ▷ MCI Integration
      $\boldsymbol{W} \leftarrow \text{Actor}(\boldsymbol{X}^{MCI}, \pi_n^{MCI})$
      $\boldsymbol{X}^{MCI} \leftarrow \text{DynamicsMCI}(t, \boldsymbol{X}^{MCI}, \boldsymbol{X}_T^{MCI}, \boldsymbol{W})$
    Assign costs c^{ECI} and c^{MCI} and calculate cost-to-go ▷ See Eqs. (7.2) and (7.3)
  $\hat{V}^{ECI} \leftarrow \text{Critic}(\boldsymbol{X}^{ECI}, C^{ECI})$ and $\hat{V}^{MCI} \leftarrow \text{Critic}(\boldsymbol{X}^{MCI}, C^{MCI})$
  $\hat{A}^{ECI} \leftarrow \text{AdvFn}(\boldsymbol{X}^{ECI}, c^{ECI}, \hat{V}^{ECI})$ and $\hat{A}^{MCI} \leftarrow \text{AdvFn}(\boldsymbol{X}^{MCI}, c^{MCI}, \hat{V}^{MCI})$
  PPO Update ▷ See Eqs. (2.74) and (2.75)
  Update new policy for each network ▷ See Eq. (2.70)
Deploy using final θ_k^{MCI} and θ_k^{ECI}

t_{aim} the maximum allowed integration time. t_{conv} is the convergence criteria on the LPO (set to 0.25 days) and t_{res}^{MCI} is the residual on the final state. If $r_{SOI} = 60{,}000$ km and $\Delta r_{SOI} = \|\boldsymbol{r}^{ECI} - \boldsymbol{r}_{Moon}^{ECI}\| - r_{SOI}$, then the total cost is $c_{\text{time}} = c_{\text{time}}^{ECI} + c_{\text{time}}^{MCI}$, where the two legs have costs given by:

$$c_{\text{time}}^{ECI} = \begin{cases} t^{ECI}, & \text{if } \Delta r_{SOI} \leq 0, \\ t_{\text{aim}} + \Delta r_{SOI} + v_\infty, & \text{otherwise.} \end{cases} \tag{7.2a}$$

$$c_{\text{time}}^{MCI} = \begin{cases} t^{MCI}, & \text{if } t_{\text{res}}^{MCI} \leq t_{\text{conv}}, \\ t_{\text{aim}}, & \text{otherwise.} \end{cases} \tag{7.2b}$$

Here $v_\infty = \|\dot{\boldsymbol{r}} - \dot{\boldsymbol{r}}_s\|$ in the ECI frame is used to help ensure a temporary capture around the Moon. Otherwise trajectories which arrive and escape the Lunar SOI are observed. In a similar fashion, using $m_{max} = 350$ kg, the total mass-optimal cost is $c_{\text{mass}} = c_{\text{mass}}^{ECI} + c_{\text{mass}}^{MCI}$, where the two legs are given by:

$$c_{\text{mass}}^{ECI} = \begin{cases} m_p^{ECI}, & \text{if } \Delta r_{SOI} \leq 0, \\ m_p^{ECI} + t^{ECI} + t^{MCI}, & \text{if } t^{ECI} + t^{MCI} \geq t_{\text{aim}}, \\ m_{max} + \Delta r_{SOI} + v_\infty, & \text{otherwise.} \end{cases} \tag{7.3a}$$

$$c_{\text{mass}}^{MCI} = \begin{cases} m_p^{MCI}, & \text{if } t_{\text{res}}^{MCI} \leq t_{\text{conv}}, \\ m_p^{MCI} + t^{ECI} + t^{MCI}, & \text{if } t^{ECI} + t^{MCI} \geq t_{\text{aim}}, \\ m_{max}, & \text{otherwise.} \end{cases} \tag{7.3b}$$

This includes an upper bound on the allowed time-of-flight, t_{aim}, which is used for

literature comparison. Simulations show that even longer time-of-flights allow for better mass-optimal solutions. In both cases, the effectivity threshold η_a^t is used to introduce coasting periods if required. In addition, if the osculating Keplerian orbit is hyperbolic, no control is calculated and instead the spacecraft coasts until the the osculating Keplerian orbit is closed. It should be noted this likely affects the optimality of the time-of-flight simulations presented.

7.4 Backwards Propagation

7.4.1 Fixed geometry and Arrival Mass

In this setup, the geometry of the problem is fixed. A grid search over the arrival RAAN, Ω_{arr}, was done using the classical Q-law to determine a fixed value for comparison. For simulations minimising time-of-flight and propellant mass these are $\Omega_{\text{arr}} = 166°$ and $\Omega_{\text{arr}} = 154°$ respectively. Naturally these Ω_{arr} are not expected to be the optimal ones for a trained RL algorithm, however they can be used to fix the geometry of the problem to allow a fair comparison between all methods. Owing to the backwards propagation, the arrival mass at LPO, m_{arr}, is fixed following the literature values provided in Table 7.2. Hence, for the time-optimal simulations the LPO arrival mass m_{arr} is 275 kg and RAAN Ω_{arr} is 166°. For the mass-optimal simulations m_{arr} is 291.26 kg and RAAN Ω_{arr} is 154°. Lunar arrival epoch July 31, 2003.

Table 7.3: Backwards propagation with fixed m_{arr} and Ω_{arr} for the GTO-LPO transfer.

Objective	Method	Time (days)	Propellant (kg)	m_{dep} (kg)
	Classical	247.72	95.32	370.32
Time	PSO free	216.87	83.45	358.45
	RL Q-law	211.05	81.21	356.21
	RL Q-Jac+CC	214.42	82.35	357.35
Mass	PSO free	250.00	68.66	359.92
($\leq$ 250 days)	RL Q-law	249.92	71.36	362.62
	RL Q-Jac+CC	249.14	70.43	361.69

Table 7.3 provides a summary of both the time and mass-optimal results. As expected, the classical Q-law is particularly bad, taking 53.47 days longer than the optimal solution found in [70]. The "PSO free" is able to tune the weights to improve by 30.85 days and 26.66 kg in the time and mass-optimal cases respectively. It it clear that the state-dependence offered by the RL approach can improve the optimality of the time-of-flight results in the fixed geometry setup. The best time of flight is 211.05 days, a 5.82 day (2.7%) improvement on the "PSO free". The best mass-optimal transfers, however, is still 1.77 kg (2.6%) worse compared to equivalent "PSO free". It seems the RL Q-law is more suited for the this setup than the RL Q-Jac+CC, although the two are very similar for the mass-optimal case. However, note the RL Q-Jac+CC guarantees stability when the control authority allows it and this may come at the cost of optimality. As the arrival

mass is fixed, there is an offset between the theoretical spacecraft mass of 350 kg and the results of the backwards propagation. There is no expectation to reduce this gap whilst the geometry remains fixed.

7.4.2 Free geometry and Arrival Mass

As a natural extension to the results from the previous section, here the setup is very similar but now the arrival RAAN Ω_{arr} and mass m_{arr} are determined by the actor network and need to be learnt. Note that a potentially easier problem could involve fixing the arrival mass m_{arr} and just letting the geometry vary. From a mission design perspective, both approaches are useful: on the one hand, given a desired Lunar orbit payload you can determine the necessary launch mass, however, as most approaches in the literature start with an expected launch mass and calculate the payload that can be delivered to Lunar orbit, the later is considered for this problem.

Table 7.4: Backwards propagation with free m_{arr} and Ω_{arr} for the GTO-LPO transfer.

Objective	Method	Time (days)	Propellant (kg)	m_{dep} (kg)	Ω_{arr} (°)
Time	PSO fixed	223.18	85.88	350.00	140.49
	PSO free	209.72	80.70	350.00	156.90
	RL Q-law	217.43	83.67	349.87	183.31
	RL Q-Jac+CC	226.12	86.86	349.85	165.46
Mass ($\leq$ 250 days)	PSO fixed	250.00	60.66	350.00	0.00
	PSO free	249.80	59.34	350.00	16.30
	RL Q-law	249.30	73.38	349.99	116.93
	RL Q-Jac+CC	249.82	70.85	348.79	74.35

Table 7.4 provides a summary of both the time and mass-optimal simulations. Owing to the increased degrees of freedom, the benchmark "PSO fixed" and "PSO free" results are better than the previous section. The time-optimal "PSO free" solution of 209.72 days is only 15.47 days (8.0%) off the optimal result in Shannon *et al.* [70] and a 7.15 day (3.3%) improvement on the fixed geometry problem. Similarly, the mass-optimal results can improve on the fixed geometry problem by 9.32 kg (13.6%) and within 0.6 kg (1.0%) of the optimal solution.

Surprisingly, the RL results are very poor. The best time-of-flight is only 217.43 days and best mass-optimal solution 70.85 kg, a long way off the optimal solutions provided by Shannon *et al.* [70] and from the PSO solutions. This led to the forwards propagation approach considered below, with much improved results. However, these results are included here as an indication of the struggles RL can have. The need to learn Ω_{arr} and m_{arr}, and subsequently handle the constraint on the GTO departure mass, m_{dep}, proved too difficult for the current RL setup. It is thought m_{arr} proves particularly challenging, as the spacecraft mass directly affects the control acceleration available at each point along the transfer. Hence, any change in this value has a highly non-linear effect on the state-action pairs for the remainder of the backwards propagated transfer.

7.5 Forwards Propagation

Due to the challenges discussed in the previous section, a forward propagation from Earth GTO to Moon LPO is considered. This removes the need to define a patch point *a priori*, and instead only requires a criteria to switch betweenn the ECI and MCI legs - see Algorithm 5. Conventionally, Lyapunov controllers are not suitable for this sort of approach as they are unable to rendezvous with a particular point on an orbit. However, using the RL setup this issue can be circumnavigated by introducing a penalty in the cost function as shown in Eqs. (7.2) and (7.3). The departure epoch is now set to 20^{th} December 2002.

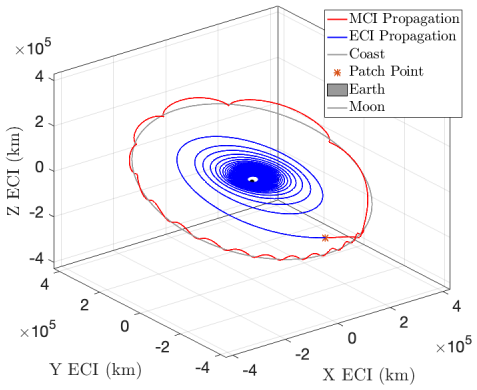
Table 7.5: Forwards propagation with free Ω_{dep} from GTO-LPO.

Objective	Method	Time (days)	Propellant (kg)	Ω_{dep} (°)
Time	PSO free	204.88	77.08	-
	RL Q-law	207.01	77.35	45.24
	RL Q-Jac+CC	208.47	79.69	341.65
Mass	PSO free	249.80	59.34	-
($\leq$ 250 days)	RL Q-law	221.82	65.00	358.46
	RL Q-Jac+CC	245.18	61.44	359.99

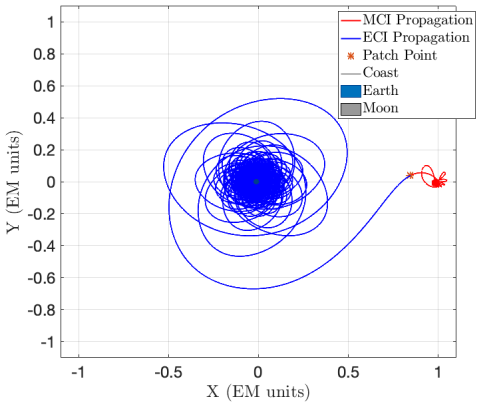
Table 7.5 provides a summary of both the time and mass-optimal results. The time-optimal RL simulations are now only 12.76 days (6.6%) away from the optimal solution using direct collocation. The best mass-optimal solution is worse than the PSO solutions obtained in the previous section. In fact, unlike the time-optimal simulations, the mass-optimal results improved when four, rather than three, deterministic trajectories were used, separating the Ω_{dep} exploration from the other actions in the ECI network. This suggests the RAAN strongly influences the other actions and needs further investigation. Overall, it appears to struggle to trade a longer time-of-flight for propellant saving. The current method could be improved such that the engine is only allowed to switch off when $\boldsymbol{a}_p$ is causing $\dot{Q} < 0$, information that is already available thanks to the cone-clock implementation. This would ensure the controller can always counteract any perturbing acceleration which is driving it away from the target orbit.

Figure 7.2 shows the time-optimal low-thrust trajectory from GTO-LPO generated using both the RL Q-law and RL Q-Jac+CC approaches. In Figs. 7.2a, 7.2b, 7.2c and 7.2d the trajectory can be seen in the ECI and Earth-Moon rotating frames respectively. The ECI leg (in blue) spirals out from GTO and targets the Keplerian orbit of the Moon. When it is within $\boldsymbol{r}_{SOI}$ of the Moon, it transitions to the MCI frame (in red) where it targets the LPO. Figure 7.2e shows the evolution of the weights with respect to time. The transition from ECI to MCI is shown with a dashed vertical line.

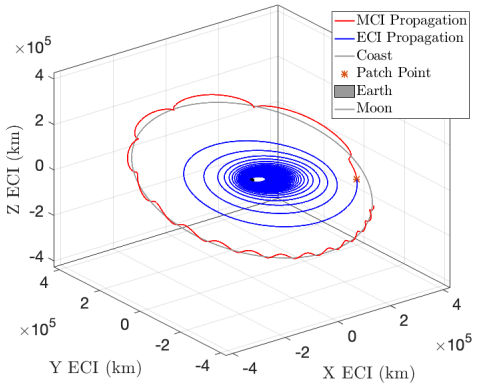
It is not required for the weights to be continuous across this boundary. In order to ensure the state and control direction is continuous, the control is set to 0 for the time-step across the boundary. Figure 7.2f indicates the cone-clock angles. The solid black



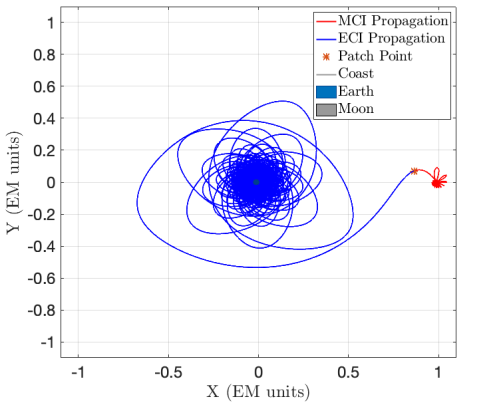
(a) RL Q-law trajectory in ECI frame



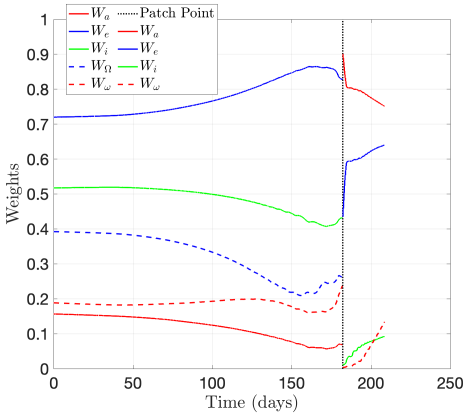
(b) RL Q-law trajectory in rotating frame



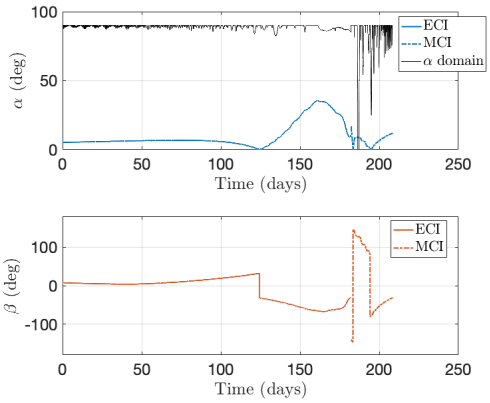
(c) RL Q-Jac+CC trajectory in ECI frame



(d) RL Q-Jac+CC trajectory in rotating frame

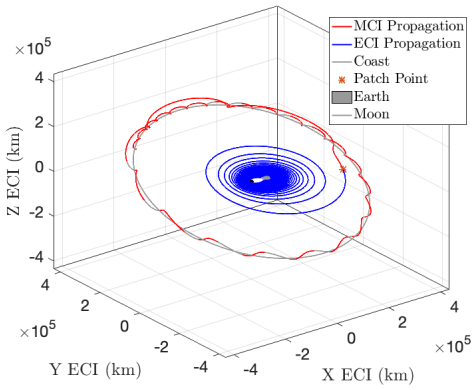


(e) RL Q-Jac+CC weights

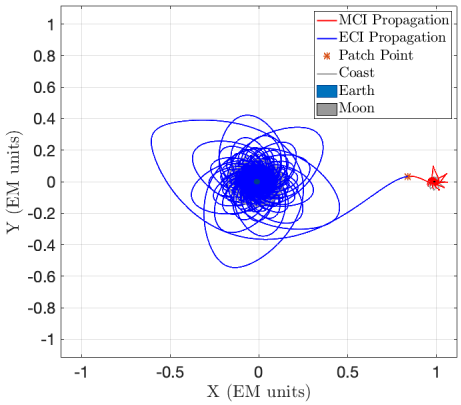


(f) RL Q-Jac+CC cone-clock angles

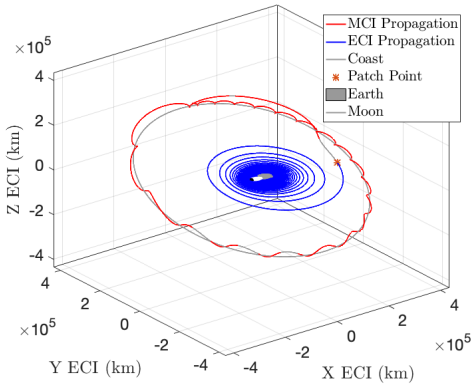
Figure 7.2: Time-optimal low-thrust trajectory from GTO-LPO generated using the RL Q-law and RL Q-Jac+CC approaches. Both the ECI and MCI legs were forward propagated.



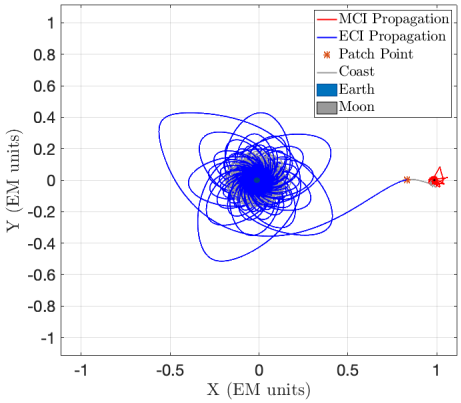
(a) RL Q-law trajectory in ECI frame



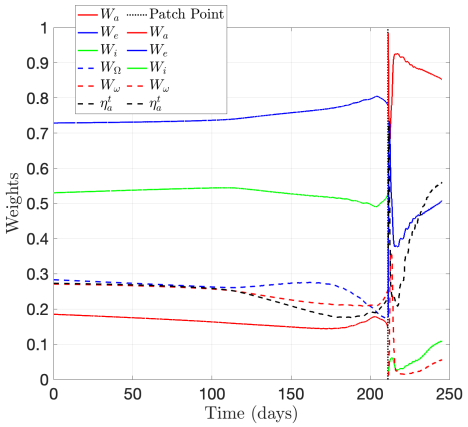
(b) RL Q-law trajectory in rotating frame



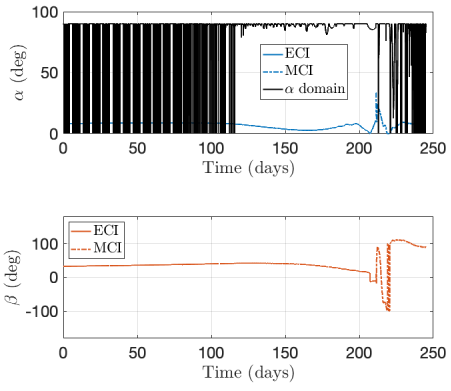
(c) RL Q-Jac+CC trajectory in ECI frame



(d) RL Q-Jac+CC trajectory in rotating frame



(e) RL Q-Jac+CC weights



(f) RL Q-Jac+CC cone-clock angles

Figure 7.3: Mass-optimal low-thrust trajectory from GTO-LPO generated using the RL Q-law and RL Q-Jac+CC approaches. Both the ECI and MCI legs were forward propagated.

line indicates the upper bound on α given by Eq. (4.10). When this boundary is at $\pi/2$ the perturbing acceleration is aiding the decrease in Q. Any value between 0 and $\pi/2$ indicates sufficient control authority to ensure $\dot{Q} < 0$ and hence alpha is bounded within the domain. The blue curve indicating α is the output of the network rather than the value used in the control. The minimum value between this and upper bound of the domain is used to compute the control direction. When the boundary is 0 it is not possible to ensure Lyapunov stability as the perturbing acceleration is too large. This appears to be the case in the region around the patch point, where the Keplerian approximation is expected to deviate significantly from the true dynamics as the third-body gravitational attraction increases. This region appears to be when the control authority is too low to guarantee stability.

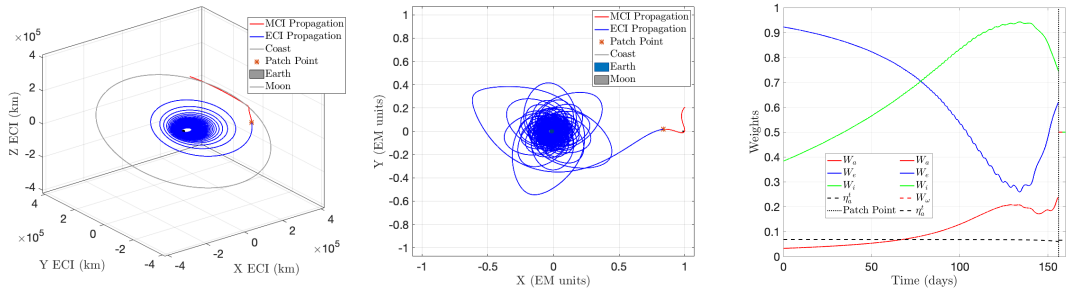
Figure 7.3 shows the mass-optimal low-thrust trajectory from GTO-LPO generated using both the RL Q-law and RL Q-Jac+CC approaches. In Figs. 7.3a, 7.3b, 7.3c and 7.3d the trajectory can be seen in the ECI and Earth-Moon rotating frames respectively. Figure 7.3e shows the evolution of the weights with respect to time. Again, it is not required for the weights to be continuous across the ECI to MCI boundary. Figure 7.3f shows the cone-clock angles and the solid black line indicates the upper bound on α given by Eq. (4.10). Initially it appears there are many regions in LEO where the control authority is not large enough to ensure Lyapunov stability. However, this is not the case and instead is a by-product of the way α is calculated. If the engine is off then α is not calculated and the boundary is set to 0.

7.6 Avenues for future improvement

The are two main challenges with the forwards propagation seen in the previous section: encouraging rendezvous with the Moon and ensuring the spacecraft remains captured around the Moon. In this section, two avenues of future work are presented, offering further promise to the improve the convergence of the trajectory.

In Section 7.5 the main difficulty comes from updating the ECI network. This is a due to the overall cost function's reliance on the ECI network rendezvousing with the Moon, and perhaps more importantly the MCI trajectory remaining captured, which is also heavily dependent on the ECI trajectory - see Eqs. (7.2) and (7.3). Both of these events could be considered non-smooth: a small change in the ECI trajectory could easily result in missing the Moon or providing too much energy to the MCI network at the patch point, resulting in an escape from the Lunar SOI. As such, encouraging the ECI network to update and improve its performance is very difficult.

To demonstrate this, Figs. 7.4 shows the trajectory and evolution of the weights $\boldsymbol{W}$ for an ECI policy where the only objective is too rendezvous with the Moon. The time-of-flight is 155.97 days, significantly faster than previous ECI legs. The issue with the above simulation is that although it arrives much earlier than the time-optimal result



(a) Trajectory in ECI frame (b) Rotating frame (c) Weights ECI Network

Figure 7.4: Time-optimal low-thrust trajectory from GTO the Lunar SOI generated using the RL Q-law approach. Both the ECI and MCI legs were forward propagated, although only the ECI leg is used during training, and the MCI is added a posteriori.

from before, the MCI leg is unable to converge, and escapes very quickly. A solution is needed to ensure convergence.

Thus, first a new convergence criteria is applied to the ECI trajectory, ensuring its energy integral of motion is less than the $L1$ point. Secondly, a new temporary target orbit is provided, using dynamical systems theory to provide more insight into the dynamical environment. In addition, a two-body energy convergence criteria is proposed as an to aid the MCI network. Due to time constraints, the following avenues have only been investigated for the time-optimal transfers.

7.6.1 Integral of Motion

In the CR3BP, the Jacobi integral of motion provides a valuable insight into the state within the system. This is related to the energy of a particular state within the CR3BP via $C = -2E$ and if no perturbing forces are acting on the system, this will remain constant - see [53]:

$$C(x, y, z, \dot{x}, \dot{y}, \dot{z}) = -\left(\dot{x}^2 + \dot{y}^2 + \dot{z}^2\right) - 2\bar{U}. \tag{7.4}$$

Here $x, y, z, \dot{x}, \dot{y}, \dot{z}$ refer to the position and velocity of the spacecraft in the planar CR3BP and $\bar{U}$ is given by

$$\bar{U} = -\frac{1}{2}\left(x^2 + y^2\right) - \frac{\mu_p}{r_p} - \frac{\mu_s}{r_s} - \frac{1}{2}\mu_p\mu_s, \tag{7.5}$$

and represents the effective potential of the spacecraft.

This can be used to define realms of possible motion in the CR3BP, where zero velocity curves indicate regions which are out-of-bounds for a particular energy - see Fig. 7.5 taken from [53]. Using this, it is clear that if the spacecraft is to remain captured around the Moon, it needs to be inside the Lunar SOI ($\boldsymbol{r} < \boldsymbol{r}_{SOI}$) and have energy $E < E_{L1}$ where E_{L1} refers to E_1 in Fig. 7.5, the energy of the $L1$ equilibrium point, and when the bottleneck closes.

Using this criterion, a stricter convergence criteria on the ECI network can be used,

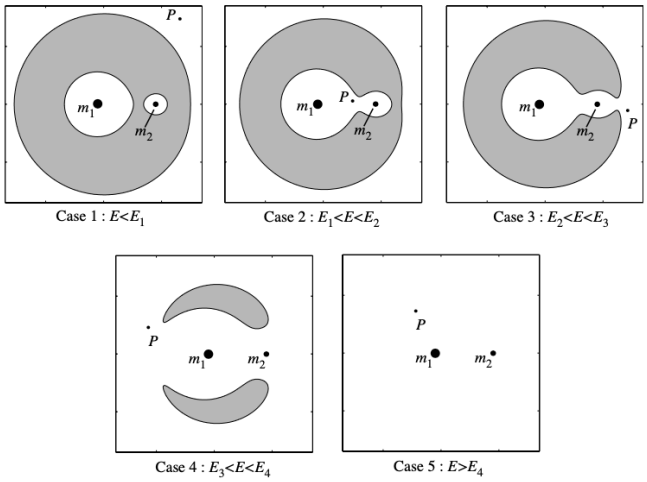


Figure 7.5: Realms of possible motion in the CR3BP taken from the book of Koon [53].

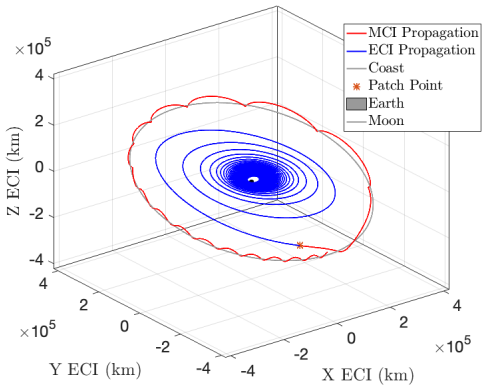
such that when the switch from ECI to MCI occurs, the MCI network will remain inside the Lunar SOI. Table 7.6 shows the results of this approach for a PSO Q-law and an RL Q-law, compared to the previous criterion. However, it appears this additional criterion makes it even more challenging for the RL approach to converge, a clear limitation of the current implementation. Even when only prioritising the ECI leg, the overall time-of-flight is approximately one Lunar month behind the optimal solutions. Bridging this one-revolution gap appears too challenging in the current setup. It is interesting to note the ECI prioritised trajectory has a better ECI time-of-flight, but overall this results in a worse total time-of-flight because it puts the MCI leg in a more challenging transfer. Note in this instance because of the ensured capture around the Moon, the MCI policy can be trained *a posteriori*.

Table 7.6: Forwards propagation using $\boldsymbol{r} < \boldsymbol{r}_{SOI}$ and $E < E_{L1}$ convergence criteria from GTO-LPO.

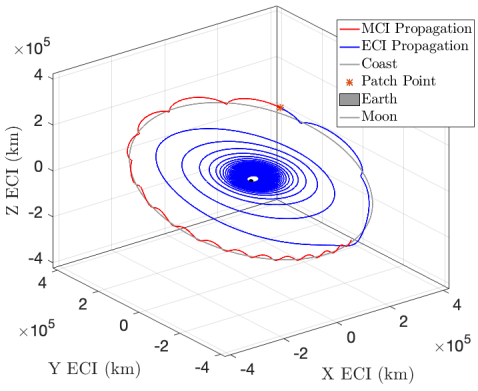
Convergence	Method	Time (Total) (days)	Time (ECI) (days)	Time (MCI) (days)
$\boldsymbol{r} < \boldsymbol{r}_{SOI}$ $E < E_{L1}$	PSO free	206.11	187.31	18.80
	RL Q-law	230.73	203.58	27.31
	RL Q-law (ECI)	231.22	203.32	27.90
$\boldsymbol{r} < \boldsymbol{r}_{SOI}$	PSO free	204.88	176.62	28.26
	RL Q-law	207.01	177.78	29.23
	RL Q-law (Seeded)	202.36	176.61	25.74

An interesting addition here is the possibility of seeding the RL simulations from the PSO simulations. As can be seen, this results in a near 2 day improvement in time-of-flight, although the ECI leg appears totally unchanged and the MCI leg is the only improvement. Hence, seeding from the PSO solution does not aid the RL in overcoming the convergence challenges for the ECI network during training.

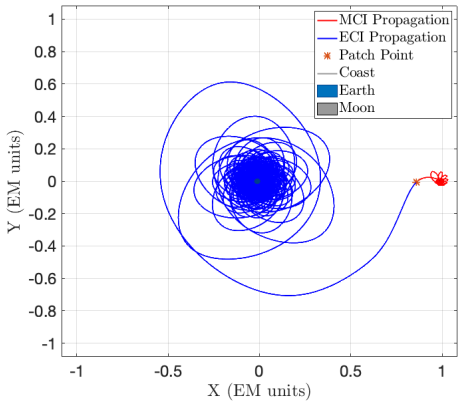
Figure 7.6 compares the PSO Q-law trajectories using the different convergence criteria on the ECI network. On the left-hand side, the convergence is $\boldsymbol{r} < \boldsymbol{r}_{SOI}$ only, and



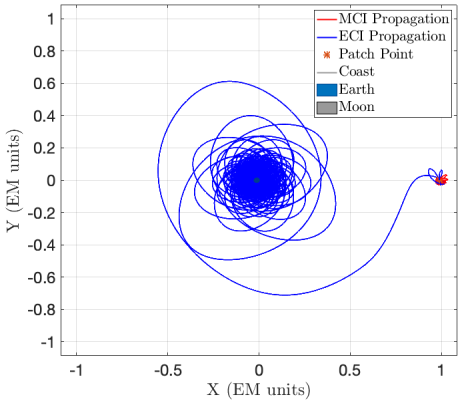
(a) $r < r_{SOI}$ Trajectory in ECI frame



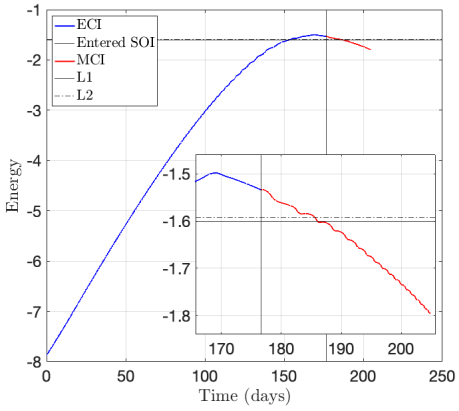
(b) $r < r_{SOI}$ and $E < E_{L1}$ Trajectory in ECI frame



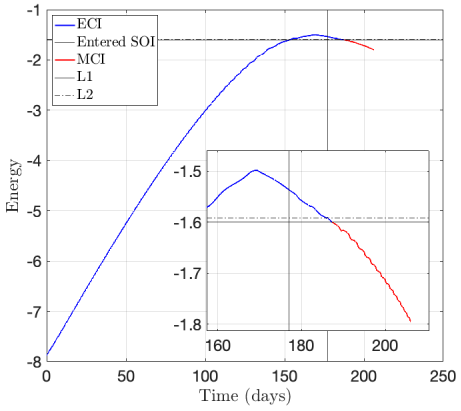
(c) $r < r_{SOI}$ Trajectory in rotating frame



(d) $r < r_{SOI}$ and $E < E_{L1}$ Trajectory in rotating frame



(e) $r < r_{SOI}$ Energy



(f) $r < r_{SOI}$ and $E < E_{L1}$ Energy

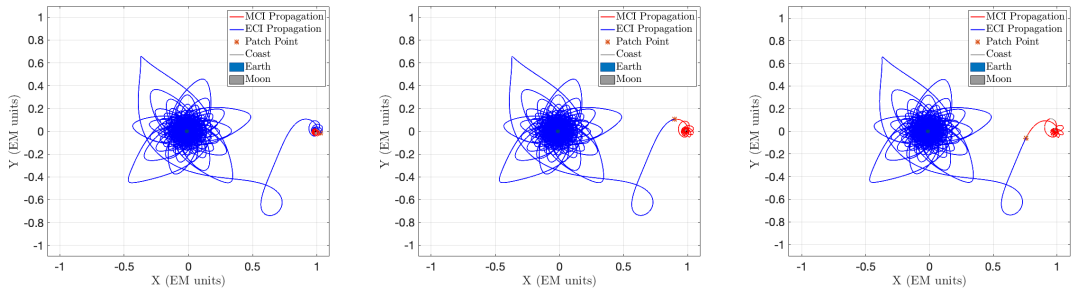
Figure 7.6: Time-optimal low-thrust trajectory from GTO-LPO generated using the PSO Q-law approach. Here the difference between the convergence criteria $r < r_{SOI}$ in Fig. 7.6e and $E < E_{L1}$ in Fig. 7.6f is compared.

results in a 204.88 day transfer. On the right, an additional criteria also requiring $E < E_{L1}$ is used, resulting in a 206.11 day transfer. Figures 7.6e and 7.6f show the energy integral of motion for both trajectories. In Fig. 7.6e it is clear that the switch from ECI to MCI occurs at the $\boldsymbol{r} < \boldsymbol{r}_{SOI}$ point, when in fact $E > E_{L1}$. In Fig. 7.6f, the spacecraft continues using the ECI policy until $E < E_{L1}$.

7.6.2 $L1$ Stable Manifolds and the Two-body Energy

Although Section 7.6.1 demonstrates a possible approach to ensuring the MCI network remains captured, it is still difficult to encourage the ECI network to converge, and better performance is still found without the additional $E < E_{L1}$ criteria. Hence, an alternative approach is sought.

Two key issues were identified as possible bottlenecks to the current approach. Firstly, a temporary target orbit is required for the ECI network, and this could be resulting in poor entrance trajectories into the MCI frame. Secondly, currently the MCI network is only switched on inside the Lunar SOI and this might be too late for it to affect the transfer significantly. Hence, is it possible to use knowledge of the dynamical environment to enable a better entrance to the Lunar SOI, and to make the most out of the MCI network.



(a) $\boldsymbol{r} < \boldsymbol{r}_{SOI}$ and $E < E_{L1}$ (b) $\boldsymbol{r} < \boldsymbol{r}_{SOI} = 50,000km$ (c) $\boldsymbol{r} < \boldsymbol{r}_{SOI} = 100,000km$

Figure 7.7: Low-thrust trajectories from GTO-LPO using a classical Q-law approach. All three use the convergence criteria $\boldsymbol{r} < \boldsymbol{r}_{SOI}$ and $E < E_{L1}$ for the ECI leg. They differ in how far they backtrack along the ECI trajectory in order to switch to the MCI leg.

However, it is not possible to activate the MCI network in a meaningful sense anywhere along the trajectory as demonstrated in Fig. 7.7, as it is not guaranteed that the trajectory will rendezvous with the Moon at the switching point. In addition, given the Lyapunov control laws are written in a two-body dynamical environment, it only makes sense to activate them when there are in a closed ($e < 1$) orbit, at least in terms of osculating orbital elements.

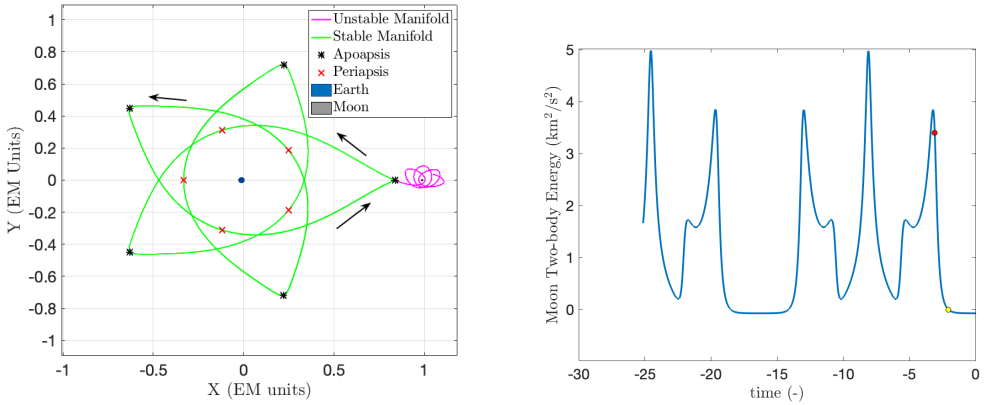
Hence, for the remainder of this section it is proposed to target a new temporary orbit for the ECI leg, and use the two-body energy as a criterion for activating the MCI leg. This way, when the MCI network is activated, it will be in a closed osculating orbit around the Moon. Given this is a poor description of the dynamics, the controller is likely to escape again, at which point the ECI network is used again. The two-body energy is written

with respect to either the Earth (ECI frame) or the Moon (MCI frame) and is given by:

$$E_{2B} = -\frac{\mu}{2a}, \tag{7.6}$$

where μ is the gravitational constant for the central body and a the osculating semi-major axis.

In order to aid the entrance to the Lunar SOI and capture around the Moon, stable manifolds emanating from $L1$ in a similar fashion to [68] are used. Figure 7.8a shows possible temporary targets at the apoapsis points (black stars) and periapsis points (red stars). Figure 7.8b shows the two-body energy in the MCI frame along this trajectory in normalised time units. The yellow point corresponds to the first time $E_{2B} < 0$ whilst the red point indicates the previous periapsis.



(a) Manifolds Emanating from $L1$

(b) MCI Two-body Energy

Figure 7.8: Stable and unstable manifolds emanating from the first Lagrange point in the Earth-Moon CR3BP. Apoapsis points (black stars) and periapsis points (red stars) are shown, along with the two-body energy in the MCI frame and the location where $E_{2B} < 0$ first occurs. Provided by N. Baresi.

Targeting this stable manifold would allow the spacecraft to subsequently coast to the $L1$ point. If the energy is matched exactly, then it will not cross over. However, if addition energy is injected intentionally, then the boundary can be crossed. Table 7.7 shows the osculating orbital elements at the periapsis, apoapsis and $E_{2B} < 0$ point along the backwards propagated leg directly from $L1$.

Table 7.7: Osculating orbital elements at $L1$ and the periapsis, apoapsis and $E_{2B} < 0$ points along the backwards propagated leg directly from $L1$. Provided by N. Baresi.

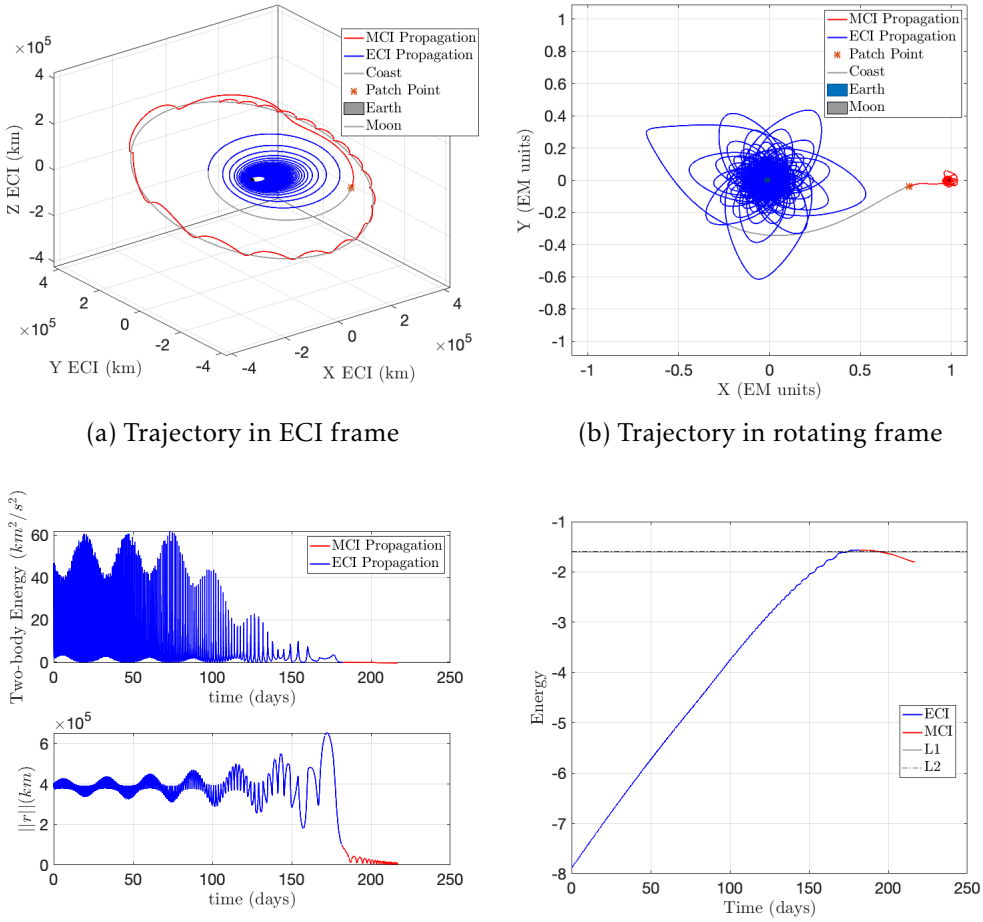
Target	a (km)	e (-)
$L1$	236812.512611744	0.380372847640193
$E_{2B} < 0$	219978.70833781	0.392811007693316
Periapsis	209227.200319762	0.394223656263373
Apoapsis	206509.936457307	0.408142428206408

Targeting the periapsis point resulted in the best performance. An additional 10,000

km was added to the target semi-major axis to provide sufficient energy to cross the $L1$ point and enter the Lunar SOI. Table 7.8 shows the results for the PSO Q-law, RL Q-law and RL Q-Jac+CC. The plots of the two-body energy are also shown. Note, in order to prevent switching whilst in the relative proximity of the Earth, a criterion that the two-body energy switch is only allowed within 100,000 km of the Moon is used.

Table 7.8: Time-optimal forward propagation using the $E_{2B} < 0$ convergence criteria from GTO-LPO, targeting the temporary apoapsis point on the stable $L1$ manifold.

Method	Time (Total) (days)	Time (ECI) (days)	Time (MCI) (days)
PSO free	212.79	181.66	31.13
RL Q-law	217.92	181.72	36.20
RL Q-Jac+CC	216.80	181.87	34.93



(a) Trajectory in ECI frame

(b) Trajectory in rotating frame

(c) E_{2B} and $\|\boldsymbol{r}\|$ in the MCI frame

(d) Weights

Figure 7.9: Time-optimal low-thrust trajectory from GTO-LPO generated using the RL Q-Jac+CC approach, targeting the osculating apoapsis point on the stable manifold emanating from $L1$.

Figure 7.9 shows the results for the the RL Q-Jac+CC simulations. The trajectory in Fig. 7.9b resembles that in Fig. 7.8a. The ECI controller is able to converge to the target apoapsis point, and then the spacecraft coasts for approximately 5 days before

$E < E_{2B}$ and the control switches to the MCI network. Clearly the spacecraft has sufficient energy to cross the bottleneck at $L1$, as indicated by Fig. 7.9d. The time-of-flight was 216.80 days. Accounting for the coast period, this puts the trajectory closer to the other results presented in the chapter, and future investigations could look at improving the temporary target orbit further to encourage better capture at the Moon. As mentioned in Section 2.2.2.3, the work by Jagannatha *et al.* [68] and Epenoy and Pérez-Palau [67] and Neves *et al.* [127] could provide further inspiration for appropriate target points.

7.7 Discussion and Summary

This chapter introduced a Reinforced Lyapunov Controller for designing low-thrust spiral transfers to Lunar space. It extends previous work on ensuring stability in the two-body environment by using the novel cone-clock angle approach to free the control direction whilst providing bounds to ensure stability in the three-body problem. This freedom allows the approach to decide if the perturbing accelerations experienced by the control are aiding to decrease the Lyapunov function value or not, and exploit them if possible. A RL framework with two actor-networks, one in the ECI and the other in the MCI, allow for forward propagation from GTO-LPO. Rendezvous with the Moon is possible thanks to the RL framework, and the initial geometry is also learnt to facilitate more optimal transfers. In addition, this approach can ensure stability in the CR3BP as long as the control authority is large enough.

Investigations for time and mass-optimal transfers from a GTO-LPO show the Reinforced Lyapunov Controller is able to compete with conventional evolutionary algorithm methods. At best, it is only 12.76 days (6.6%) and 2.7 kg (4.6%) away from the time- and mass-optimal solutions available in the literature. When seeded with the best PSO solution, this is reduced to 8.11 days (4.2%). Initially, three different approaches were considered: backwards propagation starting at the with fixed and free geometry, and forwards propagation starting at the GTO. For the fixed geometry the Reinforced Lyapunov Controller significantly outperforms the PSO. However, it struggles when the geometry is freed due to the non-linear constraint imposed on the departure mass at GTO. Hence, a forwards propagated approach is also presented with much greater success.

However, issues such as the difficulty improving the ECI network and handling the initial geometry remain. Possibilities of using a new convergence criteria based on the energy integral of motion are explored, along with target points on stable manifolds emanating from $L1$. In the future it is hoped to improve the forwards propagated model and extend it to higher-fidelity dynamics including the Sun's gravity and the spherical harmonics of the Earth. Initial results indicate the current PSO Q-law improves from 204.88 days to 202.70 days when changing to the full ephemeris model.

Chapter 8
Conclusions

The goal of this thesis was to investigate methods for improving the performance of heuristic control laws and utilising RL in trajectory design. The proposed approach combines Lyapunov control theory with RL techniques, allowing one to eliminate the drawbacks from each: the sub-optimality of Lyapunov controllers and the unknown stability of RL methods. The novelties introduced during this PhD were divided into five chapters. The first chapter introduces the fundamentals, including a novel state-weight dependence and analytical Jacobian for ensuring Lyapunov stability. The next two chapters consider the impact of this Reinforced Lyapunov Controller in unknown and uncertain dynamical environments, whilst the final two chapters stress the limitation of the control approach by increasing both the control acceleration and the dynamical perturbations. In this chapter, the main conclusions of the thesis are summarised and potential limitations and plans for future work are discussed.

8.1 Summary

In this section, the main objectives of this PhD project are revisited and compared to the contributions from each chapter, demonstrating how they were addressed. The objectives and the chapters in which they were addressed are as follows:

- To develop optimal state-dependent Lyapunov control laws for trajectory design and on-board guidance: Chapter 3.
- To investigate the stability implications of combining RL with Lyapunov control laws: Chapters 3 and 4.
- To improve the performance of Lyapunov control laws in the presence of perturbing accelerations and high-fidelity dynamics: Chapters 4 and 7.
- To assess the robustness of these methods under uncertain and stochastic environments, and pave the way for future on-board use: Chapter 5.
- To quantify the limits of these techniques, both in terms of thrust magnitude and dynamical environments: Chapters 6 and 7.

Chapter 3 introduced a novel Reinforced Lyapunov Controller framework. The proposed approach combines Lyapunov control theory with RL techniques. It presented a state-dependent controller (Objective 1) which enforces stability without compromising

optimality through the Jacobian of the state-dependent weights (Objective 2). The necessary Jacobian is available analytically through an actor network, ensuring the system remains closed-loop. After the introduction of state-dependent parameters, the Jacobian also ensures expressions for the effectivity threshold more accurately represent the efficiency of thrust locations along the orbit. The novelties were the state-weight dependence, the stability considerations and the analytical expression for the Jacobian.

Chapter 4 investigated the robustness of the Reinforced Lyapunov Controller subject to J_2 and 3^{rd}-body perturbations and eclipse effects (Objective 3). Training with these was not an issue for the closed-loop nature of the approach and a strong degree of optimality was retained. A novel cone-clock approach was introduced to allow a greater degree of freedom to the control direction and enable the controller to exploit the existence of the perturbations. It also ensures Lyapunov stability is maintained whilst enabling a greater domain of possible control directions (Objective 2).

Chapter 5 investigated the robustness of the Reinforced Lyapunov Controller in uncertain and stochastic environments (Objective 4). Due to the closed-loop nature of both the underlying Lyapunov controller and the trained actor network, the Reinforced Lyapunov Controller is robust to uncertainties in OI, OD and thruster EX errors. This was successfully investigated for a fixed control and for a free control approach, introducing an interpolation approach to allow easier training in the presence of stochastic errors.

In Chapter 6 the Reinforced Lyapunov Controller was modified to produce finite-burn manoeuvres, which in turn could be used to inform the optimal locations of impulsive manoeuvres. Operational constraints were also taken into account, restricting the maximum ΔV for a burn, and the interval between successive burns. This provided a useful exercise pushing the boundaries of the approach developed (Objective 5).

Chapter 7 introduced a Reinforced Lyapunov Controller for designing low-thrust spiral transfers to Lunar orbit. An RL framework with two actor-networks, one in the ECI frame and the other in the MCI, allows for forward propagation from GTO-LPO. This provided an opportunity to push the boundaries of the Reinforced Lyapunov Control approach developed (Objective 5). The dynamical environment could no longer be approximated as two-body, and instead the perturbing acceleration of the third-body eventually takes over to become the main acceleration. This transition is firmly outside the understanding of the Lyapunov controller and demonstrated the extended capabilities thanks to the RL architecture.

As mentioned in Chapter 1, trajectory design tools are often judged on the following criteria: *Flexibility*, *Robustness*, *Speed*, *Accuracy*, *Automation* and *Optimality*. This thesis has tackled *Flexibility* by considering LEO-LEO, LEO-GEO, GTO-GEO and GTO-LPOs in a variety of dynamics. *Robustness* was investigated in the presence of perturbing accelerations, eclipse effects and stochastic errors. The results are always at least as good as the classical Lyapunov controller and almost always better than PSO Q-law, demonstrating *Accuracy*. From a *Speed* and *Automation* perspective, the resulting trained network

has the potential to be implemented on-board. The training is more computationally intensive, however, and in its current format is restricted to the on-ground use. Finally, in terms of *Optimality*, this depends on the transfer scenario. GTO-GEO transfers are $< 0.1\%$ and 1.75% from the time- and mass-optimal solutions; LEO-GEO transfers are better than the available comparisons; GTO-LPO simulations are 4.2% and 4.6% from the time- and mass-optimal solutions; and, finally, LEO-LEO are 3.1% and 3.8% from the time- and mass-optimal solutions.

Overall the resulting Reinforced Lyapunov Controller is closed-loop and lightweight, providing stable solutions that are more optimal than conventional Lyapunov Control techniques. In addition, it is applicable to many different scenarios, as shown by the variety of test cases, and suitable for both initial trajectory design and potential on-board use, thanks to its ability to remain robust to both mis-modelled dynamics and uncertainties. Whilst the computational cost has not been directly investigated, the separation between training and closed-loop evaluation produces a computationally efficient controller with attractive on-board characteristics, requiring neither an initial guess generation, reference trajectory, onboard iteration nor numerical integration to operate.

8.2 Limitations and Future Work

8.2.1 Chapter 3: Reinforced Lyapunov Controller

The approach developed in Chapter 3 uses COEs to compute the Lyapunov control. These are known to have singularity issues, which might be cause for concern in potential on-board implementation. In addition, the current actor-network architecture takes COEs inputs to provide the state-dependence for the weights. This was done deliberately to provide intuitive understanding of the behaviours of both the control law and the actor-network. However, for the approach to be considered robust, then these could easily be exchanged for a different set of orbital parameters, such as MEEs or Milankovich elements. In fact, formulations for the Q-law exist in MEEs and it is straightforward to re-derive in Milankovich elements, something Chang *et al.* [128] have done for a Lyapunov controller.

Deep-RL is becoming increasingly popular because it enables increasingly complex functional relationships between the input state and action to be modelled [29]. The single-layer approach adopted here allowed for increasing transparency on a NN's impact on the Lyapunov controller, and enabled an in-depth investigation on the state-weight Jacobian and the impact of the network structure. With hesitancy to implement large NNs on-board and their often black-box nature, the motivation at the start of the work was to investigate the impact of a shallow network. Once the results of this appeared to provide close to optimal solutions for the initial test cases, it was no longer the priority of the investigations. However, there is potential to exploit multi-layered NNs

or recurrent NNs within this Reinforced Lyapunov Controller framework. In the case of recurrent NNs it is unclear how the Jacobian might be constructed, but it should be possible, if not tedious, to obtain a semi-analytical expression for multi-layered NNs.

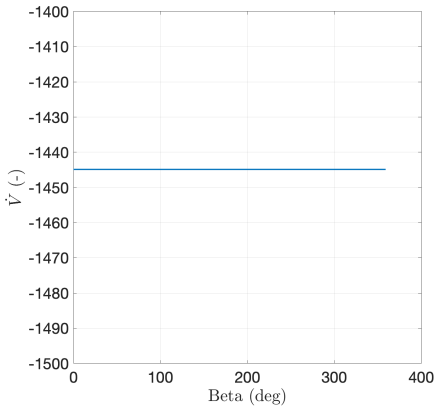
In line with these alternative RL approaches is the issue of tuning the RL hyperparameters. Many works, including those in astrodynamics, identify this as an issue (including [29, 30, 97, 98, 129]) as it can limit the performance of the resulting policy. Bayesian optimisation is fast developing as the go-to method for tuning these hyperparameters. In this work, as shown in Chapter 3, the NN hyperparameters are chosen based on the orbital environment and astrodynamics understanding. However, bayesian optimisation enables a search across hyperparameters to help tune them as best as possible - see Brochu *et al.* [130]. Bosanac *et al.* [101], Bonasera *et al.* [102] and Zavoli *et al.* [105] are already using bayesian optimisation to tune their RL policies in astrodynamics applications.

Devising Lyapunov control laws is an art; there is no automatic way to establish a Lyapunov function that guarantees stability while ensuring optimality. However, a recent work by Dai *et al.* [111] has tackled this problem by synthesising Lyapunov-stable NN controllers. Donti *et al.* [112] are also developing methods for enforcing robust control in NNs by using Lyapunov control laws as a constraint during the training. These approaches are entirely different to the one considered during this PhD study and provide an exciting opportunity to formalise the derivation of Lyapunov functions and provide stable control. Instead of utilising the same Lyapunov controller throughout and optimising it by making its parameters state-dependent using RL, they simply ensure that any control selected by the agent is one where a Lyapunov function could exist. In many ways it is synthesising Lyapunov functions for particular state-action pairs. This ensures that any control proposed by a trained network is Lyapunov stable, even if it is not optimal. This is a significant shift in mentality and opens the door to close the gap between Lyapunov control and optimal control methods.

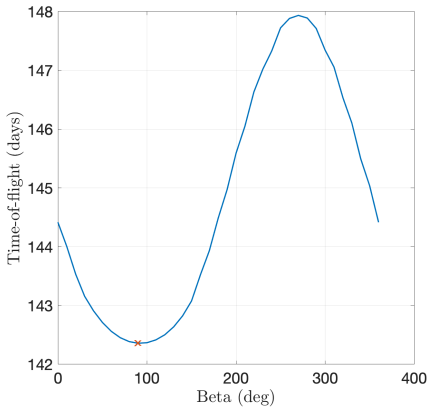
8.2.2 Chapter 4: Trajectory Design in the presence of Perturbations

Chapter 4 introduces the cone-clock approach in order to add additional freedom, the potential to exploit perturbations and ensure Lyapunov stability for longer. However, there are issues regarding the most suitable formulation of these angles, and the frequency at which they change. Currently it is not fully equipped to exploit the perturbations and whilst in some cases it performs very well, there is room for improvement. One possibility is to determine the clock angle automatically. The clock angle β lies in the plane perpendicular to $-\hat{\boldsymbol{p}}$, and as such has no instantaneous effect on the Lyapunov function V. However, this is only true for an instance in time, and as Fig. **??** shows, the value of β does affect the resulting transfer optimality and thus evolution of the Lyapunov function. Developing a method of informing the controller at a given instance in time the optimal

β would both reduce the complexity of the learning process.



(a) $\dot{V}$ as a function of β



(b) Time-of-flight for $\alpha = 5°$

In addition, Section 4.4 investigates the potential of targeting a moving target orbit. The main approximation is the lack of consideration for this in the control derivation. Consider a Lyapunov function

$$\dot{V}(\boldsymbol{X}) = \frac{1}{2}(\boldsymbol{X} - \boldsymbol{X}_T)^T \boldsymbol{W}(\boldsymbol{X} - \boldsymbol{X}_T), \tag{8.1}$$

where the state dynamics are given by $\dot{\boldsymbol{X}} = \boldsymbol{B}(\boldsymbol{X})[\boldsymbol{u} + \boldsymbol{a}_p(\boldsymbol{X})]$ and the targets are given by $\dot{\boldsymbol{X}}_T = \boldsymbol{B}(\boldsymbol{X}_T)[\boldsymbol{a}_p(\boldsymbol{X}_T)]$. Hence, taking the derivative of the Lyapunov function gives

$$\begin{aligned} \dot{V} &= \frac{\partial V}{\partial \boldsymbol{X}}\dot{\boldsymbol{X}} + \frac{\partial V}{\partial \boldsymbol{X_T}}\dot{\boldsymbol{X_T}} \\ &= \frac{\partial V}{\partial \boldsymbol{X}}\boldsymbol{B}(\boldsymbol{X})[\boldsymbol{u} + \boldsymbol{a}_p(\boldsymbol{X})] + \frac{\partial V}{\partial \boldsymbol{X_T}}\boldsymbol{B}(\boldsymbol{X}_T)[\boldsymbol{a}_p(\boldsymbol{X}_T)]. \end{aligned} \tag{8.2}$$

The second term is currently neglected in the simulations and control derivation, and undermines the Lyapunov stability of the proposed controller.

In Section 4.4 a transfer between two SSO orbits with the same shape but $\Delta\Omega = -10°$ is considered. Due to the presence of the secular J_2 perturbation, the trajectory which best conserves propellant mass utilises the difference in drift rate of orbits at different altitudes to enable the spacecraft to coast whilst the target orbit catches up with the spacecraft. This highlights the potential of intermediate target points, or *waypoints*, that could be incorporated into the current approach. This has been used with Lyapunov control laws in the Lunar environment by Peterson *et al.* [69], and can be used to provide astrodynamics assistance to the control law, or in a multi-target mission.

8.2.3 Chapter 5: Trajectory Design in the presence of Stochastic Errors

Chapter 5 highlighted the potential for the stochastic nature of the errors to complicate the learning process as the framework cannot easily distinguish stochastic errors from stochastic control actions. A particularly interesting subset of RL techniques known as

Meta-learning appears well suited to this problem [94, 95]. In Meta-learning, the concept is to teach the agent to act effectively on a series of task - as [29] put it, they "learn to learn". Here, different environmental dynamics are treated as a range of partially observable MDPs, and a policy is trained across multiple realisations of the uncertainty set. This has already been demonstrated in a variety of space applications and for Lunar landing problems [95], asteroid hovering scenarios [94] and guidance with integrated navigation [131], with great success. These examples use recurrent NNs, and as a policy experiences a different environment its hidden state stores and learns information specific to each environment, vastly increasing the range of robustness of the policy. By exploring these techniques further it could be possible to further extend the robustness of a Reinforced Lyapunov Controller. Shirobokov *et al.* [29] identify this as a promising future avenue for research.

8.2.4 Chapter 6: Trajectory Design for Approximating Finite-burn Manoeuvres

Chapter 6 takes an approach developed for low-thrust and continuous-thrust transfer and looks to re-purpose it as a finite-burn controller. Naturally, this has significant limitations but nonetheless demonstrates promising performance. The operational constraints include limiting the ΔV per burn and the coast arc between burns. Developing methods for including operational constraints more effectively in the learning process is important and could improves the performance of the possible state-dependence of the results. Future work should look at how the effectivity parameter can be better utilised in this scenario. Perhaps the relative effectivity η_r is a more appropriate measure than the absolute effectivity η_a. Further operational constraints can be considered in a similar fashion to [63, 115]. For example, shadow eclipse regions are not considered, which are clearly very influential particularly in the LEO environment. In addition, the engine model is very crude and should be improved to reflect the true evolution of the I_{sp} and thrust.

8.2.5 Chapter 7: Trajectory Design for Earth-Moon Spiral Transfers

Throughout the investigations the actor network input remained the current spacecraft state. However, in the future developing methods for incorporating more advanced concepts such as the Jacobi constant, or the difference between the current state and the target states should be investigated. This is likely to have impact on the state-weight Jacobian, but the derivation presented in Chapter 3 can be extended in this fashion. A different set of orbital elements, such as cylindrical coordinates or Milankovich elements could be appropriate in providing more information to both the controller and the RL architecture. For example, the Jacobi integral could be incorporated in the control law as a targeting objective, or it should be included as a network input.

Averaging methods could be used to provide further insights on the impact of the perturbations during long-duration many-revolution transfers. Liu *et al.* [132] use double averaging methods on both low-thrust and third-body perturbations, and devise analytical expressions for the secular variations of a spacecrafts orbit elements. Neves and Sánchez [127] also explore methods for including the third-body perturbing acceleration in the GVEs for low-thrust optimal control problems, expanding the Hamiltonian of the CR3BP to find an analytical expression for the third-body acceleration. Access to such expressions might enable their use within a Lyapunov control formulation similar Maddock and Vasile [65]. This is not done here because the purpose is to explore if the RL framework could be used to compensate for the inaccuracies of the Lyapunov formulation. However, in the future there is potential to utilise this information to devise more optimal control directions within the Lyapunov control formulation and to provide greater understanding of the perturbing accelerations. In addition, such analysis could also assist with Chapter 4, where the effect of the third-body perturbation for GTO-GEO and LEO-GEO was challenging to exploit.

As mentioned in Chapter 7, the work was done in the inertial CR3BP but ideally would be extended to the full fidelity model. These higher-fidelity dynamics could include the Sun's gravity and the spherical harmonics of the Earth. Initial results indicate the current PSO Q-law improves from 204.88 days to 202.70 days when changing to the full ephemeris model, most likely because the Moon's non-circular orbit aids the transfer. For instance, the pulsating nature of the 3rd-body perturbation could increase the energy of the ECI leg more rapidly and reduce the distance the spacecraft has to travel before entering the Lunar SOI. More work is also needed to handle the geometry of the problem and the patch point between ECI and MCI networks. The geometry problem might lend itself to a nested optimisation approach, where the internal optimisation has a fixed geometry and the outer layer is able to optimise the geometry alone.

Appendix A

Reinforced Basic Lyapunov Controller

Throughout this PhD thesis a RL approach has been combined with Lyapunov-based Q-law control law. This is optimal and stable in Keplerian dynamics, and robust to perturbations and stochastic errors. The resulting Reinforced Lyapunov Control approach offers potential for initial trajectory design, on-board autonomous guidance, and orbit reconfiguration. Future work can extend the methodology to investigate different Lyapunov control laws, varying RL architectures and the potential for on-board implementation. This Appendix is added to demonstrate how the Reinforced Lyapunov Controller can work when a different underlying Lyapunov control law is implemented. In Appendix A.1 the Lyapunov function is provided. Appendix A.2 demonstrates the performance in perturbed dynamics, and Appendix A.3 the performance in the presence of stochastic errors.

A.1 Basic Lyapunov control law

A vary basic Lyapunov control law can be devised following the implementation of the Q-law. More information can be found in Shirazi *et al.* [110], where its performance is compared to the Q-law using evolutionary algorithms. Using the same notation as in Section 2.2.2.3, the Lyapunov function is defined as:

$$V = (1 + W_P P) \sum_X W_X S_X (\delta X)^2, \quad X = a, e, i, \Omega, \omega, \tag{A.1}$$

where the current state is $\boldsymbol{X} = [a, e, i, \Omega, \omega]^{\mathrm{T}}$ and the target state $\boldsymbol{X}_T = [a_T, e_T, i_T, \Omega_T, \omega_T]^{\mathrm{T}}$. This contains the same weighting factor W_P, penalty function P, scaling factor S_X as the Q-law but without the maximum rate-of-change terms in the denominator. This removes the inbuilt time-to-go knowledge the Q-law benefits from. The control direction and rate-of-change of the Lyapunov function can all be calculated in a similar fashion to that described in Section 2.2.2.2.

A.2 Basic Lyapunov control law in the presence of Perturbations

In this Appendix, the results for the Reinforced Lyapunov Controller in the presence of perturbing accelerations are given. These are simulated for the same GTO-GEO and LEO-GEO transfers presented in Chapter 3. In Chapter 4 the results for the Q-law are

presented. Here, the same approach is used but the Q-law is replaced with a basic Lyapunov control law.

Table A.1: Comparison of time-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law).

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	163.32	251.80	2.639	-
	RL L-law	-	-	-	-	138.21	213.09	2.210	-
	RL L-Jac	-	-	-	-	139.03	214.36	2.224	-
	RL L-Jac+CC	-	-	-	-	153.13	236.06	2.463	-
J2	Classical	-	-	-	-	163.83	252.59	2.648	-
	RL L-law	138.29	213.21	2.211	0.55	138.47	213.49	2.214	0.53
	RL L-Jac	138.07	212.88	2.217	0.52	139.06	214.41	2.224	0.54
	RL L-Jac+CC	153.61	236.58	2.469	0.60	156.21	240.74	2.516	0.60
3rd Body	Classical	-	-	-	-	163.48	252.06	2.642	-
	RL L-law	138.99	214.30	2.223	0.49	138.55	213.61	2.215	0.49
	RL L-Jac	139.07	214.42	2.224	0.49	139.38	214.89	2.229	0.49
	RL L-Jac+CC	151.79	233.97	2.440	0.57	153.12	236.05	2.463	0.56
Eclipse	Classical	-	-	-	-	168.98	253.49	2.658	-
	RL L-law	139.02	212.12	2.199	-	141.58	215.25	2.204	-
	RL L-Jac	138.42	211.30	2.190	-	149.31	227.64	2.216	-
	RL L-Jac+CC	140.34	214.31	2.223	-	151.73	228.99	2.458	-

Table A.2: Comparison of mass-optimal GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Here a maximum allowed time-of-flight of 150 days is desired.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	-	163.32	251.80	2.639	-	-
	RL L-law	-	-	-	-	-	140.63	215.77	2.239	215.78	-
	RL L-Jac	-	-	-	-	-	147.94	229.12	2.341	225.04	-
	RL L-Jac+CC	-	-	-	-	-	150.43	229.25	2.388	230.31	-
J2	Classical	-	-	-	-	-	163.83	252.59	2.648	-	-
	RL L-law	137.68	212.28	2.201	212.28	0.53	141.04	216.63	2.249	216.63	0.51
	RL L-Jac	140.07	214.82	2.229	214.82	0.52	148.82	229.35	2.389	229.35	0.50
	RL L-Jac+CC	149.69	226.85	2.361	226.85	0.62	152.04	232.46	2.423	235.99	0.55
3rd Body	Classical	-	-	-	-	-	163.48	252.06	2.642	-	-
	RL L-law	140.68	216.59	2.248	216.59	0.50	140.93	216.26	2.244	216.26	0.50
	RL L-Jac	148.00	225.82	2.350	225.82	0.49	148.22	228.29	2.377	228.29	0.49
	RL L-Jac+CC	148.79	227.5	2.361	226.79	0.53	150.78	229.67	2.392	231.26	0.53
Eclipse	Classical	-	-	-	-	-	168.98	253.49	2.658	-	-
	RL L-law	141.03	213.88	2.218	213.88	-	141.58	215.25	2.233	215.25	-
	RL L-Jac	160.95	242.73	2.538	260.00	-	149.31	227.64	2.370	227.64	-
	RL L-Jac+CC	152.33	230.41	2.398	234.14	-	151.73	228.99	2.385	232.05	-

Table A.3: Comparison of time-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law).

		Trained with Perturbation				Trained without Perturbation			
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	212.59	227.99	6.819	-
	RL L-law	-	-	-	-	180.00	193.05	5.676	-
	RL L-Jac	-	-	-	-	181.14	194.27	5.715	-
	RL L-Jac+CC	-	-	-	-	182.06	194.91	5.736	-
J_2	Classical	-	-	-	-	212.78	228.20	6.826	-
	RL L-law	180.27	193.34	5.685	0.51	180.29	193.36	5.686	0.51
	RL L-Jac	181.84	195.02	5.740	0.55	181.37	194.51	5.723	0.52
	RL L-Jac+CC	181.81	194.91	5.736	0.60	182.82	195.75	5.763	0.64
3^{rd}-body	Classical	-	-	-	-	212.36	227.76	6.811	-
	RL L-law	179.97	193.01	5.675	0.49	179.99	193.04	5.676	0.49
	RL L-Jac	183.14	196.40	5.784	0.48	181.09	194.22	5.714	0.49
	RL L-Jac+CC	183.08	196.21	5.778	0.51	182.04	194.89	5.735	0.52
Eclipse	Classical	-	-	-	-	247.47	226.83	6.780	-
	RL L-law	214.04	190.32	5.588	-	216.43	191.51	5.627	-
	RL L-Jac	223.56	198.57	5.854	-	221.03	195.56	5.757	-
	RL L-Jac+CC	239.55	215.35	6.197	-	232.13	204.32	6.041	-

Table A.4: Comparison of mass-optimal LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Here a maximum allowed time-of-flight of 200 days is desired.

		Trained with Perturbation					Trained without Perturbation				
Perturbation	Method	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$	Time (days)	Prop (kg)	ΔV (km/s)	Cost	Fraction $\dot{V} < 0$
Keplerian	Classical	-	-	-	-	-	212.59	227.99	6.819	-	-
	RL L-law	-	-	-	-	-	199.57	186.68	5.472	186.68	-
	RL L-Jac	-	-	-	-	-	198.40	186.49	5.466	186.49	-
	RL L-Jac+CC	-	-	-	-	-	195.94	193.40	5.687	193.40	-
J_2	Classical	-	-	-	-	-	212.78	228.20	6.826	-	-
	RL L-law	199.70	187.14	5.487	187.14	0.52	199.91	186.90	5.479	187.07	0.49
	RL L-Jac	197.91	192.66	5.664	192.66	0.56	198.79	186.69	5.473	186.69	0.49
	RL L-Jac+CC	206.86	209.45	6.099	207.65	0.92	198.97	193.45	5.689	193.45	0.73
3^{rd}-body	Classical	-	-	-	-	-	212.36	227.76	6.811	-	-
	RL L-law	196.92	186.70	5.473	186.70	0.49	199.56	186.64	5.471	186.64	0.50
	RL L-Jac	199.78	189.42	5.560	189.45	0.49	198.38	186.45	5.465	186.46	0.49
	RL L-Jac+CC	199.95	201.39	5.945	201.61	0.51	195.88	193.36	5.686	193.36	0.59
Eclipse	Classical	-	-	-	-	-	247.47	226.83	6.780	-	-
	RL L-law	216.95	190.26	5.587	208.71	-	235.10	184.29	5.396	222.21	-
	RL L-Jac	225.54	199.77	5.893	227.44	-	249.89	186.35	5.462	240.12	-
	RL L-Jac+CC	232.23	202.87	5.993	237.71	-	240.96	192.22	5.649	236.41	-

A.3 Basic Lyapunov control law in the presence of Stochastic Errors

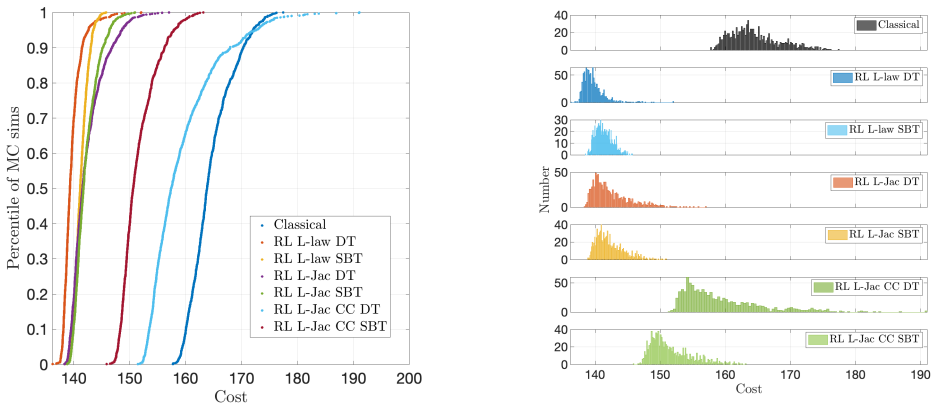
In this Appendix, the results for the Reinforced Lyapunov Controller in the presence of stochastic errors are given. These are simulated for the same GTO-GEO and LEO-GEO transfers presented in Chapter 3. In Chapter 5 the results for the Q-law are presented. Here, the same approach is used but the Q-law is replaced with a basic Lyapunov control law.

Table A.5: Time-optimal MC Simulations with stochastic disturbances for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

Errors	Method	Training	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile	Failures
OD+EX	Classical	-	163.32	164.64	3.727	158.69	163.69	174.60	0
	RL L-law	DT	138.20	139.61	1.516	138.17	139.23	145.07	0
		SBT	140.68	141.59	1.264	139.58	141.36	144.77	0
	RL L-Jac	DT	139.03	142.58	3.083	139.22	141.53	152.19	0
		SBT	143.82	147.40	2.849	143.49	146.80	157.22	0
	RL L-Jac+CC	DT	153.13	158.79	5.693	152.60	157.14	177.49	0
		SBT	146.69	149.78	3.085	146.64	148.78	161.15	0
OI+OD+EX	Classical	-	163.32	164.58	3.864	158.64	163.84	174.84	0
	RL L-law	DT	138.20	139.68	1.684	137.62	139.32	146.69	0
		SBT	140.25	141.30	1.255	139.02	141.15	144.73	0
	RL L-Jac	DT	139.03	142.45	2.993	138.83	141.58	152.05	0
		SBT	139.64	142.23	2.117	139.31	141.74	148.75	0
	RL L-Jac+CC	DT	153.13	159.35	6.157	152.40	157.42	179.65	0
		SBT	148.10	151.54	3.154	147.32	150.62	161.26	0

Table A.6: Mass-optimal MC Simulations with stochastic disturbances for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

Method	Training	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile	Failures	
OD+EX	Classical	-	272.72	276.91	11.227	258.68	274.60	307.20	0
	RL L-law	DT	215.78	217.40	3.826	214.18	216.93	222.70	0
		SBT	216.15	217.78	4.293	214.66	217.25	222.28	0
	RL L-Jac	DT	227.75	243.04	21.413	227.34	236.18	321.59	0
		SBT	220.67	225.00	5.583	218.87	223.64	247.10	0
	RL L-Jac+CC	DT	230.31	259.95	24.906	230.69	250.07	321.21	0
		SBT	227.37	235.64	10.947	226.87	232.34	279.89	0
OI+OD+EX	Classical	-	272.72	277.08	12.064	257.61	274.57	311.43	0
	RL L-law	DT	215.78	217.53	4.346	213.38	216.95	223.01	0
		SBT	216.44	218.56	4.427	214.30	218.02	223.82	0
	RL L-Jac	DT	227.75	244.26	23.034	226.53	236.41	343.61	0
		SBT	237.10	243.47	8.751	229.02	242.30	266.28	0
	RL L-Jac+CC	DT	230.31	260.38	25.295	229.88	251.42	324.82	0
		SBT	225.71	231.70	8.795	223.99	228.56	268.89	0



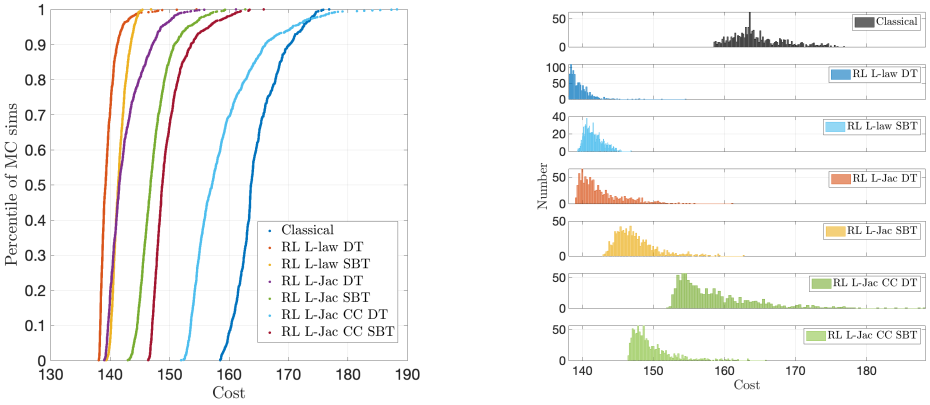
(a) Sorted Costs

(b) Histogram

Figure A.1: Time-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

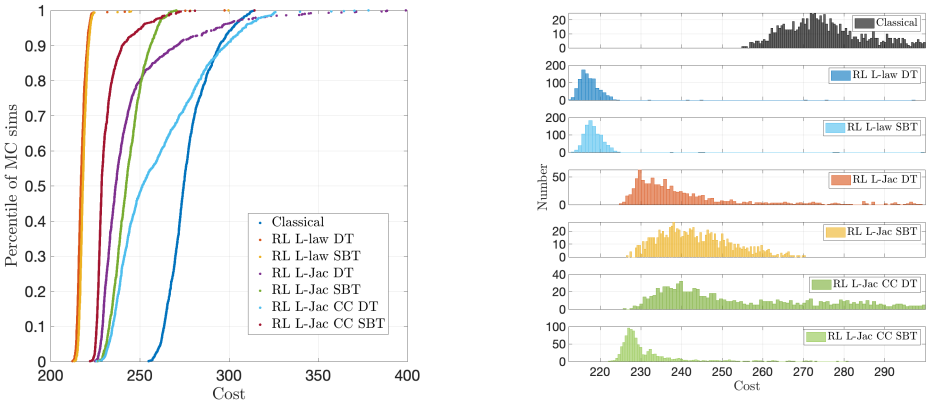
Table A.7: Time-optimal MC Simulations with stochastic disturbances for the LEO-GEO transfers using a Reinforced Lyapunov Controller. Nominal results are shown for comparison. Training is either DT or SBT.

Errors	Method	Training	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile	Failures
OD+EX	Classical	-	212.59	213.55	0.810	212.54	213.34	215.36	0
	RL L-law	DT	180.00	180.73	0.708	179.85	180.54	182.51	0
		SBT	179.86	180.63	0.706	179.73	180.44	182.35	0
	RL L-Jac	DT	181.14	184.22	3.200	181.08	183.11	195.33	131
		SBT	181.02	182.58	1.220	181.05	182.29	186.48	218
	RL L-Jac+CC	DT	182.06	186.68	3.788	182.11	185.44	197.33	247
		SBT	191.51	193.37	2.485	190.35	192.64	201.79	185
OI+OD+EX	Classical	-	212.59	213.56	0.854	212.22	213.40	215.56	1
	RL L-law	DT	180.00	180.75	0.752	179.57	180.60	182.65	0
		SBT	179.85	180.83	0.826	179.53	180.69	182.95	7
	RL L-Jac	DT	181.14	184.36	3.528	180.89	183.11	195.99	124
		SBT	180.86	183.05	4.902	180.43	181.63	207.70	71
	RL L-Jac+CC	DT	182.06	186.82	4.179	182.03	185.46	200.18	241
		SBT	183.90	186.74	1.820	184.12	186.41	193.50	3



(a) Sorted Costs (b) Histogram

Figure A.2: Time-optimal MC Simulations with stochastic disturbances (OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

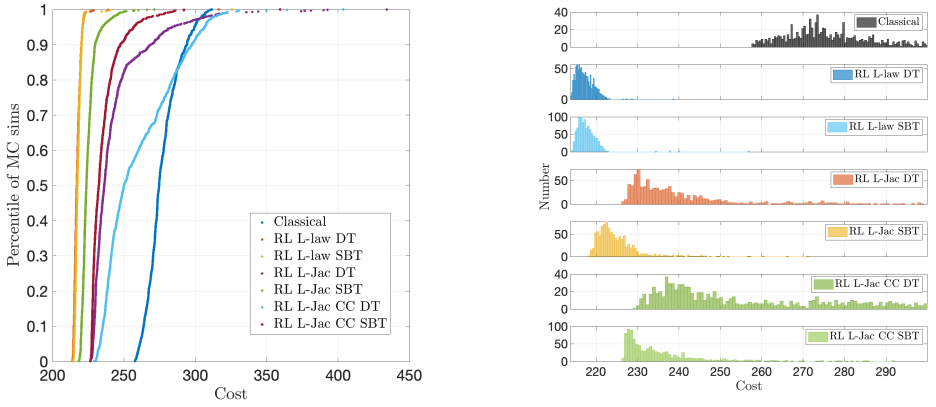


(a) Sorted Costs (b) Histogram

Figure A.3: Mass-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

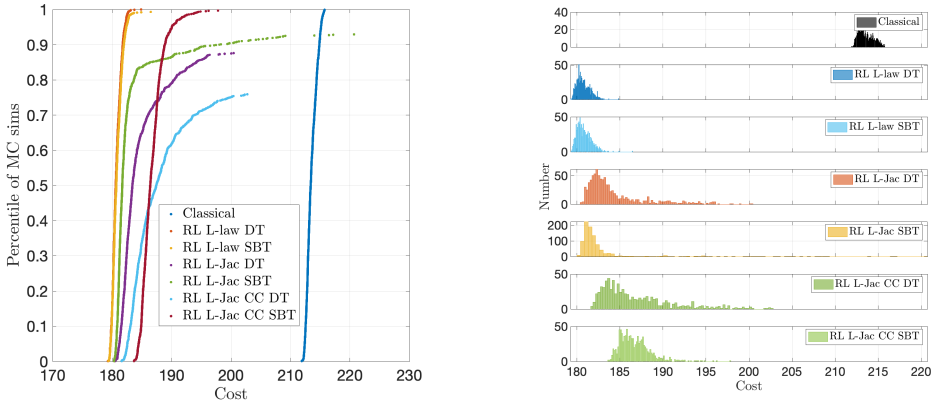
Table A.8: Mass-optimal MC Simulations with stochastic disturbances for the LEO-GEO transfers using a Reinforced Lyapunov Controller. Nominal results are shown for comparison. Training is either DT or SBT.

Errors	Method	Training	Nominal	Mean	σ	1^{st} Percentile	Median	99^{th} Percentile	Failures
OD+EX	Classical	-	241.77	243.81	1.768	241.69	243.33	247.75	1
	RL L-law	DT	186.68	186.92	1.663	185.24	186.26	191.90	0
		SBT	187.62	187.33	1.362	185.96	186.75	191.65	0
	RL L-Jac	DT	186.49	191.38	4.117	184.70	191.06	202.75	617
		SBT	188.00	188.98	2.216	186.90	188.08	196.08	112
	RL L-Jac+CC	DT	193.40	191.23	2.417	187.80	190.70	200.84	23
		SBT	200.51	202.96	2.896	200.30	202.21	217.34	0
OI+OD+EX	Classical	-	241.77	243.82	1.820	240.79	243.48	248.25	7
	RL L-law	DT	186.68	186.93	1.664	184.81	186.34	192.17	0
		SBT	186.75	188.10	2.956	184.92	187.07	197.37	0
	RL L-Jac	DT	186.49	191.15	4.292	184.25	190.57	203.10	610
		SBT	189.18	188.37	2.534	184.46	187.56	195.78	477
	RL L-Jac+CC	DT	193.40	191.38	2.470	187.90	190.86	200.86	28
		SBT	199.71	200.76	1.444	198.89	200.41	205.77	12



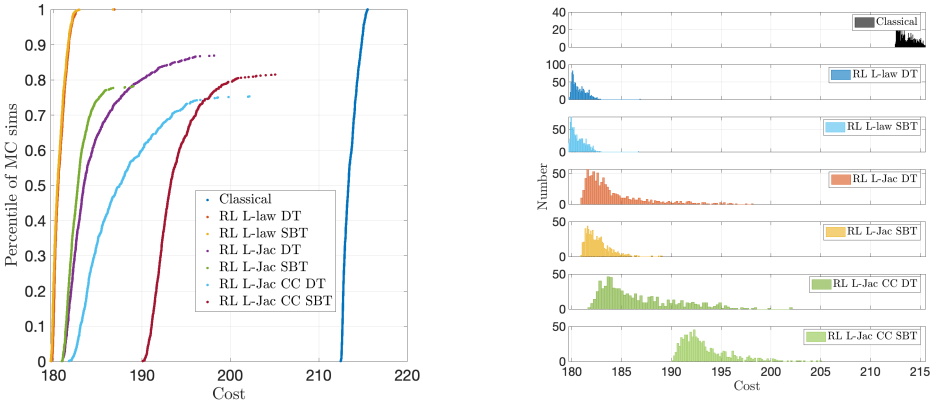
(a) Sorted Costs (b) Histogram

Figure A.4: Mass-optimal MC Simulations with stochastic disturbances (OD and EX) for the GTO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.



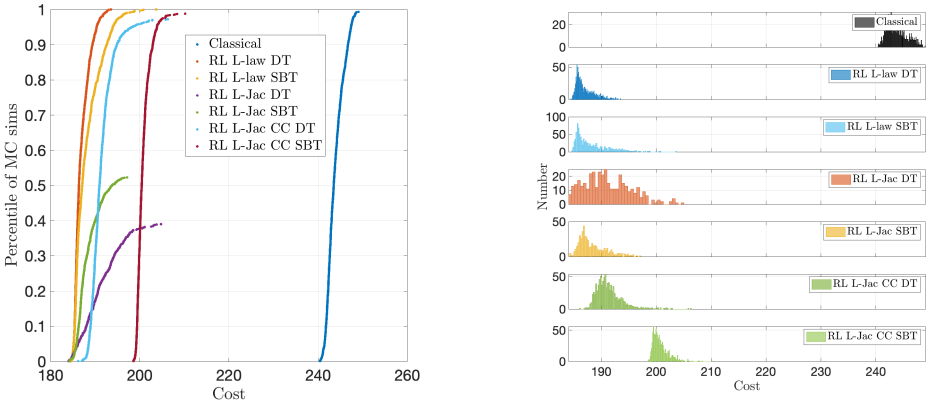
(a) Sorted Costs (b) Histogram

Figure A.5: Time-optimal MC Simulations with stochastic disturbances (OI, OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.



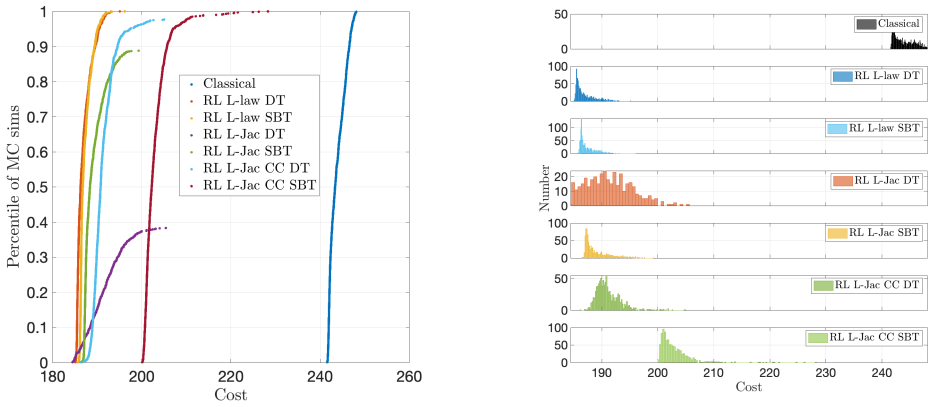
(a) Sorted Costs (b) Histogram

Figure A.6: Time-optimal MC Simulations with stochastic disturbances (OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.



(a) Sorted Costs (b) Histogram

Figure A.7: Mass-optimal MC Simulations with stochastic disturbances (OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.



(a) Sorted Costs

(b) Histogram

Figure A.8: Mass-optimal MC Simulations with stochastic disturbances (OD and EX) for the LEO-GEO transfers using a Reinforced Lyapunov Controller (basic Lyapunov control law). Nominal results are shown for comparison. Training is either DT or SBT.

References

[1] T.S. Kelso. Celestrak SATCAT Boxscore, 2019. URL https://celestrak.com/satcat/boxscore.php.

[2] Jonas Radtke, Christopher Kebschull, and Enrico Stoll. Interactions of the space debris environment with mega constellations—Using the example of the OneWeb constellation. *Acta Astronautica*, 131(May 2016):55–68, 2017.

[3] Dan R. Lev, Roger M Myers, and Kristina M Lemmer. The Technological and Commercial Expansion of Electric Propulsion in the Past 24 Years. *Proceedings of the 35th IEPC*, (October):IEPC–2017–242, 2017.

[4] Giuseppe D Racca, Bernard H Foing, and Marcello Coradini. SMART-1: The First Time of Europe to the Moon; Wandering in the Earth – Moon Space. *Earth, Moon, and Planets*, 85:379–390, 1999. doi: https://doi.org/10.1023/A:1017065326516.

[5] Eutelsat. All-electric Eutelsat 172B satellite set to transfer connectivity landscape in Asia-Pacific, 2017. URL https://www.eutelsat.com/en/news/press.html{#}/pressreleases/eutelsats-airbus-built-full\-electric-eutelsat-172b-satellite-reaches-geostationary-orbit-2208095.

[6] Valerie C Thomas, Joseph M Makowski, G Mark Brown, John F McCarthy, Dominick Bruno, J Christopher Cardoso, W Michael Chiville, Thomas F Meyer, Kenneth E Nelson, Betina E Pavri, David A Termohlen, Michael D Violet, and Jeffrey B Williams. The Dawn Spacecraft. In Christopher Russell and Carol Raymond, editors, *The Dawn Mission to Minor Planets 4 Vesta and 1 Ceres*, pages 175–249. Springer New York, New York, NY, 2012. ISBN 978-1-4614-4903-4. doi: 10.1007/978-1-4614-4903-4_10.

[7] Johannes Benkhoff, Jan van Casteren, Hajime Hayakawa, Masaki Fujimoto, Harri Laakso, Mauro Novara, Paolo Ferri, Helen R. Middleton, and Ruth Ziethe. BepiColombo-Comprehensive exploration of Mercury: Mission overview and science goals. *Planetary and Space Science*, 58(1-2):2–20, 2010. doi: 10.1016/j.pss.2009.09.020.

[8] Takanao Saiki, Jun Matsumoto, Osamu Mori, and Jun'ichiro Kawaguchi. Solar Power Sail Trajectory Design for Jovian Trojan Exploration. *Trans. JSASS Aerospace Tech. Japan*, 16(5):353–359, 2018.

[9] B Dachwald. Low-Thrust Trajectory Optimization and Interplanetary Mission Analysis Using Evolutionary Neurocontrol. *Deutscher Luft- und Raumfahrtkongress*, 2004.

[10] European Space Operations Center (ESOC). ESA's Annual Space Environment Report. Technical Report July, ESA Space Debris Office, 2019.

[11] Andrew Harris, Thibaud Teil, and Hanspeter Schaub. Spacecraft Decision-making Autonomy using Deep Reinforcement Learning. *AAS*, pages 1–19, 2019.

[12] Abolfazl Shirazi, Josu Ceberio, and Jose A. Lozano. Spacecraft trajectory optimization: A review of models, objectives, approaches and solutions. *Progress in Aerospace Sciences*, 102(August):76–98, 2018.

[13] David Morante, Manuel Sanjurjo Rivo, and Manuel Soler. A survey on low-thrust trajectory optimization approaches. *Aerospace*, 8(3), 2021.

[14] Bruce A. Conway. *Spacecraft trajectory optimization*. Cambridge University Press, jan 2010.

[15] D. F. Lawden. *Optimal Trajectories for Space Navigation*. Butterworths, London, 1963.

[16] Jean Kéchichian. Reformulation of Edelbaum's Low-Thrust Transfer Problem Using Optimal Control Theory. *Journal of Guidance, Control, and Dynamics*, 20(5): 988–994, 1997.

[17] Theodore Edelbaum. Propulsion Requirements for Controllable Satellites. *ARS Journal*, 31(8):1079–1089, aug 1961.

[18] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1967. ISSN 00765392.

[19] John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, jan 2010.

[20] Robert E. Pritchett. *Numerical Methods for Low-Thrust Trajectory Optimization*. PhD thesis, 2016.

[21] Nicholas B. Lafarge, Kathleen C Howell, and Richard Linares. (Preprint) AAS 21-302 A Hybrid Closed-loop Guidance Strategy for Low-thrust Spacecraft enabled by Neural Networks. In *AAS*, pages 1–20, 2021.

[22] Ping Lu. What is guidance? *Journal of Guidance, Control, and Dynamics*, 44(7): 1237–1238, 2021. ISSN 15333884. doi: 10.2514/1.G006191.

[23] Spencer Boone and Jay McMahon. Orbital guidance using higher-order state transition tensors. *Journal of Guidance, Control, and Dynamics*, 44(3):493–504, 2021. ISSN 15333884. doi: 10.2514/1.G005493.

[24] David C. Folta. Lunar Frozen Orbits (Presentation). *NTRS*, 2008.

[25] Anastassios E Petropoulos and James M. Longuski. Shape-based algorithm for automated design of low-thrust, gravity-assist trajectories. *Journal of Spacecraft and Rockets*, 41(5):787–796, 2004.

[26] Bradley J. Wall and Bruce A. Conway. Shape-based approach to low-thrust rendezvous trajectory design. *Journal of Guidance, Control, and Dynamics*, 32(1):95–102, jan 2009.

[27] Nicholas Nurre and Ehsan Taheri. Multiple Gravity-Assist Low-Thrust Trajectory Design Using Finite Fourier Series. *AAS Space Flight Mechanics Meeting*, (August): 1–20, 2020.

[28] Andreas Ohndorf. *Multiphase Low-Thrust Trajectory Optimization using Evolutionary Neurocontrol*. PhD thesis, 2017.

[29] Maksim Shirobokov, Sergey Trofimov, and Mikhail Ovchinnikov. Survey of machine learning techniques in spacecraft control design. *Acta Astronautica*, 186 (May):87–97, 2021. ISSN 00945765. doi: 10.1016/j.actaastro.2021.05.018.

[30] Dario Izzo, Marcus Märtens, and Binfeng Pan. A Survey on Artificial Intelligence Trends in Spacecraft Guidance Dynamics and Control. *arXiv preprint arXiv:1812.02948*, 2018.

[31] Noble Hatten. *A Critical Evaluation of Modern Low-Thrust, Feedback-Driven Spacecraft Control Laws*. Master's thesis, The University of Texas at Austin, 2012.

[32] Hanspeter Schaub and John L. Junkins. *Analytical Mechanics of Space Systems, Fourth Edition*. American Institute of Aeronautics and Astronautics, Inc., 2018.

[33] Benjamin E Joseph. *Lyapunov Feedback Control in Equinoctial Elements Applied to Low Thrust Control of Elliptical Orbit Constellations*. Msc thesis, Massachusetts Institute of Technology, 2006.

[34] Bo J. Naasz. *Classical Element Feedback Control for Spacecraft Orbital Maneuvers*. Msc thesis, Virginia Polytechnic Institute and State University, 2002.

[35] Anastassios E Petropoulos. Simple control laws for low-thrust orbit transfers. *AAS/AIAA Astrodynamics Specialists Conference*, 2003.

[36] Anastassios E Petropoulos. Refinements to the Q-law for low-thrust orbit transfers. In *Advances in the Astronautical Sciences*, volume 120, pages 963–982, 2005.

[37] Yang Gao and Xinfeng Li. Optimization of low-thrust many-revolution transfers and Lyapunov-based guidance. *Acta Astronautica*, 66(1-2):117–129, 2010.

[38] Jackson L Shannon, Martin Ozimek, Justin Atchison, and M Christine. Q-law aided Direct Trajectory Optimization for the High-fidelity, Many-revolution Low-thrust Orbit Transfer Problem. In *AAS*, number 19, page 448, 2019.

[39] Andrea Ruggiero, Pierpaolo Pergola, Salvo Marcuccio, and Mariano Andrenucci. Low-Thrust Maneuvers for the Efficient Correction of Orbital Elements. *32nd International Electric Propulsion Conference*, pages 1–13, 2011.

[40] Harry Holt, Roberto Armellin, Andrea Scorsoglio, and Roberto Furfaro. Low-Thrust Trajectory Design Using Closed-Loop Feedback-Driven Control Laws and State-Dependent Parameters. In *AIAA Scitech 2020 Forum*, 2020.

[41] Harry Holt, Roberto Armellin, Nicola Baresi, Andrea Scorsoglio, and Roberto Furfaro. Low-Thrust Trajectory Design Using State-Dependent Closed-Loop Control Laws and Reinforcement Learning. In *AAS Astrodynamics Specialist Conference*, pages 1–19, 2020.

[42] Harry Holt, Roberto Armellin, Nicola Baresi, Yoshi Hashida, Andrea Turconi, Andrea Scorsoglio, and Roberto Furfaro. Optimal Q-laws via reinforcement learning with guaranteed stability. *Acta Astronautica*, 187:511–528, 2021. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2021.07.010.

[43] Harry Holt, Nicola Baresi, and Roberto Armellin. Towards Optimal Lyapunov Controllers for Low-Thrust Lunar Transfers Via Reinforcement Learning. In *AAS Astrodynamics Specialist Conference*, pages 1–18, 2021.

[44] Nicola Baresi, Harry Holt, Nicolò Bernardini, Xiaoyu Fu, Yang Gao, C Bridges, A Lucca Fabris, Roberto Armellin, P Murzionak, and R Kruzelecky. AAS 21-251 Mission Analysis and Design of Vmmo: the Volatile Mineralogy Mapping Orbiter. In *31st AAS SFM*, pages 1–26, 2021.

[45] Hanspeter Schaub, Srinivas R. Vadali, John L. Junkins, and Kyle T. Alfriend. Spacecraft formation flying control using mean orbit elements. *Journal of the Astronautical Sciences*, 48(1):69–87, 2001.

[46] Richard Sutton and Andrew Barto. *Reinforcment Learning: An Introduction*. MIT Press, Cambridge, 1998.

[47] Karel F. Wakker. *Fundamentals of Astrodynamics*. 2015.

[48] David Gondelach. *Orbit Prediction and Analysis for Space Situational Awareness*. PhD thesis, University of Surrey, 2019.

[49] Richard H. Battin. *An Introduction to the Mathematics and Methods of Astrodynamcis*. AIAA Education Series, New York, 1987.

[50] Peter Fortescue, Graham G. Swinerd, and John Stark. *Spacecraft Systems Engineering, Fourth Edition*. John Wiley and Sons, aug 2011.

[51] Robert G. Melton. *Fundamentals of Astrodynamics and Applications*. Number 4. 2013. doi: 10.2514/2.4291.

[52] Dirk Brouwer. Solution of the problem of artificial satellite theory without drag. *The Astronomical Journal*, 64, 1959. ISSN 00046256. doi: 10.1086/107958.

[53] Wang Sang Koon, Martin Lo, Jerrold E Marsden, and Shane D Ross. *Dynamical Systems, the Three-Body Problem and Space Mission Design (Interdisciplinary Applied Mathematics)*. 2009.

[54] NAIF. SPICE: An Observation Geometry System for Planetary Science Missions, 2017.

[55] Oliver Montenbruck and Eberhard Gill. *Satellite Orbits: Models, Methods and Applications*. Springer, 2005.

[56] Ivett A Leyva, Marcus Young, William A. Jr Hargus, Richard Van Allen, and Charles M. Zakrzwski. Propulsion Systems. In *Space Mission and Analysis Design*, volume 71. 2011.

[57] Craig A. Kluever. Simple guidance scheme for low-thrust orbit transfers. *Journal of Guidance, Control, and Dynamics*, 21(6):1015–1017, 1998.

[58] R.D. Falck, W.K. Sjauw, and D.A. Smith. Comparison of Low-Thrust control laws for application in planetocentric space. *50th AIAA/ASME/SAE/ASEE Joint Propulsion Conference 2014*, pages 1–14, 2014.

[59] F. Fumenti, M. Schlotterer, and S. Theil. Quasi-Impulsive Maneuvers to Correct Mean Orbital Elements in LEO. *CEAS EuroGNC, Specialist Conference on Guidance, Nivigation and Control*, 2015.

[60] Gabor Varga and José M Sánchez Pérez. Many-Revolution Low-Thrust Orbit Transfer Computation Using Equinoctial Q-Law Including J2 and Eclipse Effects. *Icatt 2016*, pages 2463–2481, 2016.

[61] Jackson L Shannon, Donald H. Ellison, and Christine Hartzell. Analytical Partial Derivatives of the Q-Law Guidance Algorithm. In *AAS*, pages 1–15, 2021.

[62] Seungwon Lee, Anastassios E Petropoulos, and Paul von Allmen. Low-thrust Orbit Transfer Optimization with Refined Q-law and Multi-objective Genetic Algorithm. *Advances In The Astronautical Sciences*, 2005.

[63] Slim Locoche. An Analytical Method for Evaluation of Low-thrust Multi-revolutions Orbit Transfer with Perturbations and Power Constraint. In *ESA ICATT 2021*, 2021.

[64] E. G.C. Burt. On space manoeuvres with continuous thrust. *Planetary and Space Science*, 15(1):103–122, 1967. ISSN 00320633.

[65] C.A. Maddock and M. Vasile. Extension of the proximity-quotient control law for low-thrust propulsion. *International Astronautical Federation - 59th International Astronautical Congress 2008*, *IAC 2008*, 8(July):4901–4915, 2008.

[66] Nicola Baresi, Lamberto Dell'Elce, Josue' Cardoso dos Santos, and Yasuhiro Kawakatsu. Orbit maintenance of quasi-satellite trajectories via mean relative orbit elements. *Iac-2018*, pages 1–11, 2018.

[67] Richard Epenoy and D. Pérez-Palau. Lyapunov-based low-energy low-thrust transfers to the Moon. *Acta Astronautica*, 162(June):87–97, 2019.

[68] Bindu B. Jagannatha, Jean-Baptiste H. Bouvier, and Koki Ho. Preliminary Design of Low-Energy, Low-Thrust Transfers to Halo Orbits Using Feedback Control. *Journal of Guidance, Control, and Dynamics*, 2018.

[69] Joseph T A Peterson, Sandeep Kumar Singh, John L. Junkins, and Ehsan Taheri. Lyapunov Guidance in Orbit Element Space for Low-thrust Cislunar Trajectories. In *AAS Space Flight Mechanics Meeting*.

[70] Jackson L Shannon, Martin Ozimek, Justin Atchison, and Christine Hartzell. Rapid Design and Exploration of High-Fidelity Low-Thrust Transfers to the Moon. *IEEE Aerospace Conference Proceedings*, (March), 2020. ISSN 1095323X. doi: 10.1109/AERO47225.2020.9172483.

[71] Jackson L Shannon, Donald H. Ellison, and Christine Hartzell. Exploration of Low-Thrust Lunar Swingby Escape Trajectories. In *AAS/AIAA Astrodynamics Specialist Conference*, pages 1–11, 2021.

[72] Theodore Edelbaum. How many impulses? *3rd and 4th Aerospace Sciences Meeting*, 5(11):64–69, 1967. doi: 10.2514/6.1966-7.

[73] Ehsan Taheri and John L. Junkins. How Many Impulses Redux. 2019. URL http://arxiv.org/abs/1906.01839.

[74] Da Lin Yang, Bo Xu, and Lei Zhang. Optimal low-thrust spiral trajectories using Lyapunov-based guidance. *Acta Astronautica*, 126:275–285, 2016.

[75] Tim Kovacs. *Relating Ant Colony Optimisation and Reinforcement Learning Interim Report*. PhD thesis, 2007.

[76] Tom Vodopivec, Spyridon Samothrakis, and Branko Ŝter. On monte carlo tree search and reinforcement learning. *Journal of Artificial Intelligence Research*, 60: 881–936, 2017.

[77] Luís F. Simões, Dario Izzo, Evert Haasdijk, and A. E. Eiben. Multi-rendezvous spacecraft trajectory optimization with beam P-ACO. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10197 LNCS:141–156, 2017.

[78] R A C Bianchi, C H C Ribeiro, and A H R Costa. On the Relation Between Ant Colony Optimization and Heuristically Accelerated Reinforcement Learning. *1st International Workshop on Hybrid Control of Autonomous System*, (Hycas):49–55, 2009.

[79] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, may 1992.

[80] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. pages 1–12, 2017.

[81] David Silver, Guy Lever, Nicola Hess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic Policy Gradient (DPG). *ICML*, 2014.

[82] Ivo Grondman, Lucian Busoniu, Gabriel Lopes, and Robert Babuska. A survey of actor-critic reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(6):1291–1307, 2012.

[83] Volodymyr Mnih, Adria Puigdomenech Badia, Lehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *33rd International Conference on Machine Learning, ICML 2016*, 4:2850–2869, 2016.

[84] Andrew Ng. Reinforcement Learning and Control. *Lecture Notes, Course CS229, University of Stanford*, pages 1–15.

[85] Andrea Scorsoglio. *Adaptive ZEM/ZEV feedback guidance for rendezvous in lunar NRO with collision avoidance*. Master's thesis, Politecnico Di Milano, University of Arizona, 2018.

[86] Daniel Miller. *Low-thrust Spacecraft Guidance and Control using Proximal Policy Optimization*. Master's thesis.

[87] Richard Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. *Advances in Neural Information Processing Systems 12*, pages pages 1057–1063, 1999.

[88] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, and Pieter Abbeel. Trust region policy optimization. *32nd International Conference on Machine Learning, ICML 2015*, 3:1889–1897, 2015.

[89] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. pages 1–20, 2018.

[90] Guang-Bin Huang. What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt's Dream and John von Neumann's Puzzle. *Cognitive Computation*, 7(3):263–278, 2015.

[91] Dario Izzo and Ekin Öztürk. Real-Time Guidance for Low-Thrust Transfers Using Deep Neural Networks. *Journal of Guidance, Control, and Dynamics*, 44(2):315–327, 2021. doi: 10.2514/1.G005254.

[92] Brian Gaudet and Roberto Furfaro. Robust Spacecraft Hovering Near Small Bodies in Environments with Unknown Dynamics Using Reinforcement Learning. *AIAA/AAS Astrodynamics Specialist Conference*, (May 2018), 2012.

[93] Brian Gaudet, Richard Linares, and Roberto Furfaro. Deep Reinforcement Learning for Six Degree-of-Freedom Planetary Powered Descent and Landing. *arXiv preprint arXiv:1810.08719*, 2018.

[94] Brian Gaudet, Richard Linares, and Roberto Furfaro. Six Degree-of-Freedom Hovering using LIDAR Altimetry via Reinforcement Meta-Learning. pages 1–15, 2019. URL `http://arxiv.org/abs/1911.08553`.

[95] Andrea Scorsoglio, Andrea D'Ambrosio, Luca Ghilardi, Brian Gaudet, Fabio Curti, and Roberto Furfaro. Image-Based Deep Reinforcement Meta-Learning for Autonomous Lunar Landing. *Journal of Spacecraft and Rockets*, (May):1–13, 2021. ISSN 0022-4650. doi: 10.2514/1.a35072.

[96] Kirk Hovell and Steve Ulrich. Deep Reinforcement Learning for Spacecraft Proximity Operations Guidance. *Journal of Spacecraft and Rockets*, 58(2):254–264, 2021.

[97] Daniel Miller and Richard Linares. Low-Thrust Optimal Control Via Reinforcement Learning. In *AAS*, number February, pages 1–20, 2019.

[98] Nicholas B. LaFarge, Daniel Miller, Kathleen C Howell, and Richard Linares. Guidance for Closed-Loop Transfers using Reinforcement Learning with Application to Libration Point Orbits. In *AIAA Scitech 2020 Forum*.

[99] Nicholas B LaFarge, Kathleen C Howell, and David C Folta. *An Autonomous Stationkeeping Strategy for Multi-Body Orbits Leveraging Reinforcement Learning*. 2022. doi: 10.2514/6.2022-1764.

[100] Christopher J Sullivan and Natasha Bosanac. Using Reinforcement Learning to Design a Low-Thrust Approach into a Periodic Orbit in a Multi-Body System. In *AIAA Scitech 2020 Forum*, 2020.

[101] Natasha Bosanac, Stefano Bonasera, Christopher J Sullivan, Jay Mcmahon, and Nisar Ahmed. Reinforcement Learning for Reconfiguration Maneuver Design in Multi-Body Systems. In *AAS Astrodynamics Specialist Conference*, pages 1–20, 2021.

[102] Stefano Bonasera, Ian Elliott, Christopher J Sullivan, Natasha Bosanac, Nisar Ahmed, and Jay Mcmahon. AAS 21-216 Designing Impulsive Station-keeping Maneuvers near a Sun-Earth L2 Halo Orbit via Reinforcement Learning. In *AAS*, pages 1–20, 2021.

[103] Kanta Yanagida, Naoya Ozaki, and Ryu Funase. Exploration of Long Time-of-Flight Three-Body Transfers Using Deep Reinforcement Learning. In *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics (AIAA), 2020.

[104] Daniel Miller, Jacob A. Englander, and Richard Linares. Interplanetary low-thrust design using proximal policy optimization. *Advances in the Astronautical Sciences*, 171:1575–1592, 2020. ISSN 00653438.

[105] Alessandro Zavoli and Lorenzo Federici. Reinforcement Learning for Robust Trajectory Design of Interplanetary Missions. *Journal of Guidance, Control, and Dynamics*, 44(8):1440–1453, 2021. doi: 10.2514/1.G005794.

[106] Hyeokjoon Kwon, Snyoll Oghim, and Hyochoong Bang. AAS 21-315 Autonomous Guidance for multi-revolution low-thrust orbit transfer via Reinforcement Learning. In *AAS*, pages 1–16, 2021.

[107] Haiyang Li, Shiyu Chen, Dario Izzo, and Hexi Baoyin. Deep Networks as Approximators of Optimal Transfers Solutions in Multitarget Missions. *arXiv preprint arXiv:1902.00250*, 2019.

[108] Tenavi Nakamura-Zimmerer, Qi Gong, and Wei Kang. Adaptive deep learning for high-dimensional hamilton-jacobi-bellman equations. *arXiv*, pages 1–25, 2019.

[109] Mauro Pontani and Marco Pustorino. Nonlinear Earth orbit control using low-thrust propulsion. *Acta Astronautica*, 179:296–310, 2021.

[110] Abolfazl Shirazi, Harry Holt, Roberto Armellin, and Nicola Baresi. Time-Varying Lyapunov Control Laws with Enhanced Estimation of Distribution Algorithm for Low-Thrust Trajectory Design. In *Modeling and Optimization in Space Engineering – New Concepts and Approaches (in preparation 2022)*. Springer.

[111] Hongkai Dai, Benoit Landry, Lujie Yang, Marco Pavone, and Russ Tedrake. Lyapunov-stable neural-network control. *Robotics: Science and Systems*, 2021.

[112] Priya L. Donti, Melrose Roderick, Mahyar Fazlyab, and J. Zico Kolter. Enforcing robust control guarantees within neural network policies. In *ICLR*, pages 1–26, 2021.

[113] Gabor Varga and José M Sánchez Pérez. Many-revolution low-thrust orbit transfer computation using equinoctial Q-law including J2 and eclipse effects. In *6th International Conference on Astrodynamics Tools and Techniques*, 2016.

[114] Sophie Geffroy and Richard Epenoy. Optimal low-thrust transfers with constraints - Generalization of averaging techniques. *Acta Astronautica*, 41(3):133–149, 1997.

[115] Slim Locoche, Kristen Lagadec, Sven O. Erb, and Celia Yabar Valles. Operational Concepts With Reduced Cost Enabled By Autonomous Guidance For Electrical Orbit Raising. In *1st European Workshop on Space Flight Dynamics Services, Systems and Operations (2021)*, 2021.

[116] Kyle J. DeMars and Moriba K. Jah. Probabilistic initial orbit determination using Gaussian mixture models. *Journal of Guidance, Control, and Dynamics*, 36(5):1324–1335, 2013.

[117] Roberto Furfaro, Andrea Scorsoglio, Richard Linares, and Mauro Massari. Adaptive generalized ZEM-ZEV feedback guidance for planetary landing via a deep reinforcement learning approach. *Acta Astronautica*, 171:156–171, 2020.

[118] Carlos Aguilar Ibañez, O. Gutiérrez Frias, and M. Suárez Castañón. Lyapunov-based controller for the inverted pendulum cart system. *Nonlinear Dynamics*, 40 (4):367–374, 2005. ISSN 0924090X. doi: 10.1007/s11071-005-7290-y.

[119] Byron D. Tapley, Bob E. Schutz, and George H. Born. *Statistical Orbit Determination*. 2004. doi: 10.1016/B978-0-12-683630-1.X5019-X.

[120] M. Pavone, B. Açıkmese, I.A. Nesnas, and J. Starek. Spacecraft Autonomy Challenges for Next Generation Space Missions. *Springer Lecture Notes in Control and Information Sciences*, pages 1–34, 2014. URL `http://web.stanford.edu/{~}pavone/papers/Pavone.Acikmese.ea.LNS14.pdf`.

[121] Zhengfan Zhu, Qingbo Gan, Xin Yang, and Yang Gao. Solving fuel-optimal low-thrust orbital transfers with bang-bang control using a novel continuation technique. *Acta Astronautica*, 137:98–113, 2017.

[122] SSTL. Telesat LEO Phase 1 satellite: Launched 2018, 2018. URL `https://www.sstl.co.uk/space-portfolio/launched-missions/2010-2019/telesat-leo-phase-1-satellite-launched-2018`.

[123] Peter Peterson, Daniel A. Herman, Hani Kamhawi, Jason Frieman, Wensheng Huang, Tim Verhey, Dragos Dinca, Kristen Boomer, Luis Pinero, Kenneth Criswell,

Scott Hall, Arthur G. Birchenough, James Gilland, Richard Robert Hofer, James E. Polk, Vernon H. Chaplin, Robert Bryant Lobbia, Charles E. Garner, and Matthew Kowalkowski. Overview of NASA's Solar Electric Propulsion Project. *36th International Electric Propulsion Conference*, pages IEPC–2019–836, 2019.

[124] Jun'ichiro Kawaguchi, Akira Fujiwara, and Tono K Uesugi. *The Ion Engines Cruise Operation and the Earth Swingby of 'Hayabusa' (MUSES-C)*. 2012. doi: 10.2514/6.IAC-04-Q.5.02.

[125] Yuichi Tsuda, Makoto Yoshikawa, Masanao Abe, Hiroyuki Minamino, and Satoru Nakazawa. System design of the hayabusa 2-asteroid sample return mission to 1999 JU3. *Acta Astronautica*, 91:356–362, 2013. doi: 10.1016/j.actaastro.2013.06.028.

[126] John T. Betts and Sven O. Erb. Optimal low thrust trajectories to the moon. *SIAM Journal on Applied Dynamical Systems*, 2(2):144–170, 2003.

[127] Rita Neves and Joan Pau SÃ¡nchez. Gauss' variational equations for low-thrust optimal control problems in low-energy regimes. *Proceedings of the International Astronautical Congress, IAC*, 2018-Octob(October):1–5, 2018. ISSN 00741795.

[128] Dong Eui Chang, David F. Chichka, and Jerrold E Marsden. Lyapunov-based transfer between elliptic Keplerian orbits. *Discrete and Continuous Dynamical Systems - Series B*, 2(1):57–67, 2002.

[129] Andrea Scorsoglio, Roberto Furfaro, Richard Linares, and Mauro Massari. Actor-critic reinforcement learning approach to relative motion guidance in near-rectilinear orbit. *Advances in the Astronautical Sciences*, pages Paper AAS–19–441 pp. 1–20, 2019.

[130] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *arXiv preprint arXiv:1012.2599v1*, 2010.

[131] Brian Gaudet, Richard Linares, and Roberto Furfaro. Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta Astronautica*, 169:180–190, 2020. ISSN 00945765. doi: 10.1016/j.actaastro.2020.01.007.

[132] Xiaoyu Liu, Colin McInnes, and Matteo Ceriotti. AAS 19-390 Secular orbital element variations due to continuous low-thrust control and third-body perturbations. In *AAS Astrodynamics Specialist Conference*, 2019.